

VARCHART XGantt

.NET Edition 5.2
User's and
Reference Guide



VARCHART XGantt .NET Edition

Version 5.2

User' s Guide

NETRONIC Software GmbH
Pascalstrasse 15
52076 Aachen
Germany
Phone +49 (0) 2408 141-0
Fax +49 (0) 2408 141-33
Email sales@netronic.com
www.netronic.com

© Copyright 2020 NETRONIC Software GmbH
All rights reserved.

Information in this document is subject to change without notice and does not represent a commitment on the part of NETRONIC Software GmbH. The software described in this document is furnished under a license agreement. The software may be used or copied only in accordance with the terms of the agreement. It is against the law to copy documentation on magnetic tape, disk, or any other medium for any purpose other than the purchaser's personal use.

Microsoft Windows, Microsoft Explorer, Microsoft Visual Basic and Microsoft Visual Studio are trademarks of MICROSOFT Corp., USA.

Last Revision: 27 April 2020

Table of Contents

1	Introduction	11
1.1	VARCHART XGantt at a Glance	11
1.2	Installation	13
1.3	Licensing	16
1.4	Delivery	17
1.5	Usage of the German version	19
1.6	Support and Advice	21
2	Tutorial	23
2.1	Overview	23
2.2	Placing the Control on a Form	25
2.3	Supplying Data	27
2.4	Calculating End Dates	33
2.5	Marking Non-working Intervals in Activities	37
2.6	Interactions in the Table and Diagram Area	39
2.7	Interactions with Activities	41
2.8	Using Layers	43
2.9	Using Filters	46
2.10	Creating Histograms	50
2.11	Printing the Diagram	68
2.12	Exporting a Diagram	69
2.13	Saving the Configuration	70
3	Important Concepts	71
3.1	Boxes	71
3.2	Data Tables	75
3.3	Date Lines	87
3.4	Dates and Daylight Saving Time	92
3.5	Drag & Drop	94

4 Table of Contents

3.6	Dragging Tools	96
3.7	Events	108
3.8	Filters	109
3.9	Graphics Formats	111
3.10	Grouping	115
3.11	Hierarchical Order	121
3.12	Histograms	124
3.13	How to Use a Calendar	131
3.14	Interaction Events	151
3.15	Layers	160
3.16	Legend View	163
3.17	Link Appearance	165
3.18	Links	166
3.19	Live Update	170
3.20	Localization of Text Output	176
3.21	Maps	178
3.22	MultiState Fields	183
3.23	Node (Activity)	185
3.24	Platforms x86 and x64	187
3.25	Resource Scheduler	190
3.26	Schedule	194
3.27	Security Guidelines for the Deployment in the Internet Explorer	197
3.28	Sorting	199
3.29	Table	205
3.30	Time Scale	207
3.31	Tooltips during Runtime	213
3.32	Unicode	214
3.33	Using the Control in a Browser Environment	215
3.34	Viewer Metafile (*.vmf)	219
3.35	World View	220
3.36	Writing PDF files	221

4	Property Pages and Dialog Boxes	223
4.1	General Information	223
4.2	The "General" Property Page	224
4.3	The "Border Area" Property Page	235
4.4	The "Nodes" Property Page	237
4.5	The "Additional Views" Property Page	243
4.6	The "Layout" Property Page	247
4.7	The "Objects" Property Page	251
4.8	The "Links" Property Page	253
4.9	The "Schedule" Property Page	255
4.10	The "Administrate Update Behaviors" Dialog Box	257
4.11	The "Edit Update Behaviors" Dialog Box	258
4.12	The "Administrate Data Tables" Dialog Box	260
4.13	The "Specify Bar Appearance" Dialog Box	263
4.14	The "Edit Layer" Dialog Box	267
4.15	The "Edit Layer Format" Dialog Box	272
4.16	The "Administrate Filters" Dialog Box	276
4.17	The "Edit Filter" Dialog Box	278
4.18	The "Administrate Line formats" Dialog Box	282
4.19	The "Edit Line format" Dialog Box	284
4.20	The "Grouping" Dialog Box	288
4.21	The "Administrate Calendar grids" Dialog Box	298
4.22	The "Administrate Line grids" Dialog Box	300
4.23	The "Administrate Maps" Dialog Box	303
4.24	The "Edit Map" Dialog Box	305
4.25	The "Configure Mapping" Dialog Box	307
4.26	The "Administrate Boxes" Dialog Box	308
4.27	The "Edit Box" Dialog Box	313
4.28	The "Administrate Box Formats" Dialog Box	314
4.29	The "Edit Box Format" Dialog Box	316
4.30	The "Administrate Link Appearances" Dialog Box	319
4.31	The "Specify Table" Dialog Box	323
4.32	The "Edit Table" Dialog Box	325

6 Table of Contents

4.33	The "Edit Table Format" Dialog Box	327
4.34	The "Edit Line Attributes" Dialog Box	332
4.35	The "Edit Pattern Attributes" Dialog Box	333
4.36	The "Specify Calendars" Dialog Box	334
4.37	The "Administrate Intervals" Dialog Box (Calendar)	336
4.38	The "Administrate Calendar Profiles" Dialog Box	338
4.39	The "Administrate Intervals" Dialog Box (Calendar Profiles, Profile Type <Day Profile>)	340
4.40	The "Administrate Intervals" Dialog Box (Calendar Profiles, Profile Type <Week Profile>)	342
4.41	The "Administrate Intervals" Dialog Box (Calendar Profiles, Profile Type <Variable Profile>)	343
4.42	The "Administrate Intervals" Dialog Box (Calendar Profiles, Profile Type <Year Profile>)	345
4.43	The "Specify Time Scale" Dialog Box	347
4.44	The "Edit Time Scale Section" Dialog Box	350
4.45	The "Administrate Histograms" Dialog Box	356
4.46	The "Edit Histogram" Dialog Box	358
4.47	The "Select Curve Data Source" Dialog Box	362
4.48	The "Select Ribbon Type" Dialog Box	363
4.49	The "Specify Date Lines" Dialog Box	365
4.50	The "Edit Date Line" Dialog Box	368
4.51	The "Specification of Texts, Graphics and Legend" Dialog Box	370
4.52	The "Legend Attributes Dialog Box"	373
4.53	The "Licensing" Dialog Box	375
4.54	The "Request License Information" Dialog Box	377

5 User Interface 379

5.1	Overview	379
5.2	Navigation in the Diagram and in the Table	381
5.3	Zooming	382
5.4	Marking Nodes or Layers	383
5.5	Creating Nodes	384
5.6	Moving Nodes by Mouse	385

5.7	Moving Nodes and Modify Duration by Keys	387
5.8	Moving Layers	388
5.9	Change Start/End Date	389
5.10	Delete, Cut, Copy and Paste Nodes	390
5.11	Edit Node Data	391
5.12	Edit Links	393
5.13	Anchor Box to Node	394
5.14	Edit Group Data	396
5.15	Collapsing/Expanding Groups	397
5.16	Moving Groups	398
5.17	Editing Fields in the Table	399
5.18	Modifying Table Column Width	400
5.19	Modifying the Table/Diagram Ratio	401
5.20	Inserting table rows	402
5.21	Modifying the Timescale	403
5.22	Modifying the Scaling and the Borders of Sections	405
5.23	Moving the Date Line	406
5.24	Setting up pages	407
5.25	Print Preview	412
5.26	Context Menu of the Diagram	415
5.27	Context Menu of Nodes	420
5.28	Context Menu of Links	422
5.29	Context Menu of Groups	423
5.30	Context Menu of the Time Scale	425
5.31	Context Menu of the Curve	426
5.32	Context Menu of the Legend	428
5.33	Context Menu of Boxes	429
<hr/>		
6	Frequently Asked Questions	431
6.1	How to Upgrade from VARCHART XGantt .NET 4.4 to VARCHART XGantt .NET 5.0?	431
6.2	How to Upgrade from one Build of VARCHART XGantt .NET to a new one (within the same version)?	433

8 Table of Contents

6.3	Why does an error message occur, when I create a new project in Visual Studio 2010 and try to drag the control onto the form?	435
6.4	How can I Activate the License File?	436
6.5	How can I Limit the Timescale Width?	437
6.6	How can I Move a Bar into the Visible Area by Clicking on the Table?	438
6.7	How can I Make Overlapping Activities in a Group Visible?	439
6.8	How can I Save and Reload the Order of Activities?	440
6.9	Why can I not Create Nodes Interactively at Times?	441
6.10	How can I Disable the Default Context Menus?	442
6.11	How can I Improve the Performance?	443
6.12	What to do if the Control Does Not Work With a User Account of a Computer	446
6.13	Can All Fonts be Used?	447

7 API Reference 449

7.1	Object Types	449
7.2	VcBorderArea	452
7.3	VcBorderBox	454
7.4	VcBox	462
7.5	VcBoxCollection	478
7.6	VcBoxFormat	485
7.7	VcBoxFormatCollection	490
7.8	VcBoxFormatField	497
7.9	VcCalendar	505
7.10	VcCalendarCollection	514
7.11	VcCalendarGrid	521
7.12	VcCalendarGridCollection	538
7.13	VcCalendarProfile	545
7.14	VcCalendarProfileCollection	548
7.15	VcCurve	554
7.16	VcCurveCollection	588
7.17	VcDataDefinitionField	595
7.18	VcDataDefinitionTable	600

7.19	VcDataRecord	606
7.20	VcDataRecordCollection	613
7.21	VcDataTable	622
7.22	VcDataTableCollection	625
7.23	VcDataTableField	631
7.24	VcDataTableFieldCollection	638
7.25	VcDateLine	644
7.26	VcDateLineCollection	656
7.27	VcDateLineGrid	664
7.28	VcDateLineGridCollection	676
7.29	VcFilter	683
7.30	VcFilterCollection	690
7.31	VcFilterSubCondition	696
7.32	VcGantt	701
7.33	VcGroup	971
7.34	VcGroupCollection	982
7.35	VcGroupLevelLayout	986
7.36	VcGroupLevelLayoutCollection	1009
7.37	VcHierarchyLevelLayout	1014
7.38	VcHistogram	1021
7.39	VcHistogramCollection	1032
7.40	VcInfoWindow	1037
7.41	VcInterval	1044
7.42	VcIntervalCollection	1060
7.43	VcLayer	1065
7.44	VcLayerCollection	1108
7.45	VcLayerFormat	1114
7.46	VcLayerFormatField	1117
7.47	VcLegendView	1125
7.48	VcLineFormat	1133
7.49	VcLineFormatCollection	1136
7.50	VcLineFormatField	1142
7.51	VcLink	1154
7.52	VcLinkAppearance	1160

10 Table of Contents

7.53	VcLinkAppearanceCollection	1169
7.54	VcLinkCollection	1176
7.55	VcMap	1180
7.56	VcMapCollection	1187
7.57	VcMapEntry	1194
7.58	VcNode	1204
7.59	VcNodeCollection	1216
7.60	VcNodeLevelLayout	1220
7.61	VcNumericScale	1229
7.62	VcNumericScaleCollection	1244
7.63	VcPrinter	1249
7.64	VcRect	1274
7.65	VcResourceScheduler2	1278
7.66	VcRibbon	1344
7.67	VcScheduler	1358
7.68	VcSection	1366
7.69	VcTable	1372
7.70	VcTableCollection	1377
7.71	VcTableFormat	1380
7.72	VcTableFormatCollection	1387
7.73	VcTableFormatField	1391
7.74	VcTimeScale	1407
7.75	VcTimeScaleCollection	1413
7.76	VcUpdateBehavior	1417
7.77	VcUpdateBehaviorCollection	1420
7.78	VcUpdateBehaviorContext	1427
7.79	VcWorldView	1431

8	Index	1439
----------	--------------	-------------

1 Introduction

1.1 VARCHART XGantt at a Glance

Gantt charts allow to display and plan the chronological sequence of tasks and the capacity of resources. Due to their graphical visualization, interrelations and changes become obvious at a glance. Besides being employed in the project management, Gantt diagrams have been established above all in control panels of the manufacturing and in systems of resource management and disposition.

VARCHART XGantt is an interactive graphic component which can easily be integrated into your own applications within short time because there is no time-consuming programming of graphical charts. Due to the great variety of layout options, VARCHART XGantt meets individual graphical demands. The print-out is of first-class quality.

VARCHART XGantt .NET is a Windows Forms control which was completely syntonized to the Microsoft .NET framework.

> The functionalities of VARCHART XGantt are:

- Creating, deleting or shifting of nodes
- Creating and deleting of links
- Visualization of date fields by bars or symbols
- Data driven allocation of graphical attributes
- Sorting and grouping according to various criteria
- Collapsing or expanding of groups of activities
- Variable structure of the time scale
- Flexible design of the table area
- Adding of date lines and line grids
- Continuous zooming of diagrams
- Zooming of diagram sections to full screen size
- Integrated page preview and print-out with paging
- Exchange of the application data via files or the programming interface
- Various design options for histograms

12 Introduction

- Easy customization of properties via the property pages
- Customization of default interactions via events
- Powerful programming interface

Note: The source code samples of this documentation are written in VB.NET and C#.

1.2 Installation

To develop an application on the basis of .NET you need a development environment such as Microsoft Visual Studio 2010 and upwards that supports the .NET framework 2.0 at least and is compatible with mixed-mode components. As operating system only the 32bit or 64bit (x64) editions of Windows from XP Service Pack 3 upwards can be used.

Before installing

- **If you want to upgrade from XGantt 4.4 to XGantt 5.0**

Please perform the steps as described in in chapter 6.1 "How to to Upgrade from VARCHART XGantt .NET 4.4 to VARCHART XGantt .NET 5.0?"

- **If you want to upgrade to a new build within the same version**

- Please performe the steps as described in in chapter 6.2 "How to Upgrade from one Build of VARCHART XGantt .NET to a new one (within the same version)?"

Installation

To install the VARCHART XGantt .NET control on your computer, please start the setup program and follow the instructions.


By default, the control and ist associated files will be stored below the folder

c:\Program Files\NETRONIC (32bit-Windows) or

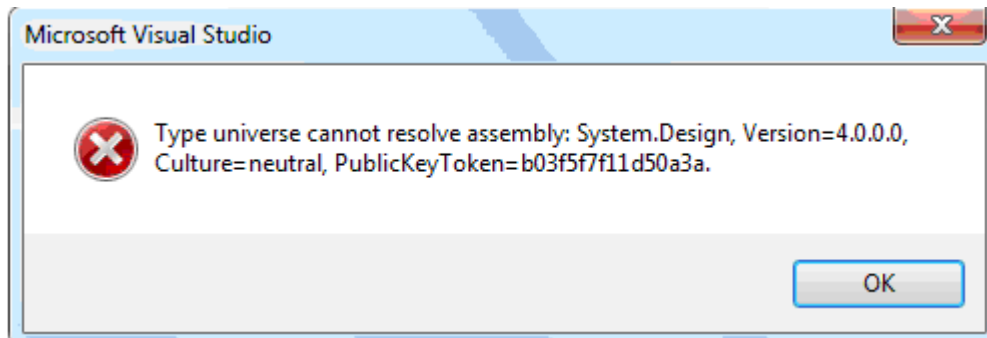
c:\Program Files (x86)\NETRONIC (64bit-Windows).

After installing you should add the control to the toolbox of your developing environment.

We give an example of how to proceed in Microsoft Visual Studio; in other programming environments the procedure is similar:

1. In Visual Studio create a new project of the type **Windows Application**. It doesn't matter which language you choose, but please mind that the toolbox be visible. If it is not, click on **View Toolbox**.
2. Open the context menu by a right mouse click on the toolbox and select **Choose Items**.
3. By clicking on **Browse** of the tab **.NET Framework Components** you can choose the assembly **NETRONIC.XGantt.dll** from the installation directory. After confirming by **OK** the icon of VARCHART XGantt .NET  will be added to the toolbox.

4. Important for the users of **Visual Studio 2010**: **Before** you drag the control to the form, you have to change the target framework from **.NET Framework Client Profile** to **.NET Framework 4** in the **Application Settings (C#)** or **Advanced Compiler Settings (VB)** since the former lacks the `System.Design.dll`, which is required by the property pages at design-time. If you don't change the framework, the following error message will pop up when you try to drag the control onto the form:



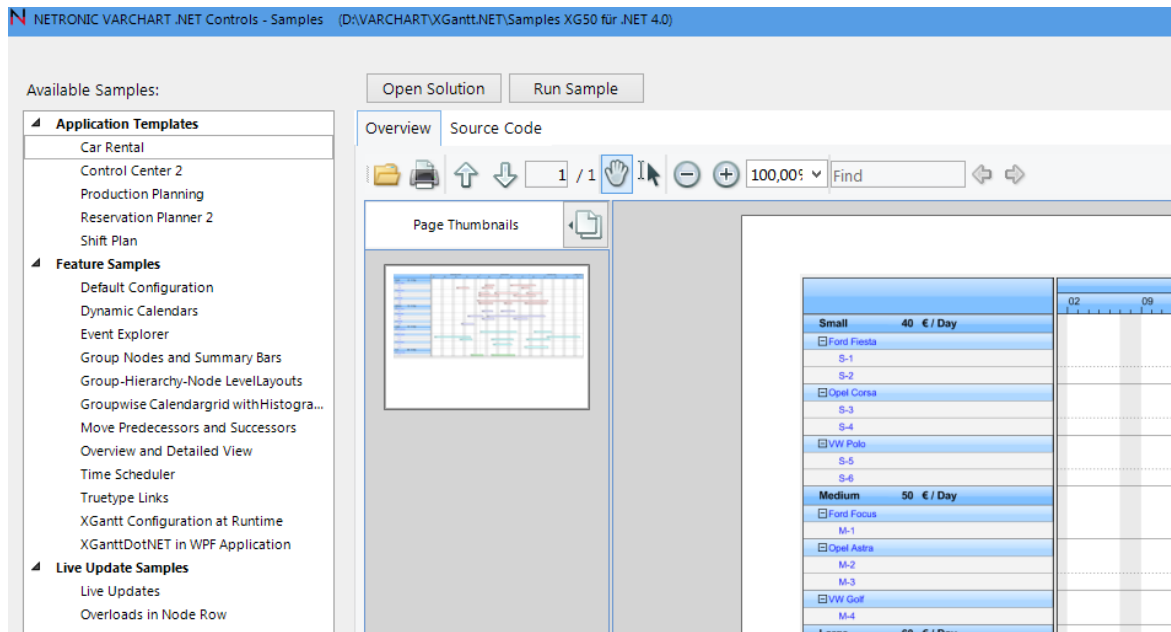
Alternatively, you can make an unattended installation of VARCHART XGantt. For this, please enter:

```
start/wait (NameOfTheSetupFile).exe /L1033 /s /V"/qn ADDLOCAL=ALL"
```

By this call, the installation will run without user interaction and without status information displayed on the screen. Please note:

1. The invoking procedure, such as a DOS box, needs to be run with administrator privileges; otherwise a UAC message may appear that requests a user entry.
2. Language parameters: /L1033: installation in English; /L1031: installation in German
3. Progress information: /qb: progress information will be displayed; /qn: no progress information will appear; you won't see anything on the screen.
4. Start/wait you should use in case the installation is run by a batch file; if you don't use 'wait', the batch file will run parallel to the installation.

During the setup of VARCHART XGantt a sample collection will be installed as well, the symbol of which being placed on the desktop. Double-click this symbol to open the sample collection.



To run a sample, mark it in the list and click the according button **Run Sample**. To view the source code or the current configuration of XGantt on the property pages, click **Open Solution**. The source code can, however, always be viewed in text mode on the **Source Code** tab.

The samples are grouped. The first group **Application Templates** contains some practical examples of use:


1. Car Rental: Leasing orders for vehicles are scheduled and managed
2. Control Center 2: You can schedule orders on machines that are grouped. The orders are stored in an Access data base.
3. Production Planning: A classic production planning
4. Reservation Planner 2. Manage training rooms and trainers for seminars and have conflicts visualized graphically.
5. Shift Plan: Assign different shifts to your employees.

The group **Feature Samples** presents some showcase features of XGantt.

The examples of the **Live Update Samples** illustrate how the consequences of a mouse interaction are being visualized immediately during the action and not only upon ending it.

1.3 Licensing

1.3.1 Developer Licenses

For licensing the VARCHART XGantt control please click the icon  and draw the control onto the form.

Open the **Property Pages** by a right mouse click on the control.

On the **General** tab, please open the licensing dialog by clicking on the **Licensing...** button.

By clicking on the button **Request license information from NETRONIC** the according dialog will open.

Three items are needed for the registration:

- the license number
- your name
- the name of the company

Please fill in the information needed. You will find the license number "BXnnnn" on the delivery note of your order.

If you click on **Send email to NETRONIC...**, an email will be generated that only needs to be dispatched. Alternatively, you can write an email manually that contains the required information. Please send all enquiries concerning the licensing to license@netronic.com.

After sending the mail, you will immediately receive a license file. To finish the licensing procedure, please copy the file to the installation directory (directory that contains the file **NETRONIC.XGantt.dll**).

1.4 Delivery

If you wish to deliver to a customer an application developed by yourself having used XGantt .NET, the following files need to be delivered with the application. All other files belonging to VARCHART XGantt .NET are only used during the phase of development and must **not** be passed on to your customers.

> Framework .NET 2.0/3.0/3.5

- **In the according processor version for x86 or x64**

NETRONIC.XGantt.dll

NETRONIC.XGanttd.dll (if you want to use the German version)

NETRONIC.XGanttc.dll (if you want to use the Chinese version)

mfc80u.dll

mfc80u.dll

msvc80.dll

msvcr80.dll

In order to install the libraries *mfc80u.dll*, *msvc80.dll*, *mfc80u.dll* and *msvcr80.dll* please use the setup file *vc80_x86.exe* or *vc80_x64.exe* respectively. You will find these files in the installation folder of XGantt .NET in the subfolder **redist**.

For further information please see:

[msdn2.microsoft.com/en-us/library/ms235285\(VS.80\).aspx](http://msdn2.microsoft.com/en-us/library/ms235285(VS.80).aspx).

- **If your application uses the resource scheduling module:**

- for the **x86** version: *opsaps.dll*
- for the **x64** version: *opsaps64.dll*

> Framework .NET 4.0/4.5

- **In the according processor version for x86 or x64**

NETRONIC.XGantt.dll

NETRONIC.XGanttd.dll (if you want to use the German version)

NETRONIC.XGanttc.dll (if you want to use the Chinese version)

mfc100u.dll

18 Introduction

mfc100u.dll

msvc100.dll

msvcr100.dll

In order to install the libraries *mfc100u.dll*, *msvc100.dll*, *mfc100u.dll* and *msvcr100.dll* you can either copy them directly to the Windows system directory or you can use the setup file *vc_redist_vs2010_x86.exe* or *vc_redist_vs2010_x64.exe* respectively. You find these files in the installation folder of XGantt .NET in the subfolder **redist**.

- **If your application uses the resource scheduling module:**

- for the **x86** version: opsaps.dll
- for the **x64** version: opsaps64.dll

VARCHART XGantt .NET can be run on the the below platforms:

- Windows 8
- Windows 7
- Windows Server 2008
- Windows Vista
- Windows Server 2003
- Windows XP SP3 or later

using the .NET framework 2.0 at least (for further information, see

msdn.microsoft.com/netframework/technologyinfo/sysreqs/default.aspx)

Tip:

How to check which .NET Framework is already installed:

In the **Control Panel** double click on the **Software** icon and look for 'Microsoft .NET Framework' in the list of applications.

1.5 Usage of the German version

The VARCHART XGantt .NET Edition is available in German, English and in (Simplified) Chinese. When installing the German or Chinese version, the resource assembly NETRONIC.XGantt~~d~~**c**.dll or NETRONIC.XGantt~~d~~**c**.dll is copied to the installation directory in addition to the control assembly NETRONIC.XGantt.dll.

Usage at design time

If the **Regional Options** (Control Panel, Regional and Language Options) were set to **German/Chinese**, the resource assembly is loaded from the installation directory and the German/Chinese dialogs and property pages are available at design time.

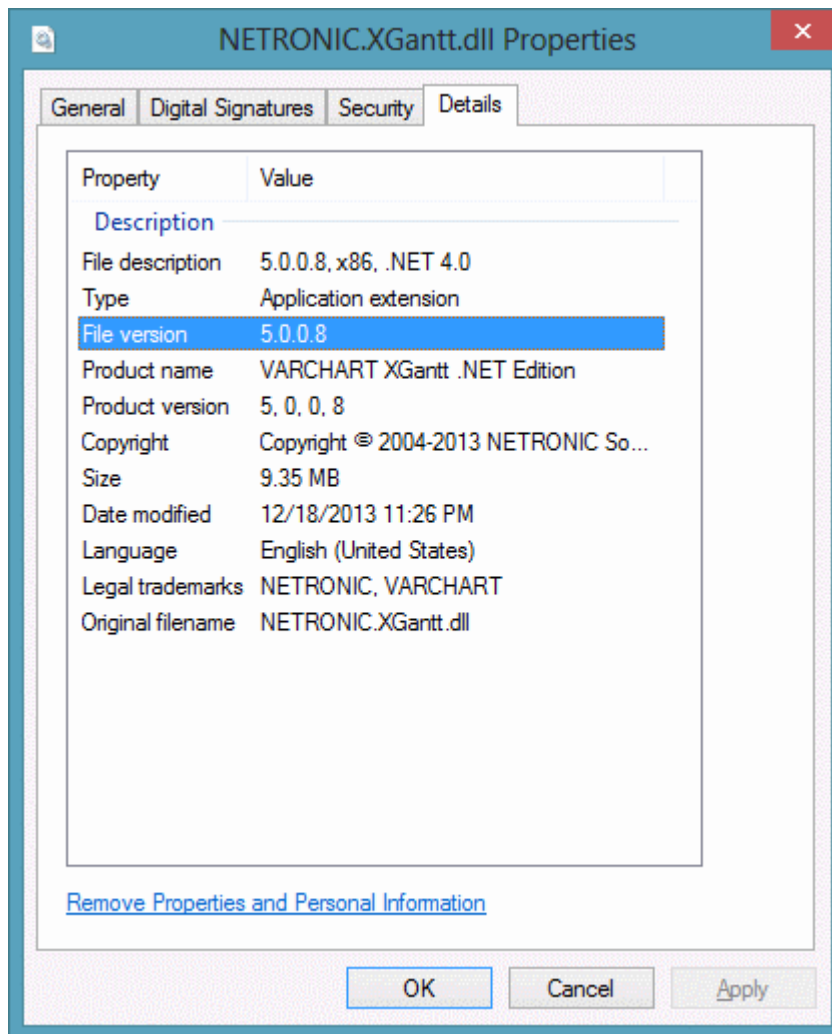
Usage at run time

If you want to make sure that the resource assembly is used at run time as well and German/Chinese dialogs are available you have to copy the resource assembly to the application directory. For this, a reference to the assembly has to be added in the project ("Add Reference").

Tip: Because the development environment sets the parameter "Copy-Local" to **False** by default, you will have to set it to **True** manually. When the solution is rebuilt afterwards, the resource assembly is copied to the according application directory and will be loaded from there.

In case of problems you should check whether the file version numbers of the assemblies match (Windows Explorer, context menu of the file, **Properties**, tab **Version**).

20 Introduction



1.6 Support and Advice

Are you wondering whether VARCHART XGantt is going to meet the special requirements of your Gantt chart?

Are you trying to make a plan of how much effort it could be to program a special feature of your Gantt chart?

Have you just started testing VARCHART XGantt and are you wondering how to get to a special feature of your Gantt chart?

We would be glad to assist you with any queries you may have. Please contact

NETRONIC Software GmbH

Pascalstr. 15

52076 Aachen

Germany

Phone +49-2408-141-0

Fax +49-2408-141-33

Email support@netronic.com

www.netronic.com

...by the way: you may order our support and maintenance service that lasts longer than the 30 days of free support during the initial testing phase. The service includes:

- A support hotline
- Detailed expert advice to questions of application
- Quick fixing of possible bugs in the software
- Upgrades to new VARCHART XGantt releases for development and runtime versions.

We also offer training classes and workshops (at your or at our place).

2 Tutorial

2.1 Overview

In this tutorial, we will get you acquainted with the fundamentals of VARCHART XGantt that are essential for integrating a bar chart into your own web application.

Step by step, we will explain to you aspects of VARCHART XGantt that are important for the development of an application and we will introduce the wide range of design options to you. We recommend to read this tutorial chapter by chapter, while the other parts of the user guide rather serve for consulting on specific situations.

- **Property pages and dialogs**

In this chapter you will find comprehensive information on the property pages and dialogs which allow to configure VARCHART XGantt at design time without having to write a single code line.

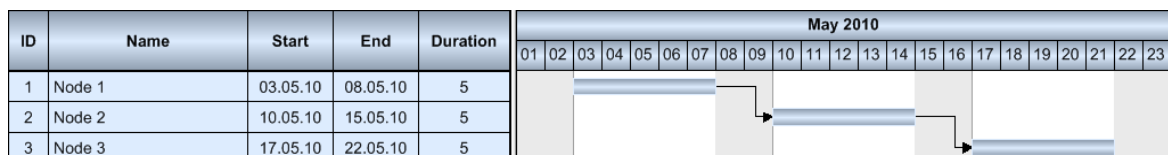
- **Elements of the user interface**

In this chapter the inbuilt interactions of the diagram are described. Details of the user interface can be fitted or changed individually.

- **API Reference**

In the above chapter you will find detailed information on all objects, properties, methods and events of VARCHART XGantt.

As the developing environment for the code samples, we use Visual Studio .NET 2005. Our first program sample will show the below result:

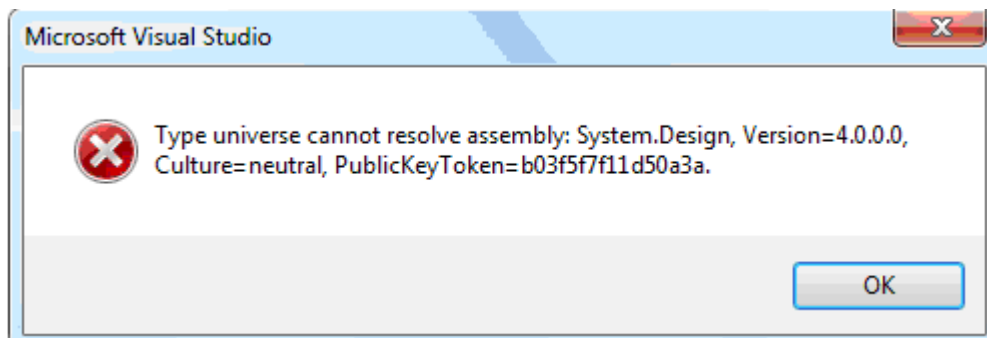



You will find the starter sample in the folder **UserGuideSamples\VB.NET\XGantt_Tutorial01_App** or **UserGuideSamples\Csharp\XGantt_Tutorial01_App**.

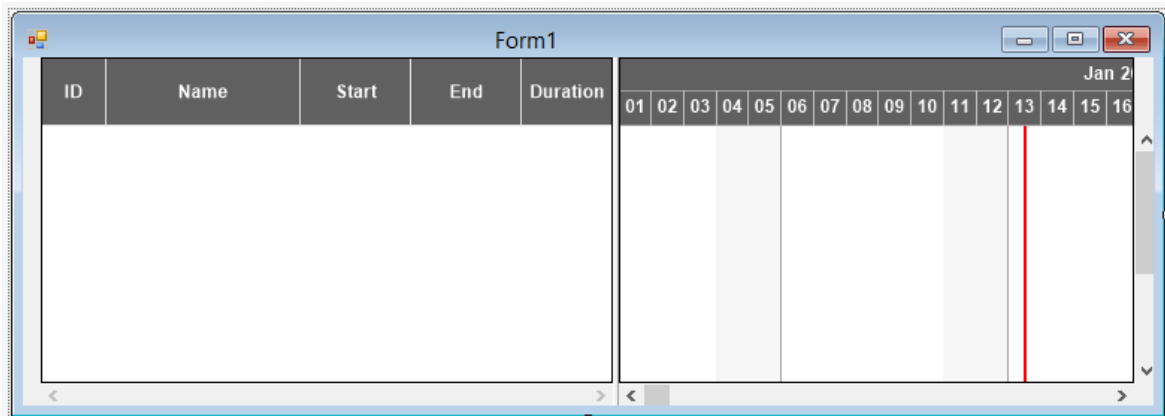
The program sample will primarily demonstrate the inbuilt interactions of VARCHART XGantt.

2.2 Placing the Control on a Form

Important for the users of **Visual Studio 2010!:** Before you drag the control to the form, you have to change the target framework from **.NET Framework Client Profile** to **.NET Framework 4** in the **Application Settings (C#)** or **Advanced Compiler Settings (VB)** since the former lacks the `System.Design.dll`, which is required by the property pages at design-time. If you don't change the framework, the following error message will pop up when you try to drag the control onto the form:



To place the VARCHART XGantt control on the form, please select its icon  in the toolbox and draw a frame at the position in the form where you want it to appear.



If you wish the bottom and right-hand side of the VARCHART Windows Forms control to adjust to the full size of the window during runtime, the "load" and "resize" events of the form need to contain the below code:

Example Code VB.NET

```
Private Sub Form1_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    VcGantt1.Width = ClientSize.Width - VcGantt1.Left
    VcGantt1.Height = ClientSize.Height - VcGantt1.Top
End Sub

Private Sub Form1_Resize(ByVal sender As Object, ByVal e As
System.EventArgs) Handles MyBase.Resize
```

26 Placing the Control on a Form

```
VcGantt1.Width = ClientSize.Width - VcGantt1.Left  
VcGantt1.Height = ClientSize.Height - VcGantt1.Top  
End Sub
```

Example Code C#

```
private void Form1_Load(object sender, System.EventArgs e)  
{  
    vcGantt1.Width = ClientSize.Width - vcGantt1.Left;  
    vcGantt1.Height = ClientSize.Height - vcGantt1.Top;  
}  
  
Private void Form1_Resize(object sender, System.EventArgs e)  
{  
    vcGantt1.Width = ClientSize.Width - vcGantt1.Left;  
    vcGantt1.Height = ClientSize.Height - vcGantt1.Top;  
}
```

Tip:

A "name space" instruction at the beginning of the program will save you the detailed reference indication when using data types and "enum" elements.

VB: Imports NETRONIC.XGantt

C#: using NETRONIC.XGantt;

For example instead of **NETRONIC.XGantt.VcNodeCollection** you only need to write **VcNodeCollection**.

2.3 Supplying Data

For activities and links to be displayed, VARCHART XGantt needs the supply of data. By default, for the communication associated two tables are used:

1. NodeTable (also called Maindata)
2. LinkTable (also called Relations)

When placing a VARCHART XGantt in the form, the basic fields were already provided in advance.

Fields of the Maindata data table:

Index	Name	Primary key	Type	DateFormat	Editable	Hidden
0	ID	True	Integer		True	False
1	Name	False	String		False	False
2	Start	False	DateTime	DD.MM.YYYY	False	False
3	End	False	DateTime	DD.MM.YYYY	True	False
4	Duration	False	Integer		False	False

Fields of the Relations data table:

Index	Name	Primary key	Type	Editable	Hidden
0	Link ID	True	String	False	False
1	Predecessor Node ID	False	String	True	False
2	Successor Node ID	False	String	True	False

Further fields required need to be defined manually. You can do this at design time by the dialog **Administrative Data Tables** (lower section) or at run time by the method **Add(...)** of the object **VcDataTableFieldCollection**.

If you need more tables than the ones defined by default you can create them in the upper section of the dialog box **Administrative Data Tables** after having clicked **Extended data tables enabled** on the property page **General**.

Administrative Data tables

Data tables

Name	Status	Multiple primary keys allowed	Description
Maindata	<input type="checkbox"/>	<input type="checkbox"/>	
Relations	<input type="checkbox"/>	<input type="checkbox"/>	

Data table fields

Index	Name	Primary key	Type	Date format	Editable	Hidden
0	ID	<input checked="" type="checkbox"/>	Integer		<input type="checkbox"/>	<input type="checkbox"/>
1	Name	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>
2	Start	<input type="checkbox"/>	Date/Time	DD.MM.YYYY	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3	End	<input type="checkbox"/>	Date/Time	DD.MM.YYYY	<input checked="" type="checkbox"/>	<input type="checkbox"/>
4	Duration	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	Completion	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>
6	Group Level 1	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>
7	Group Level 2	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>
8	Release Date	<input type="checkbox"/>	Date/Time	DD.MM.YYYY	<input checked="" type="checkbox"/>	<input type="checkbox"/>
9	Due Date	<input type="checkbox"/>	Date/Time	DD.MM.YYYY	<input checked="" type="checkbox"/>	<input type="checkbox"/>

OK Cancel Apply Help

The method **DataRecordByID()** of **VcDataRecordCollection** permits to quickly find objects by means of the primary key.

In order to make activities and links visible in our starter sample, you need to enter some records into the data table first.

This can be done by using the method **Add(...)** of the object type **VcDataRecordCollection**. The method **EndLoading** completes the data input for the corresponding chart to be composed. For this, please enter the following code lines in the **Load** event of the form.

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection

VcGantt1.ExtendedDataTablesEnabled = True
```

```

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection
dataRecCltn.Add("1;Node 1;07.05.2010;;5")
dataRecCltn.Add("2;Node 2;14.05.2010;;5")
dataRecCltn.Add("3;Node 3;21.05.2010;;5")

dataTable = VcGantt1.DataTableCollection.DataTableByName("Relations")
dataRecCltn = dataTable.DataRecordCollection
dataRecCltn.Add("1;1;2")
dataRecCltn.Add("2;2;3")

VcGantt1.EndLoading

```

Example Code C#

```

vcGantt1.ExtendedDataTablesEnabled = true;
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;
dataRecCltn.Add("1;Node 1;07.05.2010;;5");
dataRecCltn.Add("2;Node 2;14.05.2010;;5");
dataRecCltn.Add("3;Node 3;21.05.2010;;5");

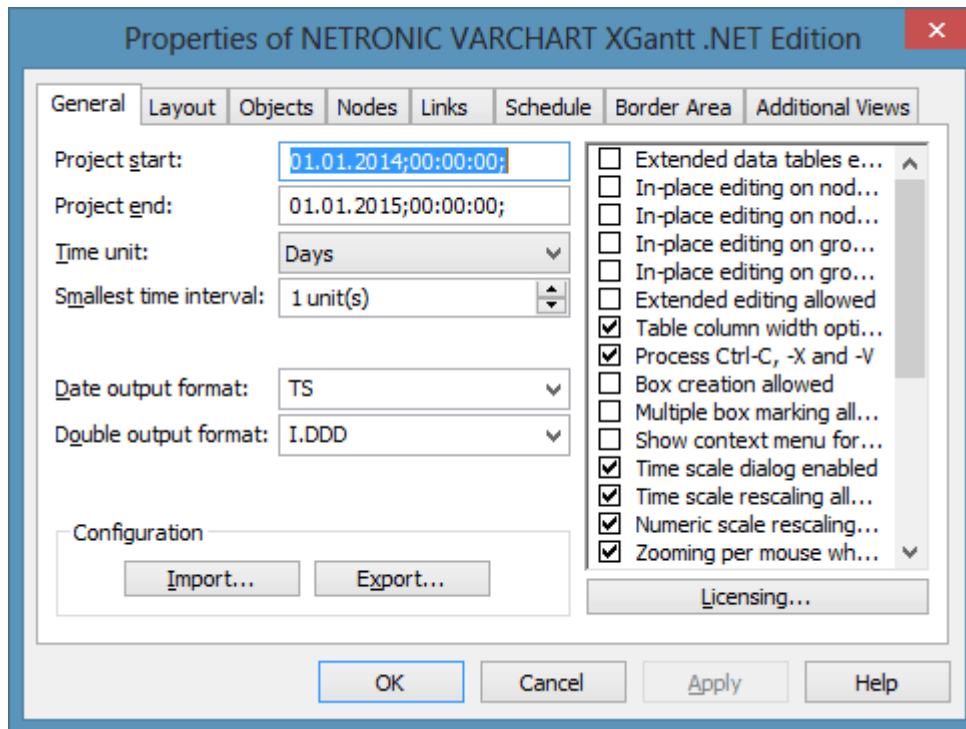
dataTable =
vcGantt1.DataTableCollection.DataTableByName("Relations");
dataRecCltn = dataTable.DataRecordCollection;
dataRecCltn.Add("1;1;2");
dataRecCltn.Add("2;2;3");

vcGantt1.EndLoading;

```

The values in a record are separated by semicolons. The order of the fields has to correspond to the order of the fields in the data definition. New records have to have an unambiguous identification which is not empty. The date in the record has to correspond to the `DateFormat` definition in the data definition table. The interpretation of the duration depends on the settings of **Time unit**. It is pre-set to **days**, which you can modify on the **General** property page.

The **Date output format** is defined consistently for the table and for each dialog on the **General** property page.



Loading data from a CSV file

Alternatively, you may also load the data from a CSV file. The structure of the file has to correspond to the below scheme:

Example Code

```
1;Node 1;07.05.2010;;5;
2;Node 2;14.05.2010;;5;
3;Node 3;21.05.2010;;5;
****
1;1;2;
2;2;3;
```

Each record has its own line. The contents of the lines correspond to the parameters passed by the method **Add(...)** of the object type **VcDataRecordCollection**.

The records of the Maindata data table are listed first, followed by the records of the Relations data table. Use ****** Table name ****** to mark the beginning of a record group.

If you saved this kind of file for example by the name **intro.csv**, you may import the data as follows:

Example Code VB.NET

```
VcGantt1.Open("c:\intro.csv")
```

Example Code C#

```
vcGantt1.Open(@"c:\intro.csv");
```

Specifying the period of time which is represented

Until now, you will see no activities, because the time scale has not been adjusted to the corresponding period. The displayed range of the time scale can be defined via the properties **TimeScaleStart** and **TimeScaleEnd** or determined from the data by the method **OptimizeTimeScaleStartEnd(...)** of the object **VcGantt**.

Example Code VB.NET

```
VcGantt1.TimeScaleEnd = New DateTime(2011, 1, 1)
VcGantt1.TimeScaleStart = New DateTime(2010, 5, 4)
```

Example Code C#

```
vcGantt1.TimeScaleEnd = new DateTime(2011,1,1);
vcGantt1.TimeScaleStart =new DateTime(2010,5,4);
```

Below you can find the code which you will need for our starter sample.

Example Code VB.NET

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    VcGantt1.Width = ClientSize.Width - VcGantt1.Left
    VcGantt1.Height = ClientSize.Height - VcGantt1.Top

    Dim dataTable As VcDataTable
    Dim dataRecCltn As VcDataRecordCollection

    vcGantt1.ExtendedDataTablesEnabled = True
    dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
    dataRecCltn = dataTable.DataRecordCollection
    dataRecCltn.Add("1;Node 1;03.05.2010;;5")
    dataRecCltn.Add("2;Node 2;08.05.2010;;5")
    dataRecCltn.Add("3;Node 3;15.05.2010;;5")

    dataTable = VcGantt1.DataTableCollection.DataTableByName("Relations")
    dataRecCltn = dataTable.DataRecordCollection
    dataRecCltn.Add("1;1;2")
    dataRecCltn.Add("2;2;3")

    VcGantt1.EndLoading()

    VcGantt1.OptimizeTimeScaleStartEnd(3)
End Sub

Private Sub Form1_Resize(ByVal sender As Object, ByVal e As
System.EventArgs) Handles MyBase.Resize
```

32 Supplying Data

```
VcGantt1.Width = ClientSize.Width - VcGantt1.Left
VcGantt1.Height = ClientSize.Height - VcGantt1.Top
End Sub
```

Example Code C#

```
private void Form1_Load(object sender, System.EventArgs e)
{
    vcGantt1.Width = ClientSize.Width - vcGantt1.Left;
    vcGantt1.Height = ClientSize.Height - vcGantt1.Top;

    vcGantt1.ExtendedDataTablesEnabled = true;
    VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
    VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;
    dataRecCltn.Add("1;Node 1;03.05.2010;;5");
    dataRecCltn.Add("2;Node 2;08.05.2010;;5");
    dataRecCltn.Add("3;Node 3;15.05.2010;;5");

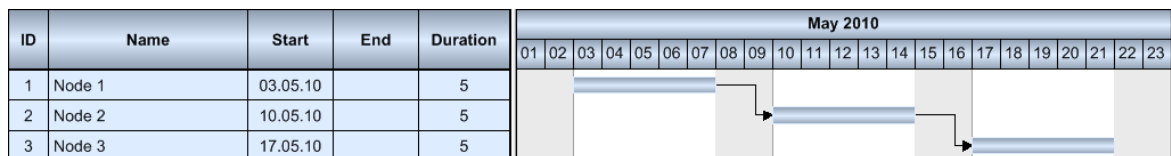
    dataTable =
vcGantt1.DataTableCollection.DataTableByName("Relations");
    dataRecCltn = dataTable.DataRecordCollection;
    dataRecCltn.Add("1;1;2");
    dataRecCltn.Add("2;2;3");

    vcGantt1.EndLoading();

    vcGantt1.OptimizeTimeScaleStartEnd(3);
}

private void Form1_Resize(object sender, System.EventArgs e)
{
    vcGantt1.Width = ClientSize.Width - vcGantt1.Left;
    vcGantt1.Height = ClientSize.Height - vcGantt1.Top;
}
```

If you run the program now, the result should be as shown in the below illustration.



2.4 Calculating End Dates

The table column that holds the end dates is still empty. The end of an activity can be calculated from the fields **Start** and **Duration** by using the calendar which is included in VARCHART XGantt.

In the default calendar, the weekdays (Monday to Friday) are pre-defined as active times and the weekends (Saturday and Sunday) are defined as non-active times.

You can recognize the non-active times in the diagram by their gray background. The calendar may be switched off by deactivating the option **Nodes use calendar** on the **Nodes** property page.

Please note the difference in calculating when using and when not using a calendar:

An activity which starts on Friday and lasts for 3 days will end on Tuesday if the calendar is activated. Without a calendar, the activity will finish on Sunday already.

The end date can be calculated by using the method **AddDuration(...)** of the object **VcCalendar**. This requires the **start** and the **duration** of each activity. The fields can be accessed via their index. After having set the end date by the method **set_DataField(...)**, the method **Update()** of **VcNode** needs to be invoked for the modifications to be displayed.

Example Code VB.NET

```
Dim tmpCal As VcCalendar
Dim tmpDate As Date
Set tmpCal = VcGantt1.CalendarCollection.Active
tmpDate = tmpCalendar.AddDuration(node.DataField(2), node.DataField(4))
node.DataField(3) = tmpDate
node.Update()
```

Example Code C#

```
VcCalendar tmpCal = vcGantt1.CalendarCollection.Active;
DateTime tmpDate = tmpCal.AddDuration((DateTime)node.get_DataField(2),
                                     Convert.ToInt32(node.get_DataField(4)));
node.set_DataField(3, tmpDate);
node.Update();
```

Start and end dates of activities that were created or modified by mouse interactions are automatically placed in active times.

34 Calculating End Dates

ID	Name	Start	End	Duration									
					01	02	03	04	05	06	07	08	09
1	Node 1	03.05.10	08.05.10	5									

In contrast, dates that were set by the API or by editing dialogs can be placed in non-working times.

ID	Name	Start	End	Duration									
					01	02	03	04	05	06	07	08	09
1	Node 1	03.05.10	08.05.10	5									

Dates that were generated by calculation are always placed in working times. In order to ensure dates set by the API to be placed in working times, the start date needs to be calculated from the end date and from the duration of the activity.

Example Code VB.NET

```
tmpDate = tmpCal.AddDuration(node.DataField(3),  
                             (-1) * node.DataField(4))  
node.DataField(2) = tmpDate
```

Example Code C#

```
tmpDate = tmpCal.AddDuration((DateTime)node.get_DataField(3), (-1) *  
Convert.ToInt32(node.get_DataField(4)));  
node.set_DataField(2, tmpDate);
```

For keeping the data consistent, missing or negative durations should be treated as improper and be reset to 0. If the start date is missing, the end date cannot be calculated. The code was resumed in a separate method called **SetNodeEndDate(...)**.

Example Code VB.NET

```
Private Sub SetNodeEndDate(ByVal node As VcNode)  
    'Avoid empty duration or negative duration  
    If node.DataField(4) = "" Or node.DataField(4) < 0 Then  
        node.DataField(4) = "0"  
    End If  
    'Start date empty then end date should also be empty  
    If node.DataField(2) = "31.12.1899 00:00:00" Then  
        node.DataField(3) = ""  
    Else  
        'Precondition is property page nodes  
        '"Assign calendar to nodes" must be true  
        Dim tmpCal As VcCalendar  
        tmpCal = VcGantt1.CalendarCollection.Active  
        Dim tmpDate As DateTime  
        tmpDate = tmpCal.AddDuration(node.DataField(2), node.DataField(4))  
        node.DataField(3) = tmpDate  
        'Start date only in active times
```

```

        tmpDate = tmpCal.AddDuration(node.DataField(3),
                                     (-1) * node.DataField(4))
        node.DataField(2) = tmpDate
        node.Update()
    End If
End Sub

```

Example Code C#

```

private void SetNodeEndDate(VcNode node)
{
    // Avoid empty duration or negative duration
    if ((string) node.get_DataField(4) == "" ||
        Convert.ToInt32(node.get_DataField(4)) < 0)
        node.set_DataField(4, "0");

    // Start Date empty then end date should also be empty
    if (node.get_DataField(2).ToString() == "31.12.1899 00:00:00")
        node.set_DataField(3, "");
    else
    {
        // Precondition in property page nodes
        // "Assign calendar to nodes" must be true
        VcCalendar tmpCal = vcGantt1.CalendarCollection.Active;

        DateTime tmpDate = tmpCal.AddDuration(
            (DateTime)node.get_DataField(2),
            Convert.ToInt32(node.get_DataField(4)));
        node.set_DataField(3, tmpDate);

        // start date only in active times
        tmpDate = tmpCal.AddDuration((DateTime)node.get_DataField(3),
            (-1) * Convert.ToInt32(node.get_DataField(4)));
        node.set_DataField(2, tmpDate);
        node.Update();
    }
}

```

The calculation of dates is required:

1. After activities were loaded
2. After dates or durations were modified by a data editing dialog or by an in-place editor
3. After activity values were modified by the API

After modifications by mouse interaction however, a calculation does not need to be initiated since in this case, an internal calculation will be carried out automatically.

A computation loop which includes all nodes can be set up by the property **NodeCollection** of the **VcGantt** object. Its code will be added to the end of the event **Form1_Load(...)**.

36 Calculating End Dates

Example Code VB.NET

```
'Calculate end date for all nodes
Dim node As VcNode
For Each node In VcGantt1.NodeCollection
    SetNodeEndDate node
Next
```

Example Code C#

```
// Calculate end date for all nodes
foreach (VcNode node in vcGantt1.NodeCollection)
{
    SetNodeEndDate (node);
}
```

Alterations of data caused by the user can be picked up by the event **VcNodeModified**. The method call added will calculate the end date.

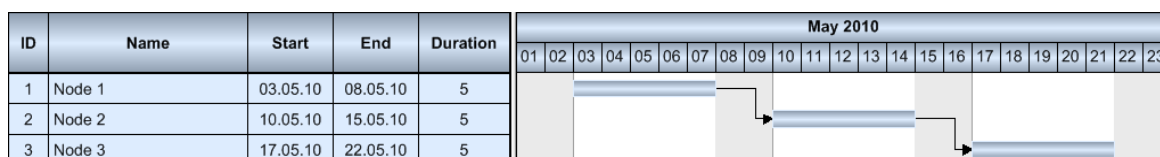
Example Code VB.NET

```
Private Sub VcGantt1_VcNodeModified(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeModifiedEventArgs) Handles VcGantt1.VcNodeModified
    SetNodeEndDate (e.Node)
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeModified(object sender,
NETRONIC.XGantt.VcNodeModifiedEventArgs e)
{
    SetNodeEndDate (e.Node);
}
```

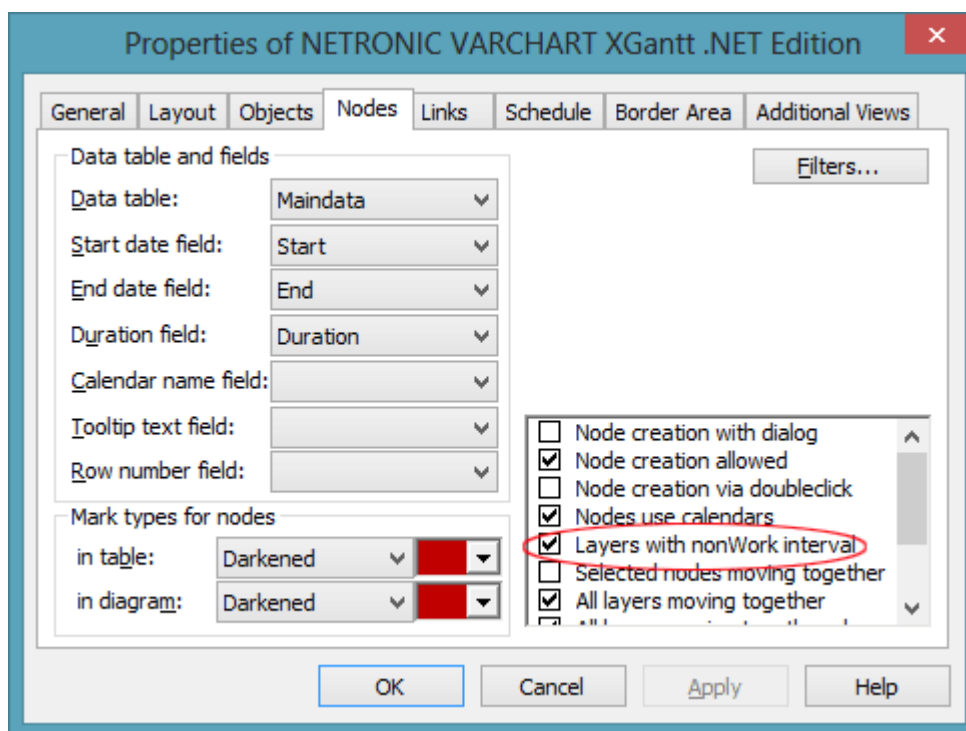
If values of data were altered by the API, the method **SetNodeEndDate(...)** has to be invoked explicitly.



2.5 Marking Non-working Intervals in Activities

The visual interruption of the activities by non working intervals can be displayed by setting the option **Layers with nonWork interval**. The option only shows if the activities depend on a calendar. To link nodes to a calendar, you can set the option **Nodes use calendars**.

The option can be activated during run time or during design time. At design time, on the property page **Nodes** you can activate the option **Layers with nonWork interval**.

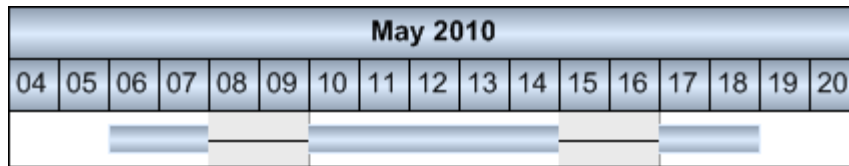


At runtime you can set the property **LayersWithNonWorkInterval** of the object VcGantt.

May 2010																
04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20

LayersWithNonWorkInterval = false

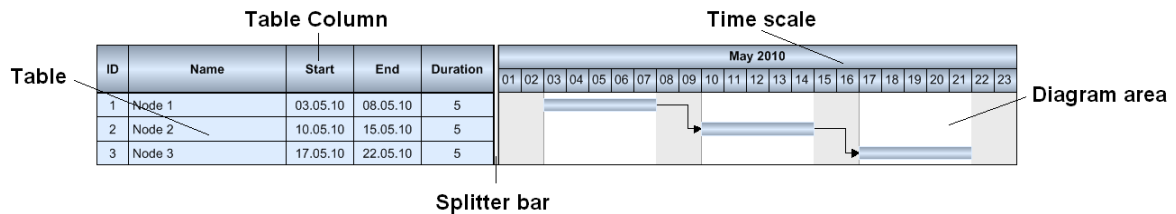
38 Marking Non-working Intervals in Activities



LayersWithNonWorkInterval = true

2.6 Interactions in the Table and Diagram Area

This subchapter and the one following will give you a general idea of interactions in the Gantt diagram. For more detailed information please see chapter **User Interface**.



> Modifying the left table/diagram width ratio

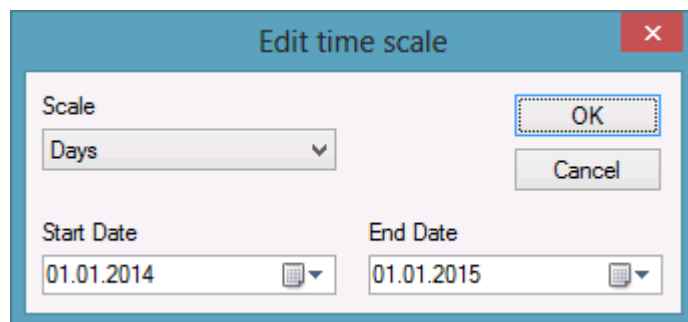
You can modify the sizes of the table and the diagram section of a Gantt chart by moving the vertical splitter bar between them. The ratio between the sections is pre-set (to show a value on the start) on the **Layout** property page in the field **Left table/diagram width ratio**.

> Modifying the table column

By dragging the vertical separation line on the right of a table caption you can modify the width of a table column. You can automatically adjust the column width to the length of its contents by double clicking on the separation line. The automatical adjustment can be switched on or off on the **General** property page in the field **Table column width optimization allowed**.

> Defining the start and end date of the time scale

By a double-click on the time scale you can pop up the **Edit Timescale** dialog box. It lets you edit the start and end dates of the time scale. This option may be activated or blocked on the **General** property page in the field **Time scale dialog enabled**.



> **Scaling the Time Scale**

By dragging to the left or to the right in the time scale section you may enlarge or reduce the width of the unit of the time scale. This feature can be activated or or deactivated on the **General** property page at **Time scale rescaling allowed**.

2.7 Interactions with Activities

> Creating a new activity

To create an activity, please change to the mode **Create Node** by the context menu of the Gantt graph (right mouse button on an empty area).

The mouse pointer will adopt the shape of a small cross. While keeping the left mouse button pressed, please draw an activity in the desired area of the Gantt graph. When finished, please return to pointer mode by the context menu. An application program is able to interact with the "create" mode by the event **VcNodeCreated()**. This is useful for example, if you wish to pre-set data values of the activity being created.

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeCreated(ByVal sender As Object, ByVal e As
VcNodeCreatedEventArgs) Handles VcGantt1.VcNodeCreated
    e.Node.DataField(1) = "Node " + e.Node.DataField(0)
    e.Node.Update()
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeCreated(object sender,
VcNodeCreatedEventArgs e)
{
    e.Node.set_DataField(1, "Node " + e.Node.get_DataField(0));
    e.Node.Update();
}
```

The code above will modify the value of the data field "Name". It will attach the current value of the field "ID" to the term "Node".

> Modifying the duration of an activity

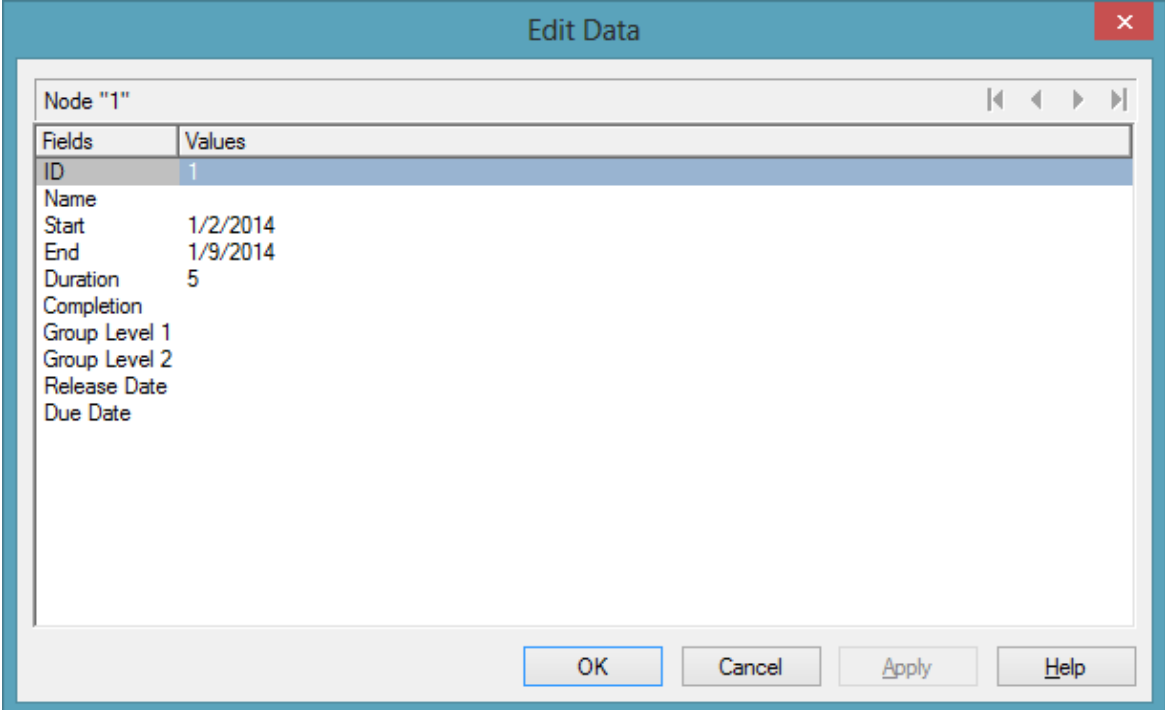
Please return to the "Select" mode and move the mouse pointer to the internal right or left margin of the activity. The mouse pointer will adopt the shape of a vertical line with an arrow. By dragging to the left or to the right, you can extend or shorten the activity.

> Moving an activity

Please change to "Select" mode and place the mouse pointer on an activity. The pointer transforms into a small square with four arrows. Now you can move the activity to any position in the Gantt graph. If you wish to move more than one activity simultaneously please activate the option **Selected nodes moving together** on the property page **Nodes**.

> Editing the data of an activity

By double clicking on the activity or on the corresponding table entry, you can open the dialog box **Edit data**.



Fields	Values
ID	1
Name	
Start	1/2/2014
End	1/9/2014
Duration	5
Completion	
Group Level 1	
Group Level 2	
Release Date	
Due Date	

> Deleting an activity

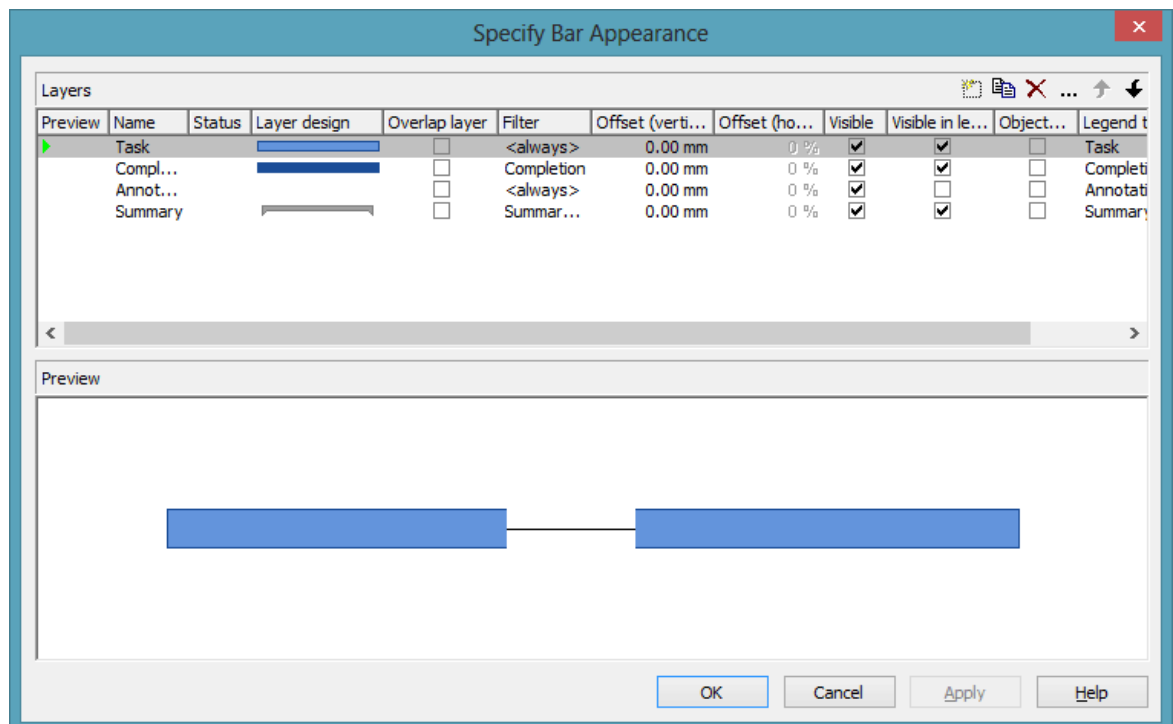
Press the **Del** button or click **Delete nodes** in the context menu of the activity to delete marked activities. You can mark an activity by clicking on it or on the corresponding line in the table. If you keep the Ctrl key pressed, you can mark more than one activity.

2.8 Using Layers

A layer is the graphical representation of a pair of dates. In addition, the same pair of dates can be displayed by several layers. Logically, the different layers stack up to a pile.

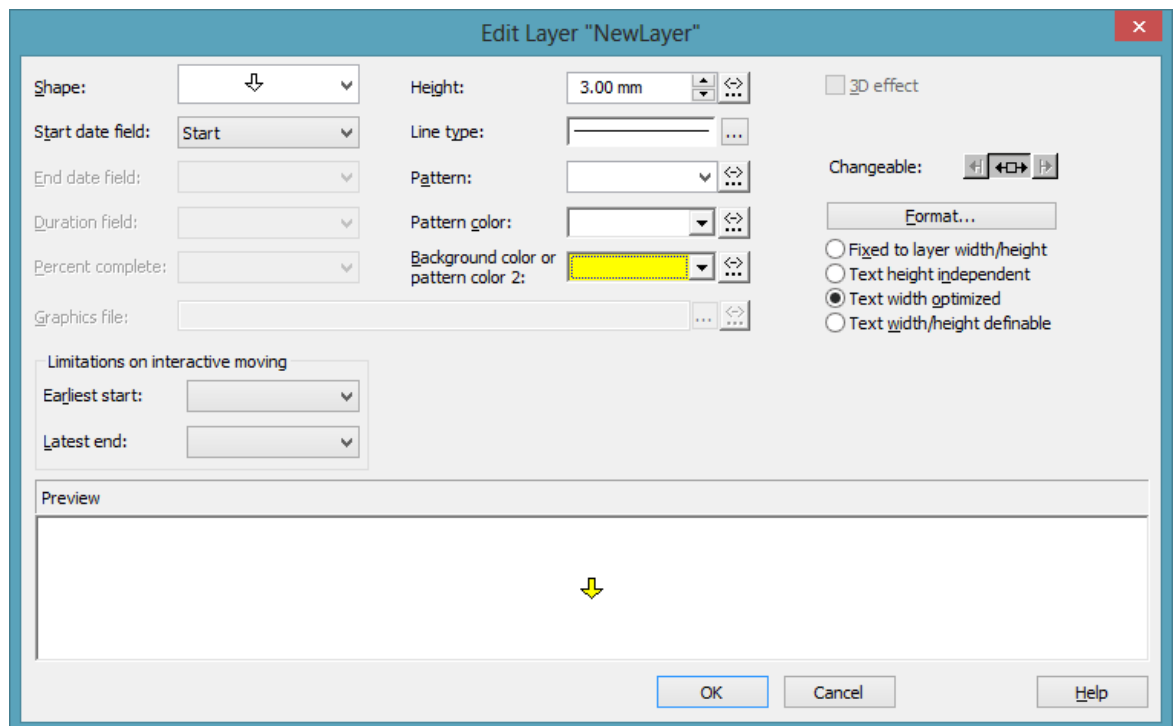
In our example, we are now going to create a second, different looking layer.

1. On the **Objects** property page, please select **Layers...**. The dialog **Specify Bar Appearance** will pop up. Please note that the layer **Start-End** was already predefined.

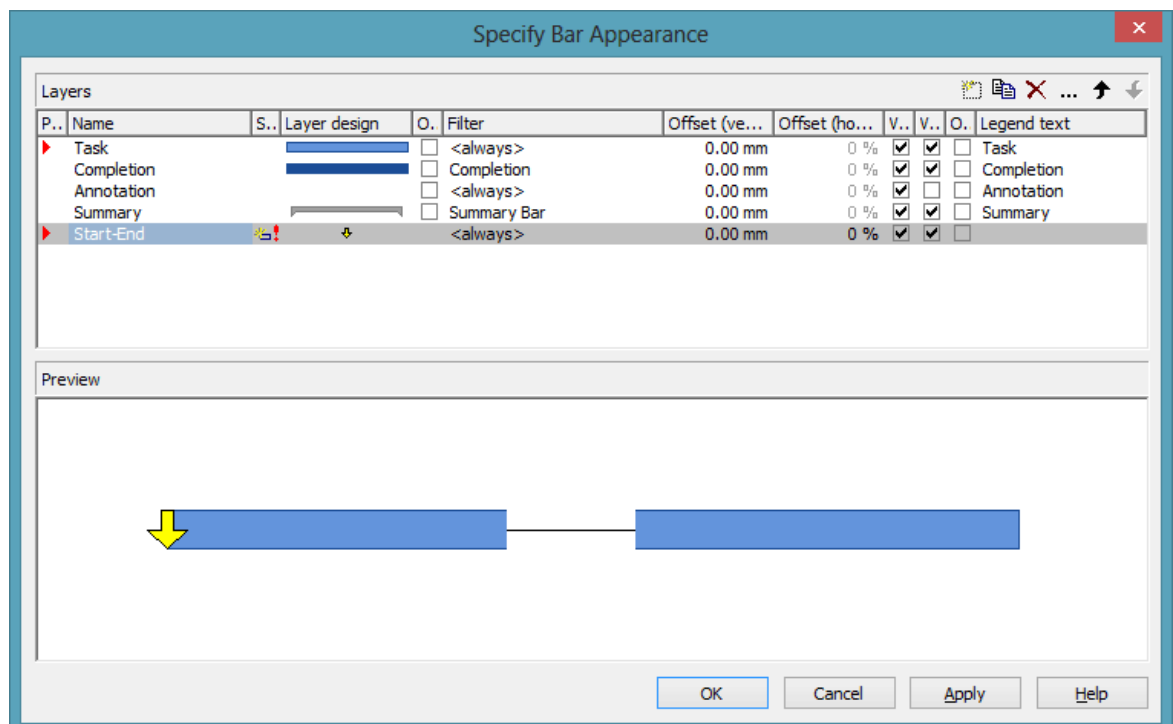


2. Copy the definition of the layer **Task** by clicking on the **Copy layer** button
3. Change the name of **NewLayer** to **Start-End** and open the **Edit Layer** dialog by clicking on
4. Please change the **Shape** to arrowhead downward and the **Background color** to yellow.

44 Using Layers

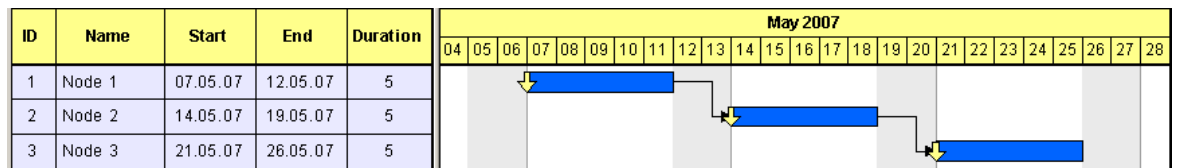


5. By clicking on OK, you will return to the dialog **Specify Bar Appearance**.
6. Each layer of a node will be displayed in the preview below if you click in the column **Preview** of the corresponding fields. A red triangle instead of a green one indicates the display of the layer in the preview window below.



7. In our programming sample, the modification of the definition shows the

8. below result:




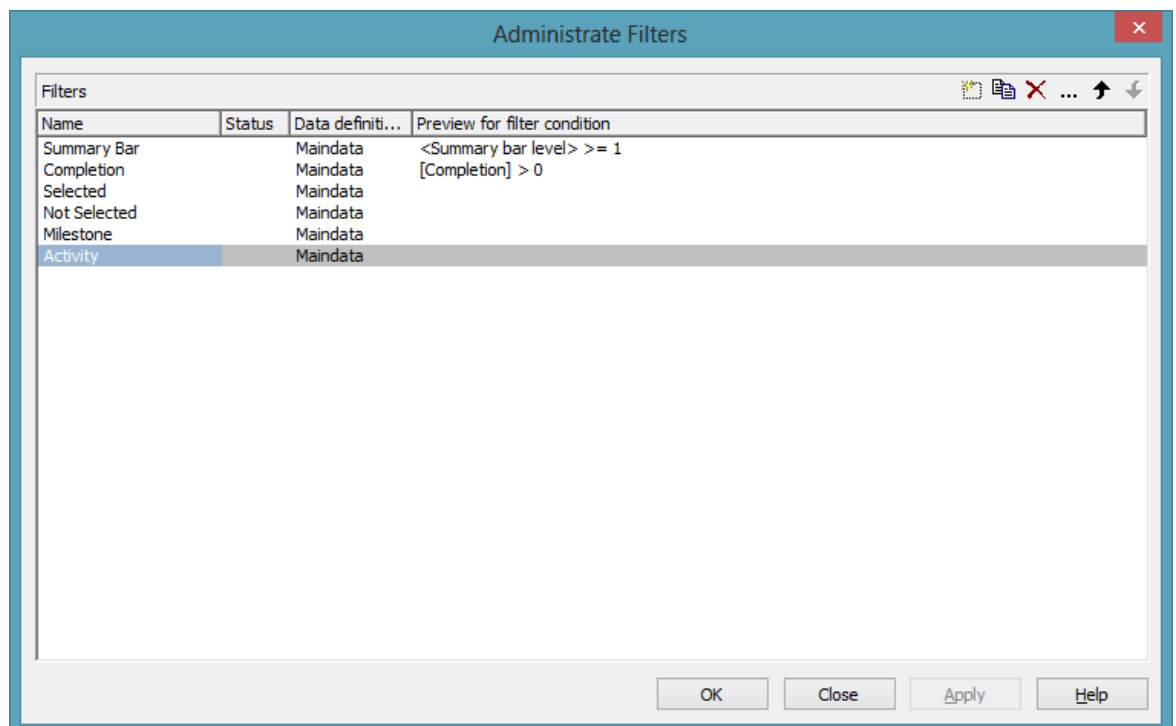
2.9 Using Filters


Next, we would like to have the red arrow appear only if the node is a milestone i.e., if the duration of the activity equals 0.

This problem can be solved easily by using filters. A filter consists of a series of linked conditions which result in a logical Yes/No statement.

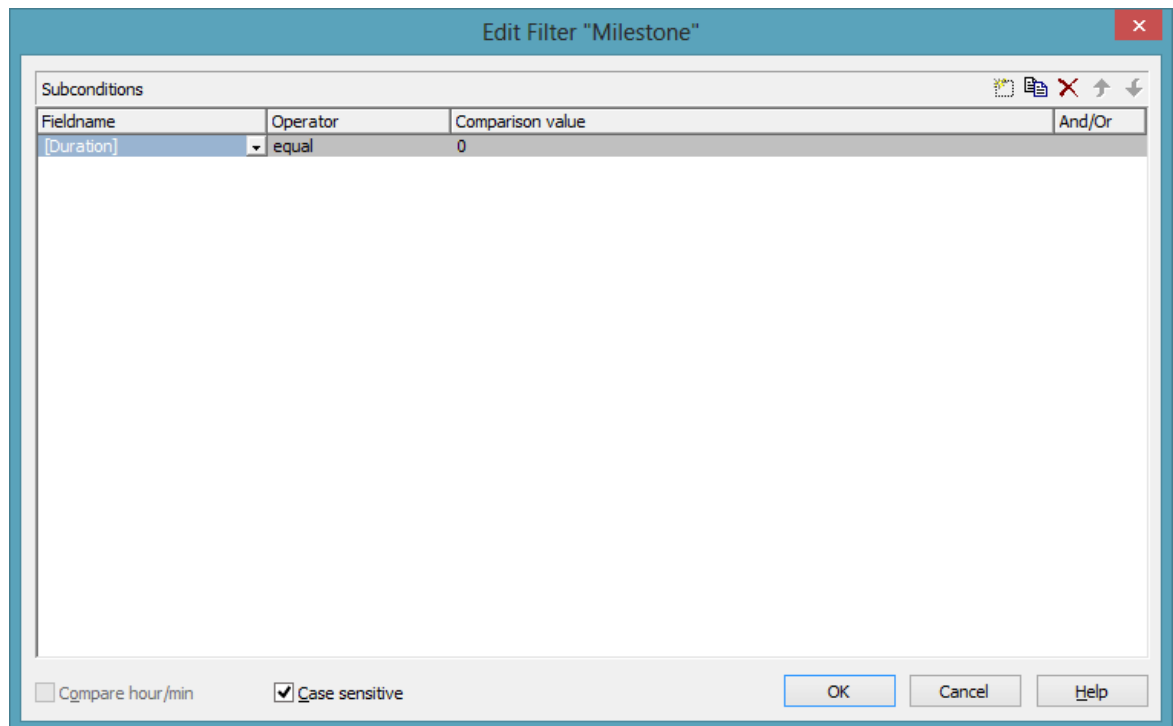
Layers are always linked to filters. The corresponding layer becomes visible only if the evaluation of the filter conditions results in "Yes". The filter `<always>`, which is assigned to a layer by default, always returns "Yes". For our example, two filters are required that contain one condition each:

- The red arrow shall appear if the duration = 0
 - The blue bar shall appear if the duration > 0
1. On the property pages **Objects** please click on the button **Filters**, which will pop up the dialog **Administrative Filters**.
 2. Now please create two new filters by clicking on the button .
 3. In the column **Name** rename "NewFilter" and "NewFilter1" into "Milestone" and "Activity".
 4. Please confirm the modifications by clicking on **Apply**.

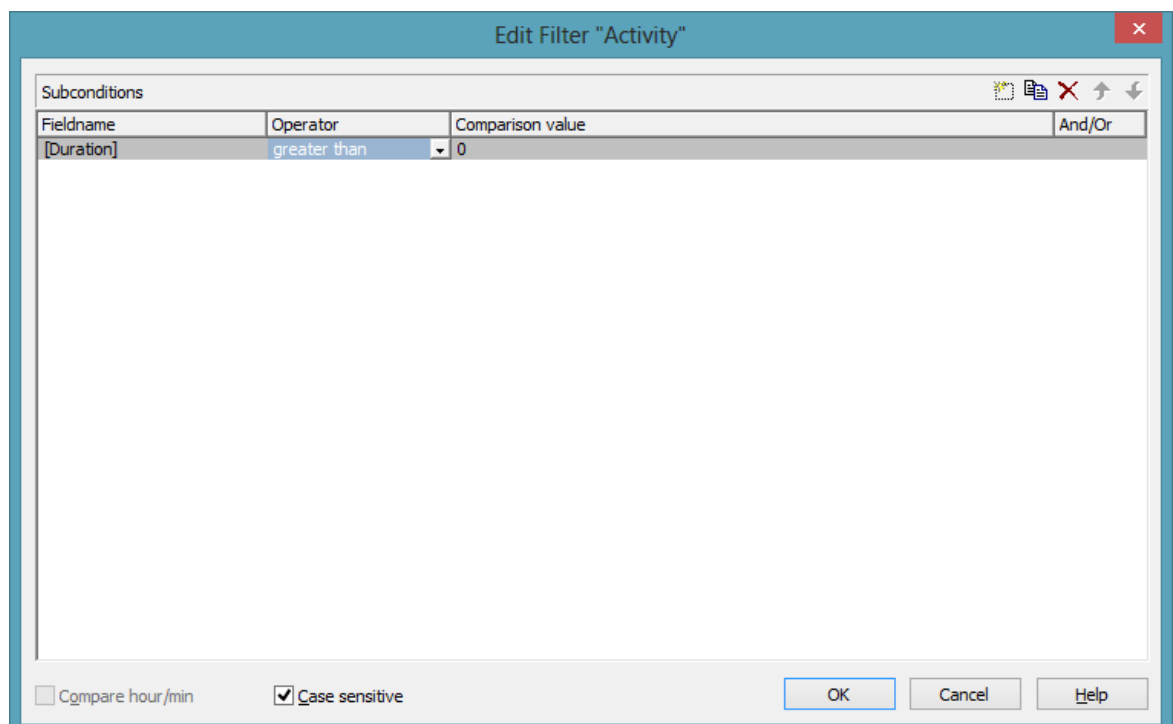


5. Select the filter "Milestone" and open the dialog **Edit Filter** by clicking on .

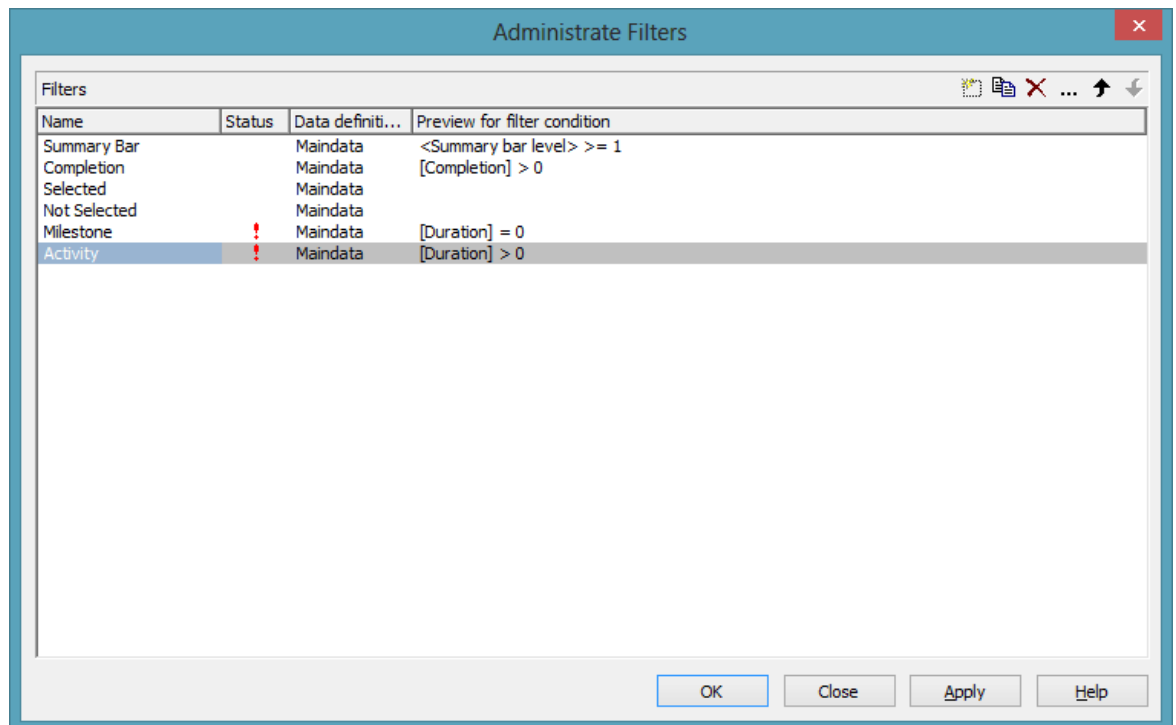
6. Select "Duration" as **Fieldname**, as **Operator** "equal" and as **Comparison value** 0.



7. Leave the dialog by clicking on **OK**.
8. Select "Activity" and by clicking **...** go again to the **Edit Filter** dialog.
9. Select "Duration" as **Fieldname**, for the **Operator** "greater than" and for the **Comparison value** 0.

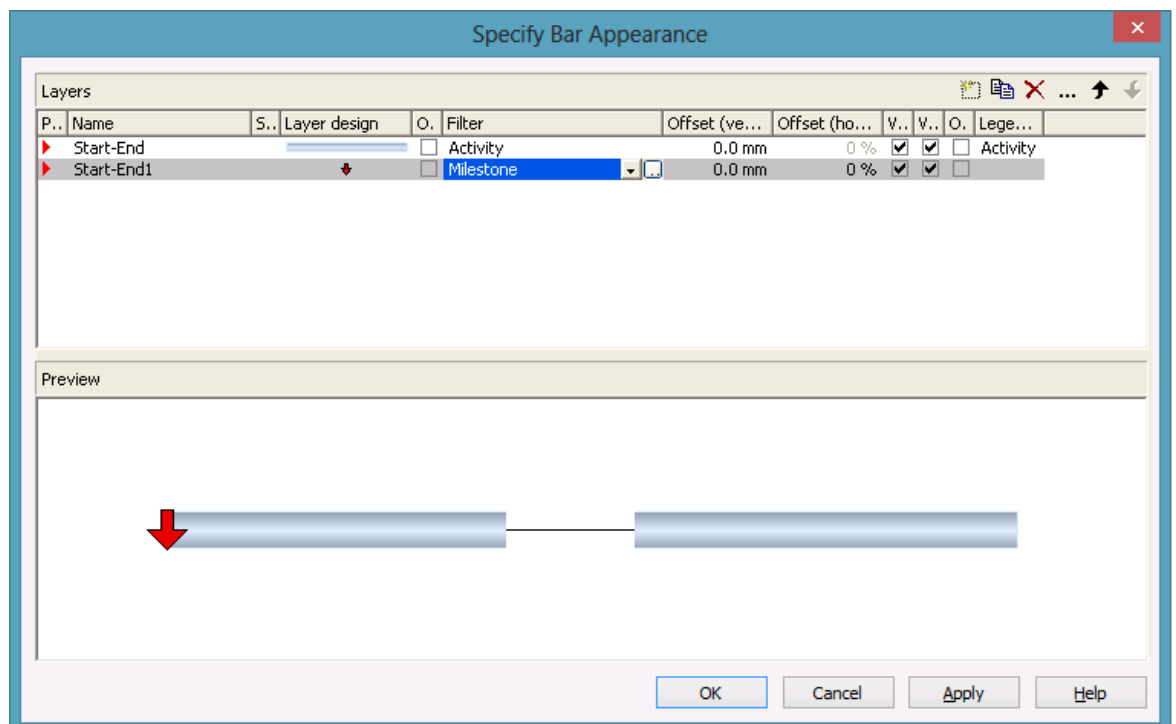


10. Leave the dialog by clicking on **OK**.

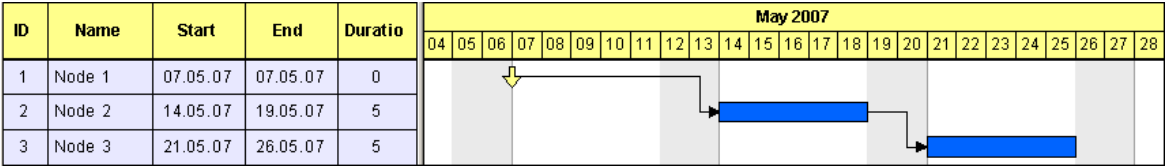


11. Click on **OK** again to return to the property pages.

12. To put the filters into operation they need to be assigned to the layers. For this, please click on the button **Layers...** to open the dialog **Specify Bar Appearance**.



13. If you run the program now and if the duration of the first activity is set to 0, the below result will be produced:



2.10 Creating Histograms

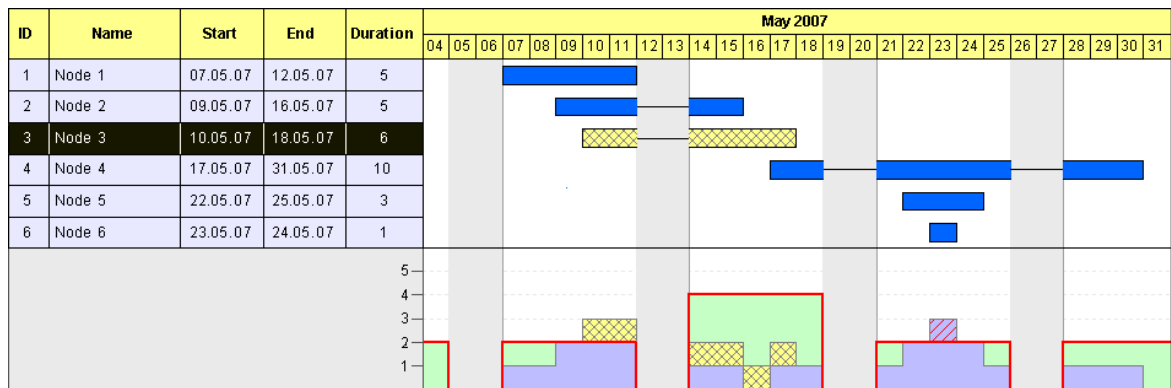
In this chapter we will demonstrate how to program a histogram.

Histograms sum up activities and reproduce the totalized result as a graphical plain or area. The line that limits the area at the top represents the workload curve (for example of a machine, of a production line or of the staff of a project – denoted from now on as a "production system").

The work load curve can refer to the capacity curve, the latter representing a set of the maximum possible workload values. While the workload curve composes of values taken from the activities, the values of the capacity curve have to be taken from data otherwise defined.

When displaying the two curves, the workload curve may exceed or fall below the capacity curve, indicating a bottleneck or a shortfall in the production system, respectively. Areas where the capacity curve is above the workload curve are shortfall areas. The workload curve being above the capacity curve indicates an overload. If both curves are at equal height, the workload has reached its optimum.

Overload and shortfall areas can differ in their appearance to visually distinguish between bottlenecks and shortfall in the production system.



The picture above shows the histogram summarizing the activities into blue unhatched areas. Marked nodes in the Gantt graph as well as in the histogram appear yellow and show a crosshatch pattern (node 3). The capacity curve is a strong red line. Overloads are displayed in blue with a crosshatch pattern (node 6). Shortfalls appear as unhatched, light green areas.

In this chapter, we will program the above histogram. You will find the complete program in the directories **UserGuideSamples \VB.NET \XGantt_Tutorial02**

and

UserGuideSamples \Csharp \XGantt_Tutorial02.

Areas that show colors and patterns in the histograms of VARCHART XGantt basically are formed by a curve, to which a reference curve is assigned. In the resulting area, colors and patterns can be filled.

To reach our aim, we will go through the below steps:

Step 1: Displaying a histogram in the Gantt chart is switched on.

Step 2: Marked activities shall appear inverted in the table while in the Gantt graph, they shall be crosshatched. As a first partial step, their markeability is switched off for the Gantt graph.

Step 3: To distinguish between selected and non-selected nodes, a data field named "Selected" is created, that stores the actual selection state of a node.

Step 4: A value is assigned to the data field, that represents the marking state.

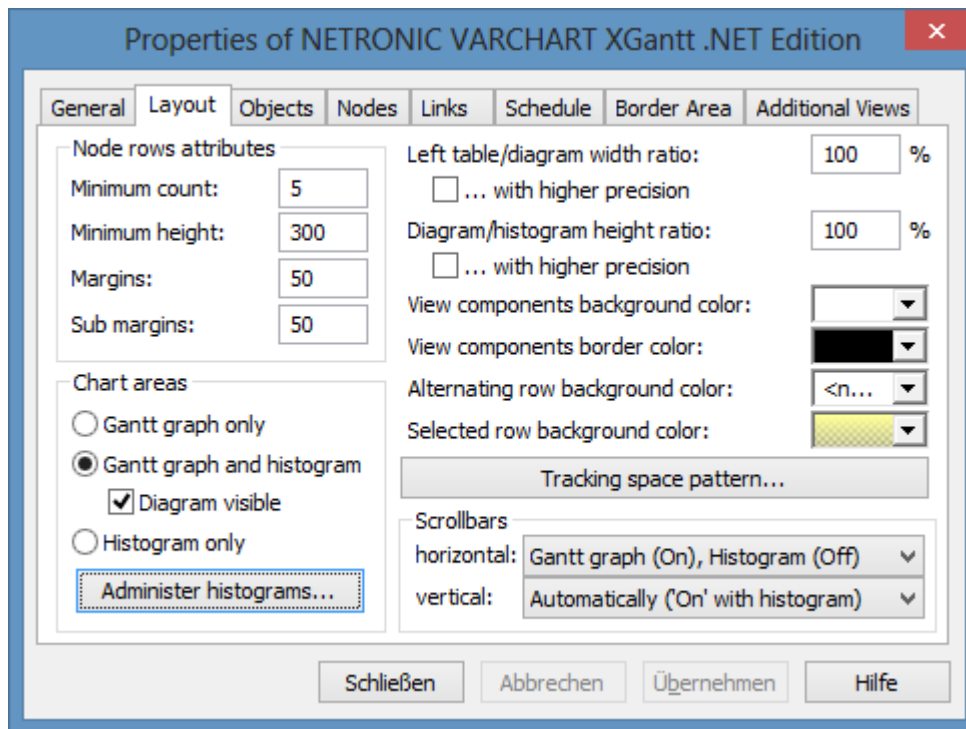
Step 5: Two different filters are created that separate selected and unselected activities.

Step 6: Two different appearances are defined for selected and unselected nodes. They are combined with the filters.

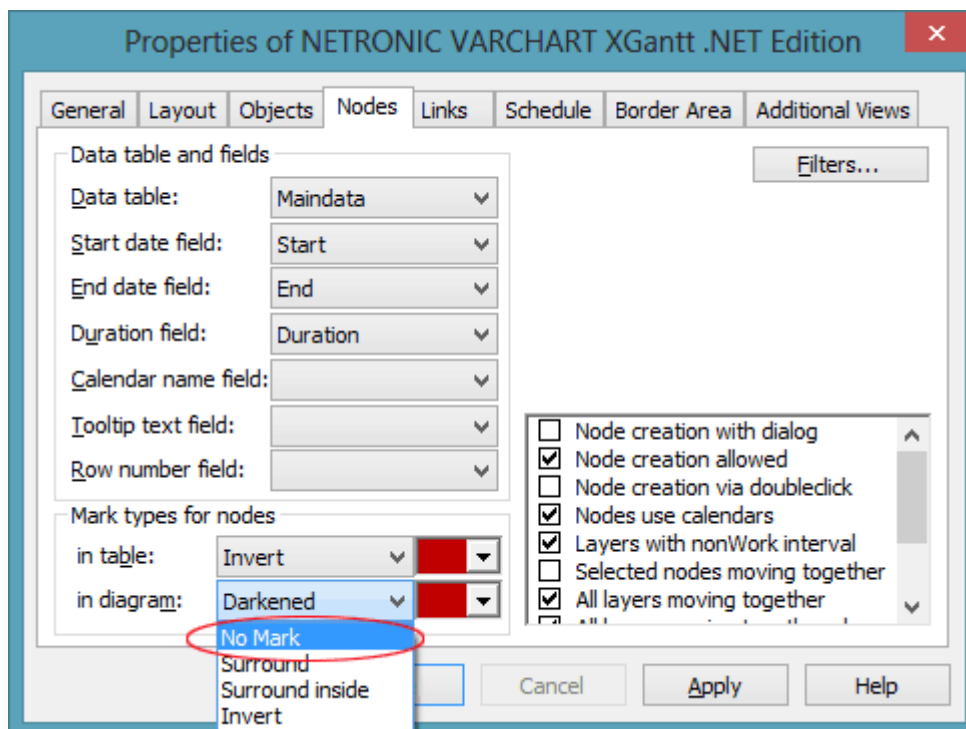
Step 7: Four curves are created for the histogram: the capacity curve, the curve of unmarked activities, the curve of marked activities and an auxiliary curve to fill an area. To the areas between the curves, colors and patterns are assigned.

Step 8: Finally, the values of the capacity curve are defined.

Step 1: First, please switch on the display of histograms in the Gantt diagram. Please invoke the property page **Layout** and find the tab section **Chart areas**, where you can set the option **Gantt graph and histogram**.



Step 2: Since marked nodes shall show a crosshatch pattern of their own, the markeability of nodes in the Gantt graph is switched off now. Please invoke the property page **Nodes**, find the tab section **Mark Types for nodes** and set the field **in diagram** to **No Mark**.



Step 3: To differentiate between selected and unselected nodes, a data field named "Selected" is created, that stores the actual selection state of a node. Please invoke the dialog **Administrative Data Tables** by clicking **Data tables** on the property page **Objects**, and edit the table **Maindata**. Here please add a field of the type **Integer** and name it "Selected". The field will make the display of the activity depend on its marking state.

The screenshot shows the 'Administrative Data Tables' dialog box. It has two main sections: 'Data tables' and 'Data table fields'.

Data tables section:

Name	Status	Multiple primary keys allowed	Description
Maindata	!	<input type="checkbox"/>	
Relations		<input type="checkbox"/>	

Data table fields section:

Index	Name	Primary key	Type	Date format	Editable	Hidden
0	ID	<input checked="" type="checkbox"/>	Integer		<input type="checkbox"/>	<input type="checkbox"/>
1	Name	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>
2	Start	<input type="checkbox"/>	Date/Time	DD.MM.YYYY	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3	End	<input type="checkbox"/>	Date/Time	DD.MM.YYYY	<input checked="" type="checkbox"/>	<input type="checkbox"/>
4	Duration	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	Completion	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>
6	Group Level 1	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>
7	Group Level 2	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>
8	Release Date	<input type="checkbox"/>	Date/Time	DD.MM.YYYY	<input checked="" type="checkbox"/>	<input type="checkbox"/>
9	Due Date	<input type="checkbox"/>	Date/Time	DD.MM.YYYY	<input checked="" type="checkbox"/>	<input type="checkbox"/>
10	Selected	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>

At the bottom of the dialog are buttons for 'OK', 'Cancel', 'Apply', and 'Help'.

Step 4: The data field "Selected" will be updated each time the event **VcNodesMarked** is triggered.

Example Code VB.NET

```
Private Sub VcGantt1_VcNodesMarked(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodesMarkedEventArgs) Handles VcGantt1.VcNodesMarked
    Dim node As VcNode
```

54 Creating Histograms

```
For Each node In VcGantt1.NodeCollection
    If node.Marked = True Then
        node.DataField(5) = 1
    Else
        node.DataField(5) = 0
    End If
    node.Update()
Next
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodesMarked(object sender,
NETRONIC.XGantt.VcNodesMarkedEventArgs e)
{
    foreach (VcNode node in vcGantt1.NodeCollection)
    {
        if (node.Marked == true)
            node.set_DataField(5,1);
        else
            node.set_DataField(5,0);
        node.Update();
    }
}
```

In the event **VcNodeCreated** the below code prevents a node from appearing marked when created. Because all previously selected nodes will be unmarked when a new node is created, the field contents of "Selected" needs to be updated.

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeCreated(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeCreatedEventArgs) Handles VcGantt1.VcNodeCreated
    e.Node.DataField(1) = "Node " + e.Node.DataField(0)
    e.Node.Marked = False
    e.Node.Update()


    Dim node As VcNode
    For Each node In VcGantt1.NodeCollection
        node.DataField(5) = 0
        node.Update()
    Next
End Sub
```

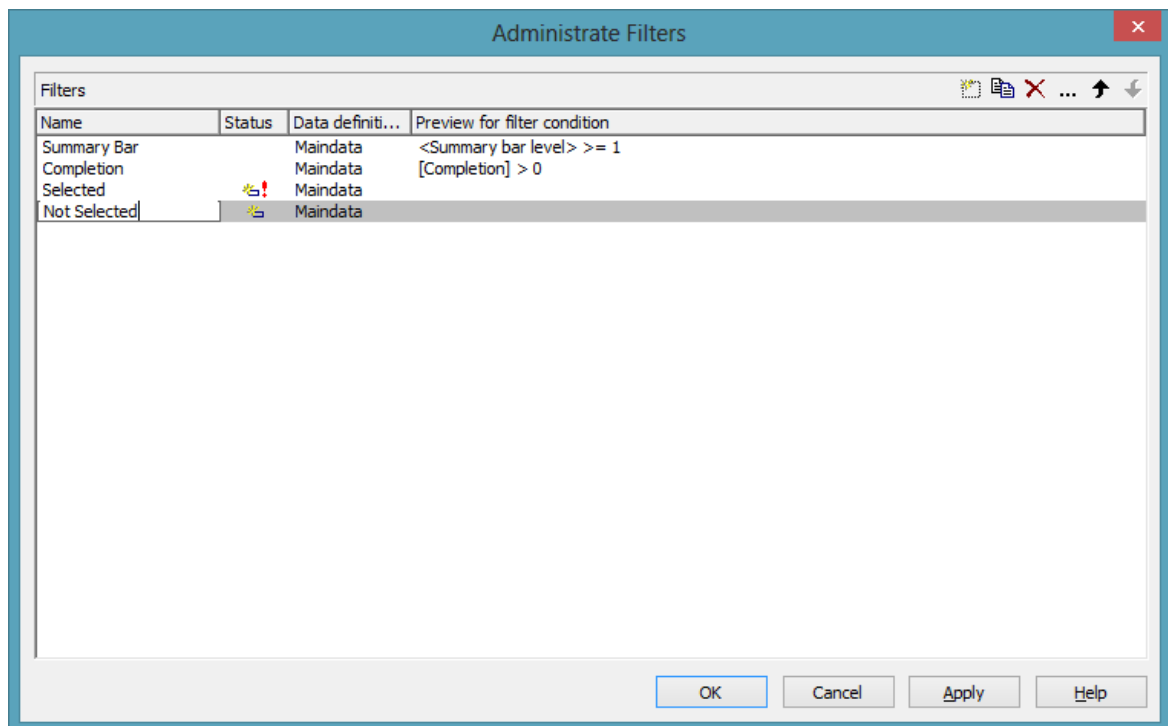
Example Code C#


```
private void vcGantt1_VcNodeCreated(object sender,
NETRONIC.XGantt.VcNodeCreatedEventArgs e)
{
    e.Node.set_DataField(1, "Node " + e.Node.get_DataField(0));
    e.Node.Marked = false;
    e.Node.Update();

    foreach (VcNode node in vcGantt1.NodeCollection)
    {
        node.set_DataField(5,0);
        node.Update();
    }
}
```

```
}
}
```

Step 5: In this step two different filters are created that separate selected from unselected activities. Please invoke the property page **Objects** and click on the button **Filter...** to get to the dialog **Administrate Filters**. Create two new filters by clicking on the button  and name them "Selected" and "Not Selected".



Now, please set the filter conditions. To the filter "Not Selected", please assign the condition "Selected not equal 1". Due to this condition, only unselected nodes will be filtered. Now please mark the filter **Not Selected** and click on the  button right-hand at the top of the dialog. It will invoke the **Edit Filter** dialog. In the column **Fieldname** please choose the field **Selected**, in the column **Operator** please choose **not equal** and in the column **Comparison value** please enter the value 1. Quit the dialog by **OK**.

56 Creating Histograms

Subconditions

Fieldname	Operator	Comparison value	And/Or
[Selected]	not equal	1	

☐ Compare hour/min ☒ Case sensitive

OK Cancel Help

Now, in the same way please assign the condition "Selected equal 1" to the filter "Selected".


Subconditions

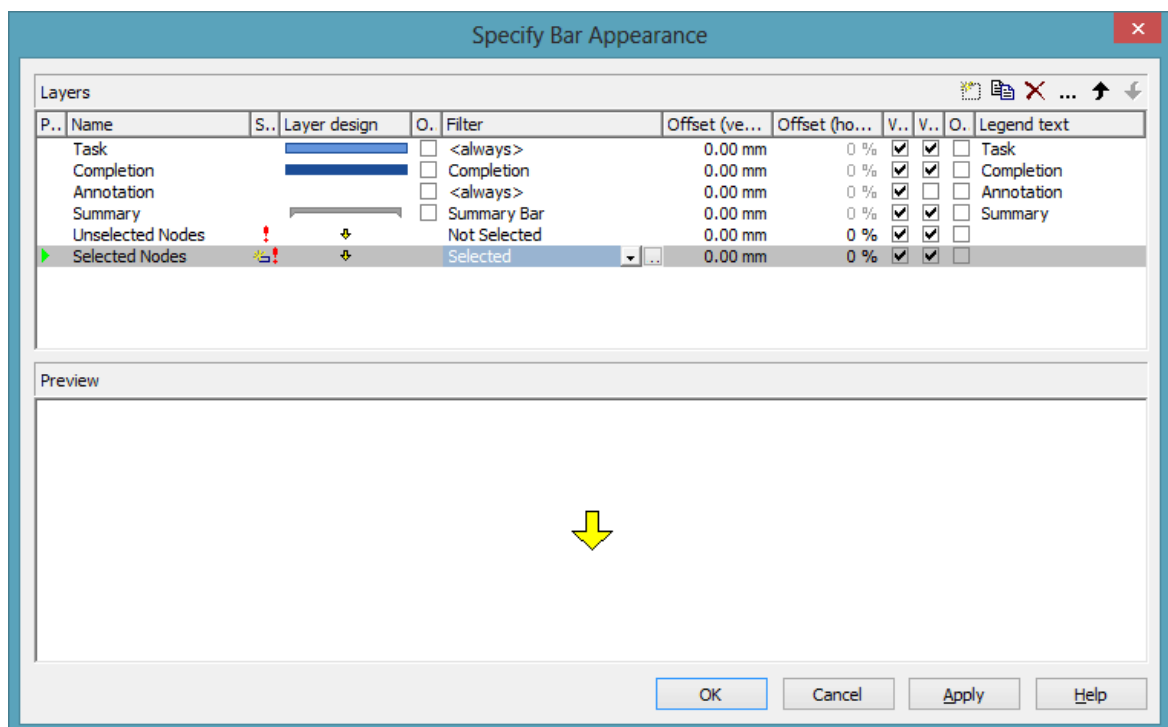
Fieldname	Operator	Comparison value	And/Or
[Selected]	equal	1	

☐ Compare hour/min ☒ Case sensitive

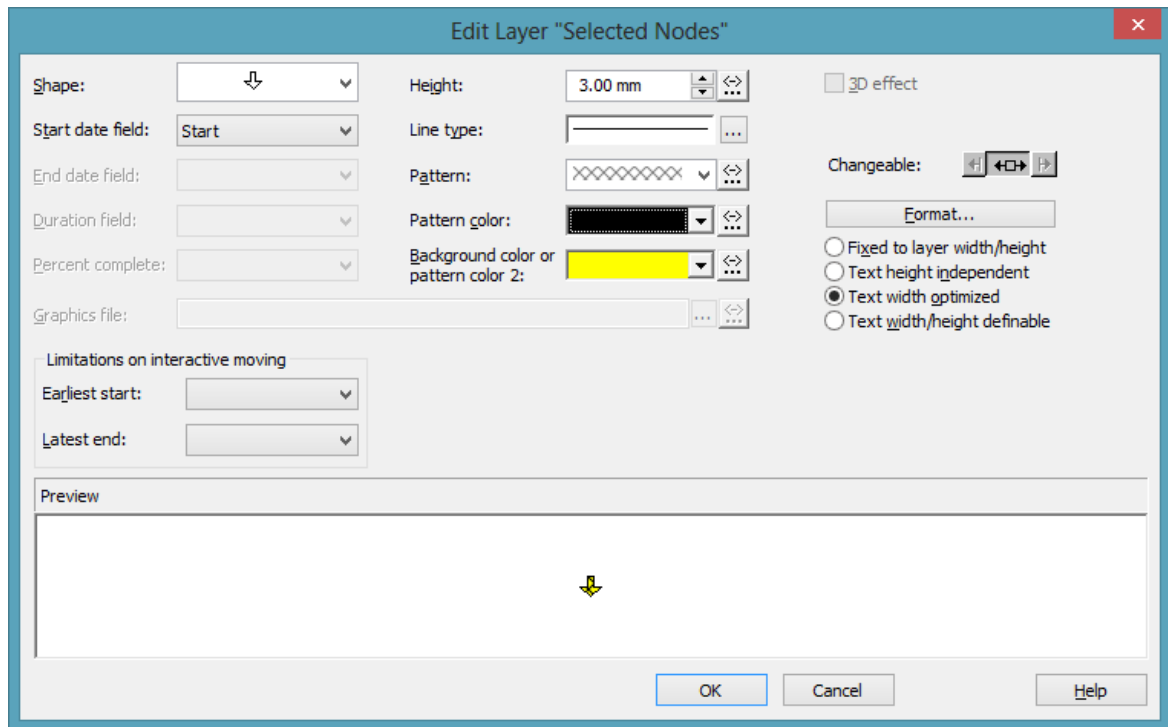
OK Cancel Help

Step 6: In this step, we will define two different appearances for selected and unselected nodes to be combined with the filters.

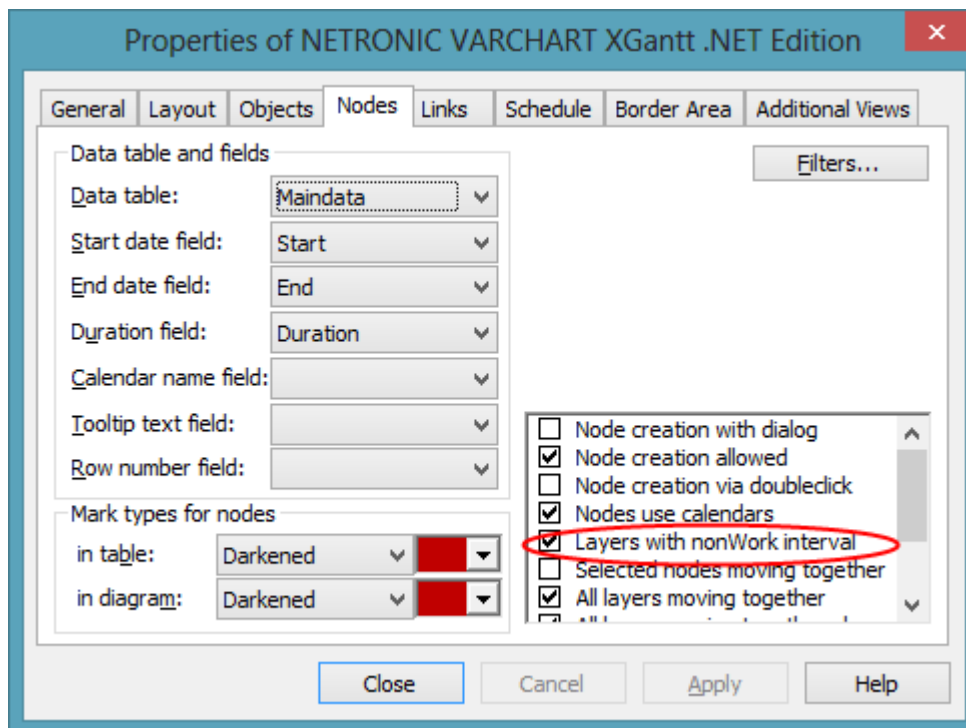
Please select the property page **Objects** and click on the object **Layers....** This will pop up the dialog **Specify Bar Appearance**. Please rename the layer "Start-End" into "Unselected Nodes" by entering the new name directly into the field in the column **Name**. Please find the column **Filter** and assign the filter "Not Selected" to the Layer. Copy the layer by clicking on the button  and name the copy "Selected Nodes". Assign the Filter "Selected" to the layer.



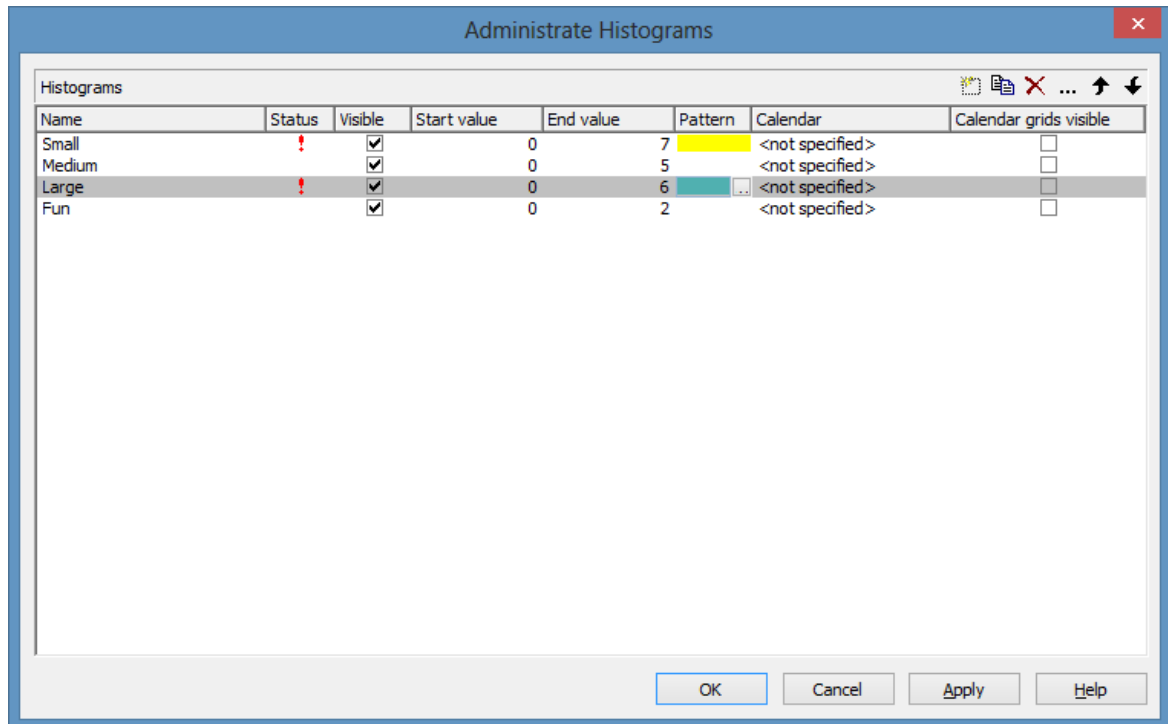
Both layers still look alike. You can modify the design of the layer "Selected Nodes" by double-clicking in the corresponding field of the column **Layer design**. The dialog **Edit Layer** will pop up. Please select a cross hatch **Pattern**, a yellow **Background color or pattern color2** and a black **Pattern color**.



To ensure that weekends in non work intervals are displayed as a line instead of a bar, the option **Layers with nonWork interval** needs to be set on the **Nodes** property page.




Step 7: In this step, four curves will be created for the histogram: the capacity curve, the curve of unmarked activities, the curve of marked activities and an auxiliary curve to fill an area. Click on **Administer histograms...** on the property page **Layout** to invoke the corresponding dialog.



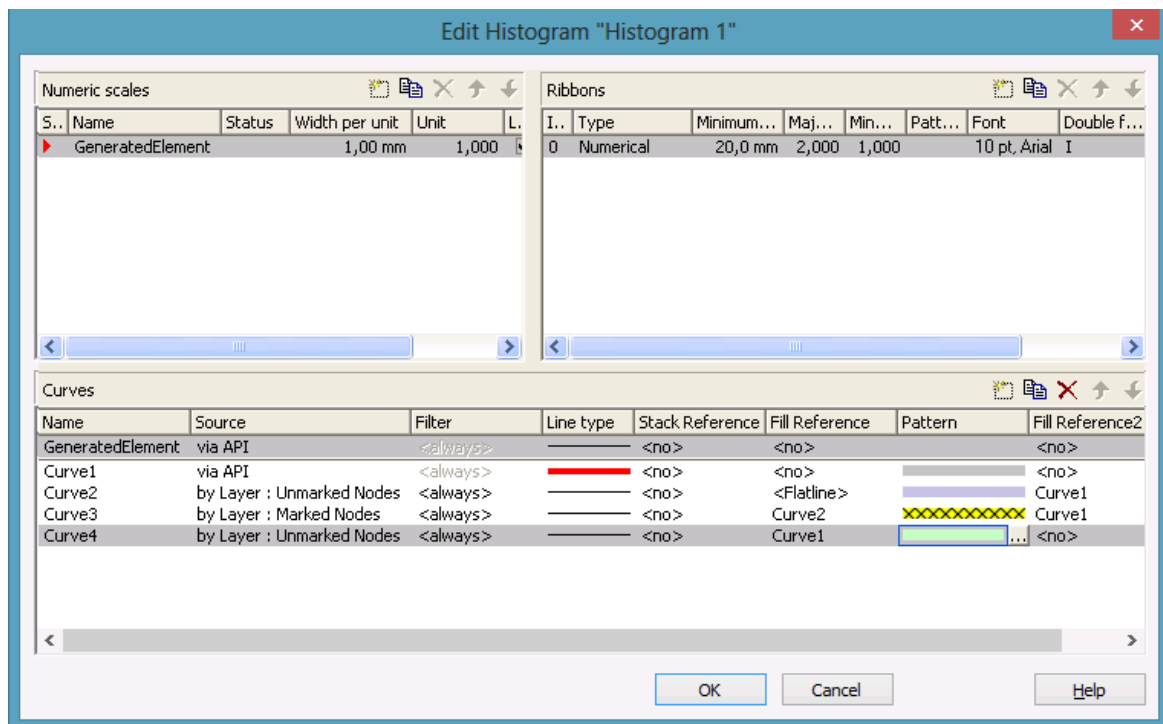
Several histograms may be present in a Gantt chart at the same time. Each of the histograms has a numeric scale of its own and contains its own curves.

Please now define the start and end values of the numeric scale. Click in the **Histogram_1** field of the **End value** column and enter 6.

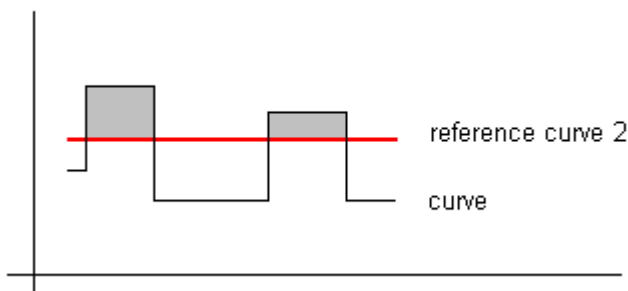
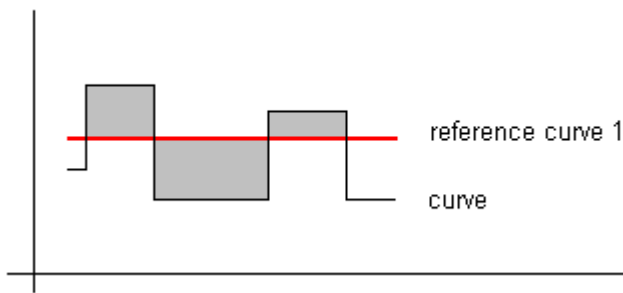
Now, please edit the histogram. For this, please click on the **Edit** button  right-hand at the top of the dialog.

"Curve 1" shall represent the capacity curve (in red). "Curve 2" shall summarize the marked nodes while "Curve 3" shall represent the unmarked nodes. "Curve 4" is an auxiliary curve to provide the green background of the shortfall areas.

One curve already exists. Please create three further curves and define their properties according to the illustration.



To a curve, two reference curves can be assigned at maximum. A curve forms areas with its reference curves, to which colors and patterns can be assigned (see sketches below). Of the first reference curve, all parts that form above and below the curve add up to the area (top sketch). Of the second reference curve, only those parts add to the area that form below the curve, that is, of which the Y-values are smaller than those of the original curve (bottom sketch). In addition, areas formed by the second reference curve are displayed at higher priority. We will see below, what the consequences will be in the histogram.



The capacity curve (Curve 1) will receive its values from a list, that we are going to provide later on by programming code. Therefore, please set its data source in the field **Source** to **via API**. Because of this, an additional filter for nodes from which data can be taken is not needed.

Please set the **Line type** to a thick red line. The values of this curve shall not be added to the values of another curve; therefore the field **Stack reference** remains empty. Also, the capacity curve is not intended to form an area with another curve, therefore the two fill references and their fill patterns remain empty. Please create Curve 1 as described by clicking on the corresponding fields in the dialog.

Curve 2 represents the nodes not selected and composes of values from the layers named "Unselected Nodes". A filter for further selection is not needed. Please choose a blue line color for the curve line. The curve values will not be added to the values of another curve, so the **Stack Reference** remains empty. The curve is supposed to form an area with the X-axis, so please in the field **Fill Reference** select the value **Flatline**.

This curve, consisting of the non-selected nodes, should also indicate in a special way, where it exceeds the capacity curve in order to mark the bottlenecks of the production system. Therefore, as soon as its Y-values exceed those of Curve 1, the area below shall be hatched. So please set Curve 1 as its second reference curve and select a hatched fill pattern.

"Curve 3" shall represent the selected nodes. So please, as its data source, assign the layers named "Selected Nodes". A filter is not needed. Please assign a light gray line color. Since the selected nodes shall be displayed

above the non-selected nodes, their values have to be added to the ones of the non-selected nodes. So please choose Curve 2 as the **Stack Reference**. The same curve also serves as the first reference curve, since the selected nodes visually shall differ from the non-selected ones. As a fill pattern, please select a gray cross hatch pattern on a yellow background.

The area formed will be visible above and below Curve 2. In addition, it shall appear above the capacity curve; therefore please assign Curve 1 as the second reference curve and fill the area with the same color and pattern. If selected nodes rise above the capacity curve, they will appear in the same color and pattern as below the capacity curve (you could distinguish between selected nodes above and below the capacity curve by assigning e.g. a red color here).

By curve 4 we are going to define an area that represents the light green background between the capacity curve and the node piles below. It indicates available resources of the production system. It is limited at its bottom by the unselected nodes, so please choose them as the data source. At the top, the area is limited by the capacity curve, which you please set as the first reference curve.

Question: Why does the area of Curve 4 not hide the selected nodes?
 Answer: Because there is a priority in the list of curves presented by this dialog. The curves listed at the bottom have a lower priority than those listed at the top. This is why areas of curve 3 are displayed on top of areas of curve 4. You can modify the priority by the arrows right-hand at the top of the window.

Step 8:

In the final step, we are going to provide the values of the capacity curve. For this, please modify the code in the **Load** event as shown below:

Example Code VB.NET

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    VcGantt1.Width = ClientSize.Width - VcGantt1.Left
    VcGantt1.Height = ClientSize.Height - VcGantt1.Top

    VcGantt1.InsertNodeRecord("1;Node 1;07.05.2007;;5")
    VcGantt1.InsertNodeRecord("2;Node 2;09.05.2007;;5")
    VcGantt1.InsertNodeRecord("3;Node 3;10.05.2007;;6")
    VcGantt1.InsertNodeRecord("4;Node 4;17.05.2007;;10")
    VcGantt1.InsertNodeRecord("5;Node 5;22.05.2007;;3")
    VcGantt1.InsertNodeRecord("6;Node 6;23.05.2007;;1")

    VcGantt1.EndLoading()
```

```

VcGantt1.OptimizeTimeScaleStartEnd(3)

'calculate end date
Dim node As VcNode

For Each node In VcGantt1.NodeCollection
    setNodeEndDate(node)
Next
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.FirstHistogram
curve = histogram.CurveCollection.CurveByName(" Curve1 ")
curve.PointsEquidistant = False
curve.SetValues("01.05.2007", "2")
curve.SetValues("05.05.2007", "0")
curve.SetValues("07.05.2007", "2")
curve.SetValues("12.05.2007", "0")
curve.SetValues("14.05.2007", "4")
curve.SetValues("19.05.2007", "0")
curve.SetValues("21.05.2007", "2")
curve.SetValues("26.05.2007", "0")
curve.SetValues("28.05.2007", "2")

End Sub

```

Example Code C#

```

private void Form1_Load(object sender, System.EventArgs e)
{
    vcGantt1.Width = ClientSize.Width - vcGantt1.Left;
    vcGantt1.Height = ClientSize.Height - vcGantt1.Top;

    vcGantt1.InsertNodeRecord("1;Node 1;07.05.2007;;5");
    vcGantt1.InsertNodeRecord("2;Node 2;09.05.2007;;5");
    vcGantt1.InsertNodeRecord("3;Node 3;10.05.2007;;6");
    vcGantt1.InsertNodeRecord("4;Node 4;17.05.2007;;10");
    vcGantt1.InsertNodeRecord("5;Node 5;22.05.2007;;3");
    vcGantt1.InsertNodeRecord("6;Node 6;23.05.2007;;1");

    vcGantt1.EndLoading();

    vcGantt1.OptimizeTimeScaleStartEnd(3);

    // calculate end date
    foreach (VcNode node in vcGantt1.NodeCollection)
    {
        SetNodeEndDate(node);
    }

    VcHistogram histogram =
    vcGantt1.HistogramCollection.FirstHistogram();
    VcCurve curve = histogram.CurveCollection.CurveByName("Curve 1");
    curve.PointsEquidistant = false;
    curve.SetValues(Convert.ToDateTime("01.05.2007"), "2");
    curve.SetValues(Convert.ToDateTime("05.05.2007"), "0");
    curve.SetValues(Convert.ToDateTime("07.05.2007"), "2");
    curve.SetValues(Convert.ToDateTime("12.05.2007"), "0");
    curve.SetValues(Convert.ToDateTime("14.05.2007"), "4");
    curve.SetValues(Convert.ToDateTime("19.05.2007"), "0");
}

```

```

curve.SetValues(Convert.ToDateTime("21.05.2007"), "2");
curve.SetValues(Convert.ToDateTime("26.05.2007"), "0");
curve.SetValues(Convert.ToDateTime("28.05.2007"), "2");
}

```

Now, please run the program and mark an activity. You can recognize immediately by the gray hatched section on a yellow background in the histogram, what part the activity occupies in the bulk of the work load displayed.

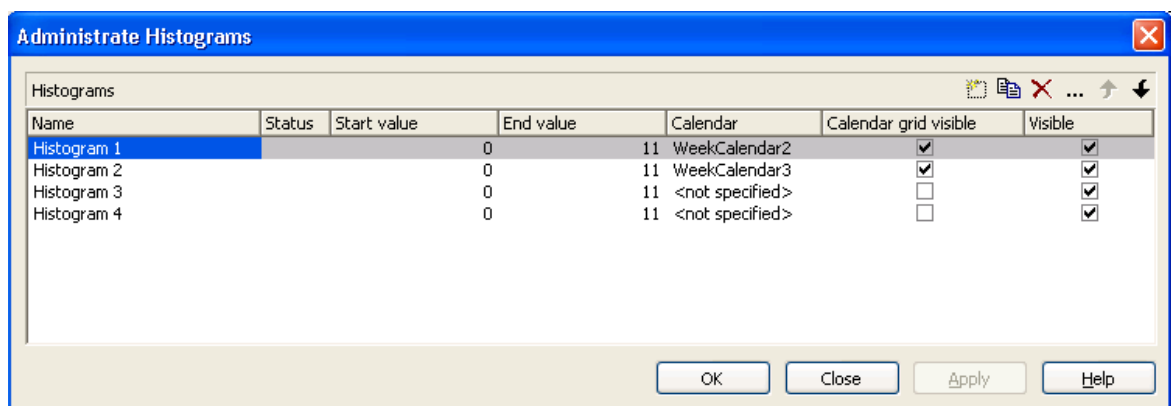
If you move activities, the workload will change and you can recognize capacity overloads and shortfalls caused by your interaction.

Calendar Grids in Histograms

You can assign one ore more calendar grids to a histogram, so that different calendar grids in the Gantt graph can also become visible in the histogram.

To have an own calendar grid assigned to a histogram, three conditions have to be fulfilled:

1. A calendar has to be assigned to the histogram
2. The calendar grid has to be switched on
3. An appearance has to be defined that enables the display of the calendar grid



Calendar assigned, calendar grid switched on

The corresponding API calls are:

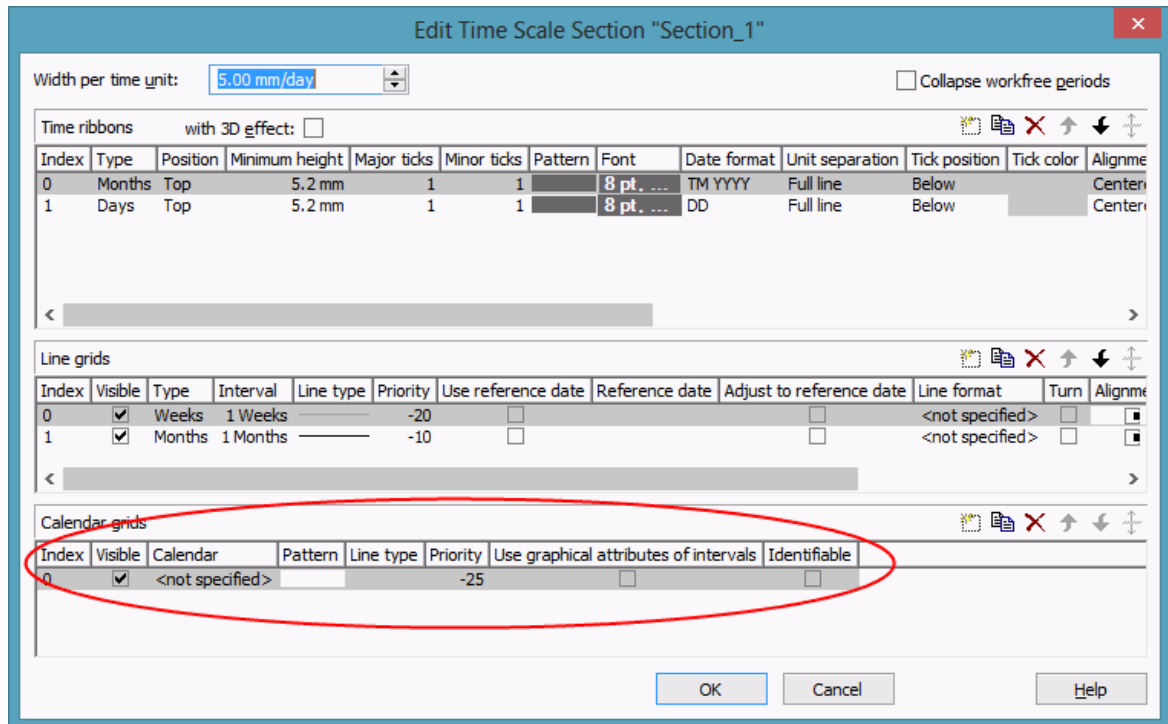
Example Code VB.NET

```

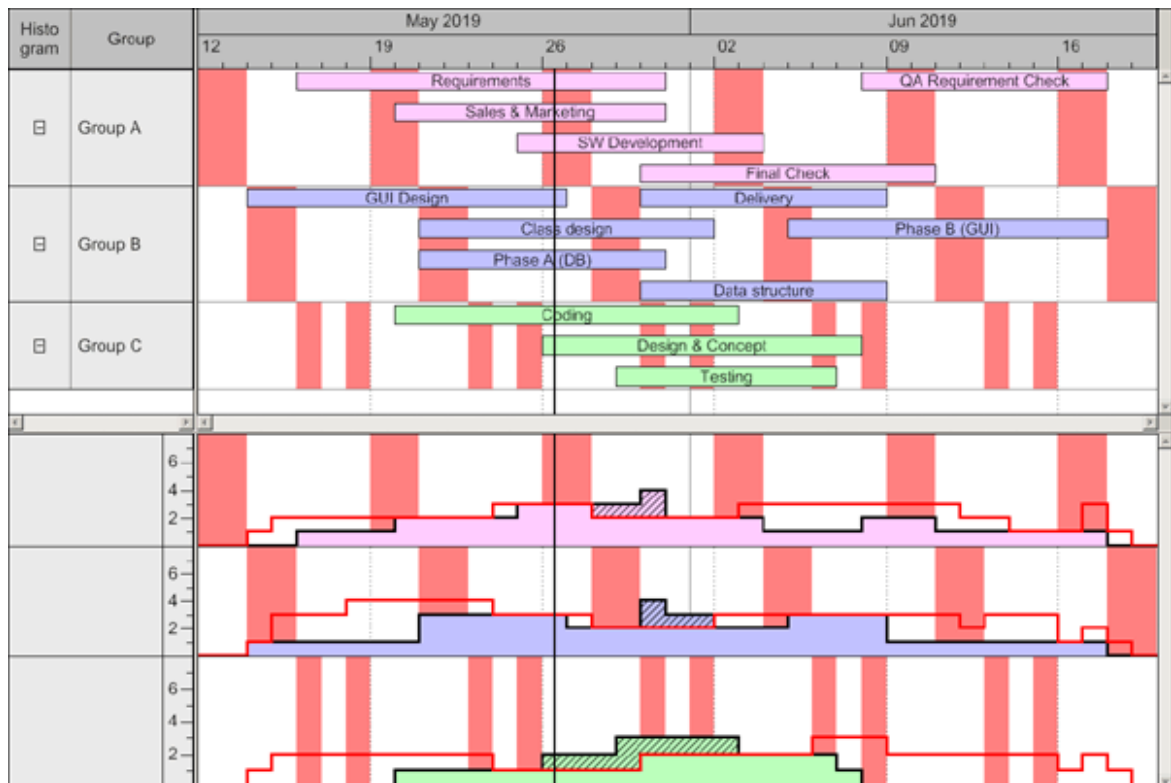
// assigning the calendar to the histogram (by the calendar name)
histogram.calendarName = group.DataField(14)
// switching the calendar grid on
histogram.ShowCalendarGrids = True
// setting the histogram visible
histogram.Visible = True

```

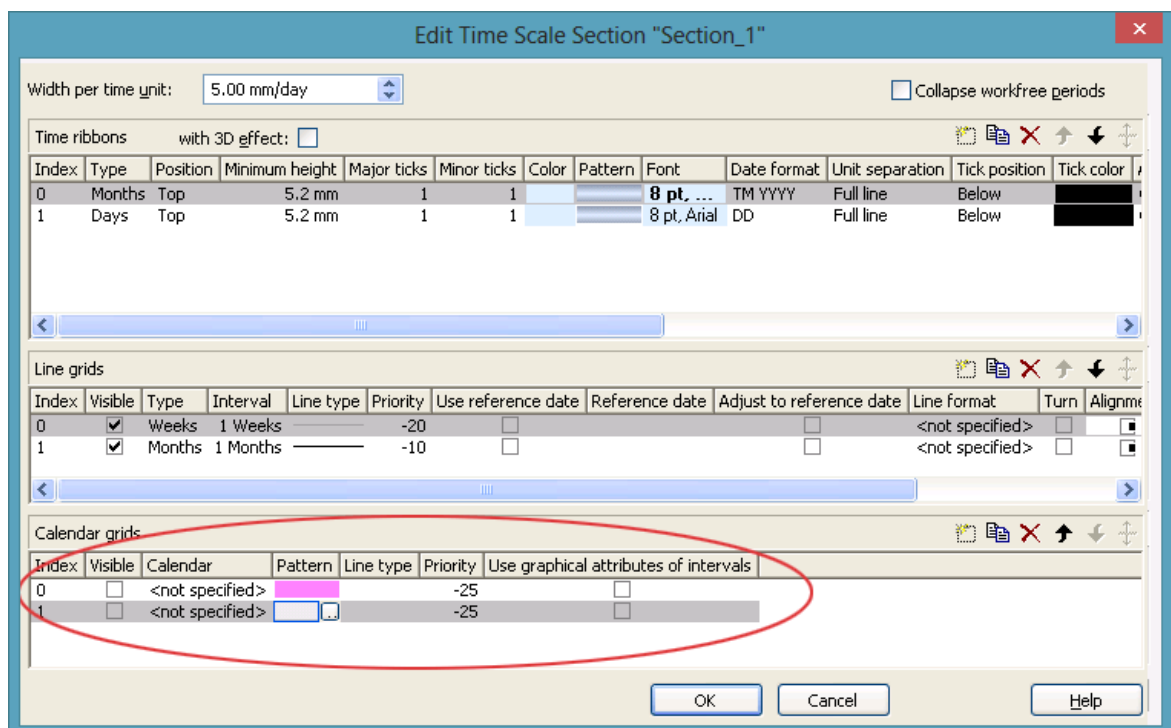
As a calendar grid for the histogram VARCHART XGantt takes the first invisible calendar grid in the first section of the time scale, if there is no other one present. This is the same calendar grid that is used groupwise in the Gantt graph:



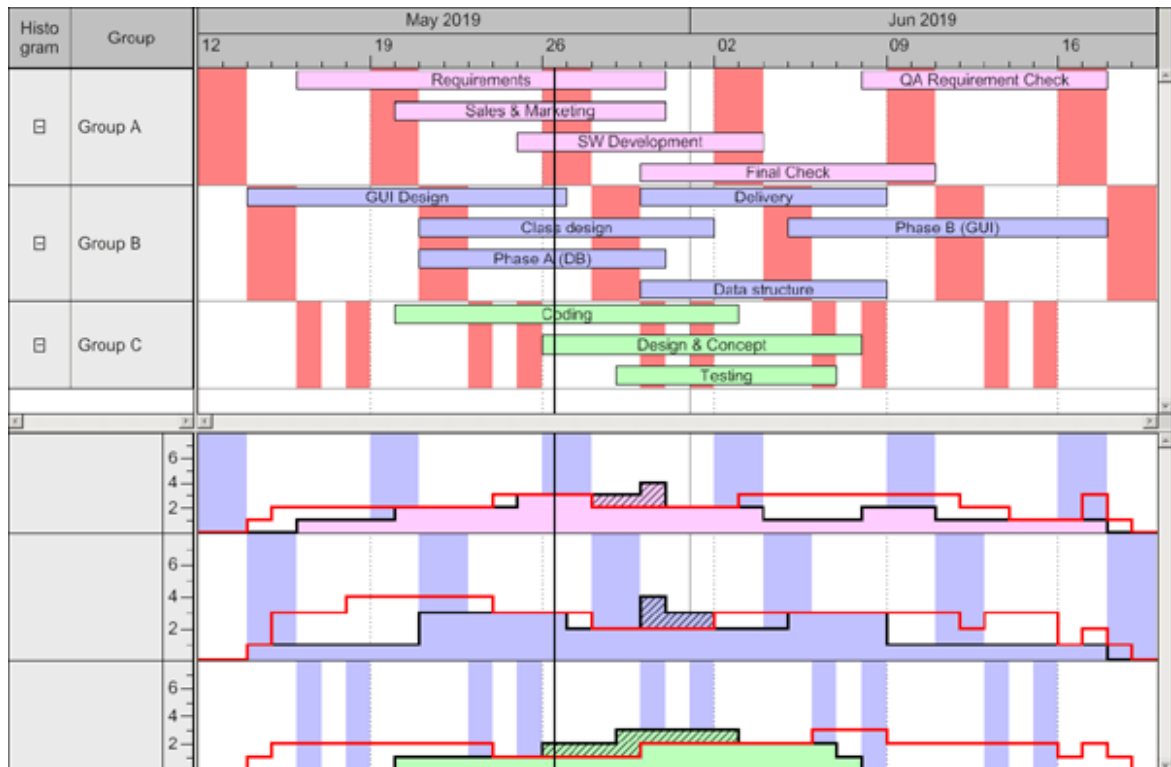
Thus the calendar grid will display the same appearance in the Gantt graph as in the histogram. In the example below it is a calendar grid that shows a different pattern for each group (groupwise calendar grid):



If you set another calendar grid to the time scale section, VARCHART XGantt will use this one for its histograms:



By using the second calendar grid, you can assign a different appearance compared to the calendar grid in the Gantt Graph. In our case, it shows a different color:



2.11 Printing the Diagram

If you have finished modeling your diagram, you can finally print it. In runtime mode, select **Print** from the context menu (right mouse click in the empty diagram). This will take you to the Windows **Printing** dialog.

You also can use the method **ShowPrintDialog** of the object VcGantt to trigger the printing of the diagram.

If you want to edit the printer settings in runtime mode, you can select the menu item **Print setup...** from the context menu and pop up the corresponding Windows dialog.

The method **PrintEx** of the object Vc Gantt lets you print the diagram directly. A dialog box will not be displayed.

If you want to edit the page settings at runtime, you can select **Page setup...** from the context menu or select **Print Preview** in the context menu and there click on the **Page Setup...** button.

You can also use the method **ShowPageSetupDialog** of the object VcGantt to open the corresponding dialog.

In the **Page Setup** dialog you can set e.g. the scaling, whether the pages shall be numbered, the margins, the alignment etc. For further information please see chapter 5.23 "Setting up Pages".

2.12 Exporting a Diagram

You can export a diagram into a graphics file. There are two different ways to this:

- Please select the menu item **Export graphics** from the default context menu. From there you can get to the Windows dialog **Save as**, that lets you save the diagram as a graphics file.
- Use the API method **ShowExportGraphicsDialog** or **ExportGraphicsToFile**.

Please find detailed information on graphics formats in the chapter: **Important Concepts:Graphics Formats**.

2.13 Saving the Configuration

All settings made on the property pages at design time are added to your project as a resource. Changes come into operation only after saving your project, since only then the embedded resource will be updated.

Tip: For this reason, you should activate in Microsoft Visual Studio .NET 2005 the Option **Save all changes** in **Tools > Options > Environment > Projects and Solutions > Build and Run**, so that your settings are automatically saved before compiling.

If you do not select this option, you will have to save your project manually if you want the settings of the property pages to be used in the program.

You can store the settings of the property pages to a configuration outside your project at any time and load them when needed. This is very useful if you want to use previous settings again or if you need the same settings for different projects.

A stored configuration consists of two files of identical names but different extensions, (INI and IFD), that both are indispensable.

How to save your current configuration:

On the **General** property page please click on the **Export...** button and enter the name of the INI file. The ifd-file of the same name will be created automatically.

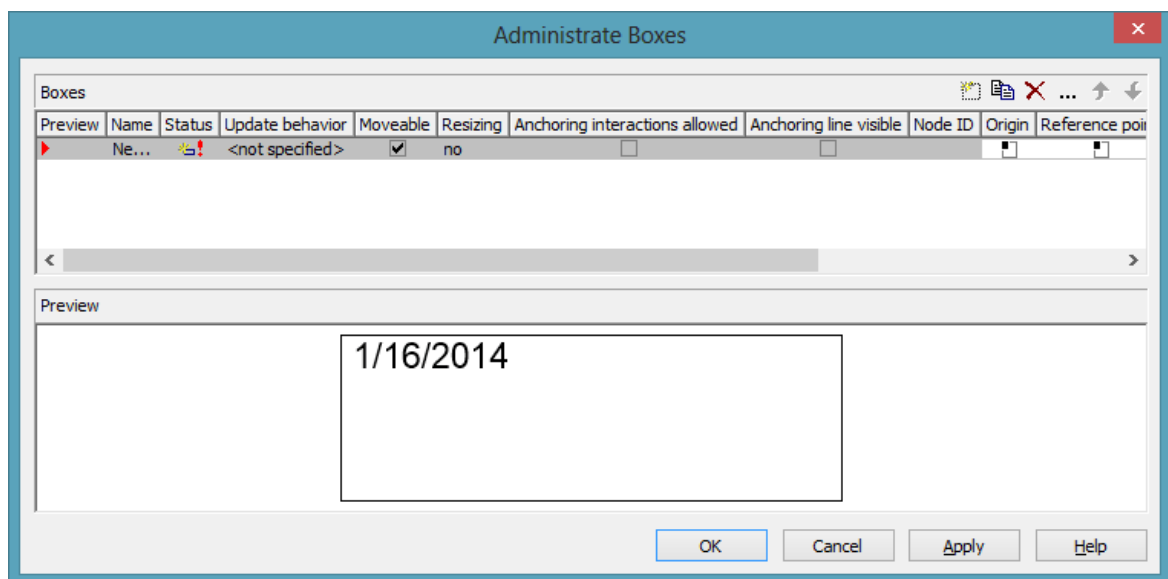
How to load a stored configuration:

On the **General** property page please click on the **Import...** button and select the desired file.

3 Important Concepts

3.1 Boxes

In the diagram area, boxes that contain texts or graphics can be displayed. To generate boxes, please select the property page **Objects** and press the **Boxes...** button. The dialog **Administrate Boxes** will open, where you can add, copy, delete or edit boxes.



The properties **Origin**, **Reference Point**, **X Offset** and **Y Offset** allow to exactly position a box in the diagram area. The relative position of the boxes is independent of the current diagram size.

For each box you can specify

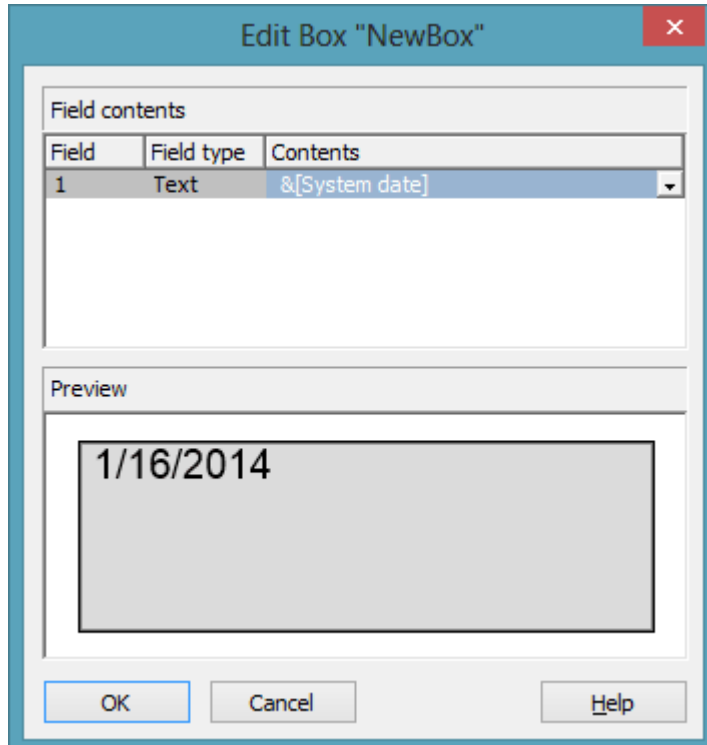
- its name
- whether the box can be moved in the diagram at run time
- whether and how the size of the box can be modified interactively
- whether anchoring interactions by mouse or over context menu are possible
- whether the reference points of the node and of the box (origin, reference point) shall be linked by a line when using the anchoring tool
- a node ID to identify the node to which the respective box shall be tied

72 Important Concepts: Boxes

- its point of origin (a point in the diagram to which the reference point refers to form what is called "the offset")
- its reference point, i. e. the complementary point of the box to form the offset
- its X or Y Offset (distance between origin and reference point in x or y direction)
- type, thickness and color of the box frame line
- its priority in comparison to other diagram objects (nodes, grids, etc.)
- whether the box is visible
- its format

> Editing boxes

The **Edit Box** dialog lets you specify the contents of the fields. This dialog box will appear at design time when you click the **Edit box** button in the **Administrate Boxes** dialog box. At run time it will appear when you double-click the box to be edited. You also can edit the texts of boxes directly at run time after having selected **In-place editing allowed** on the property page **General**



The **Field** column contains the numbers of the box fields. (The number of fields depends on the selected box format.)

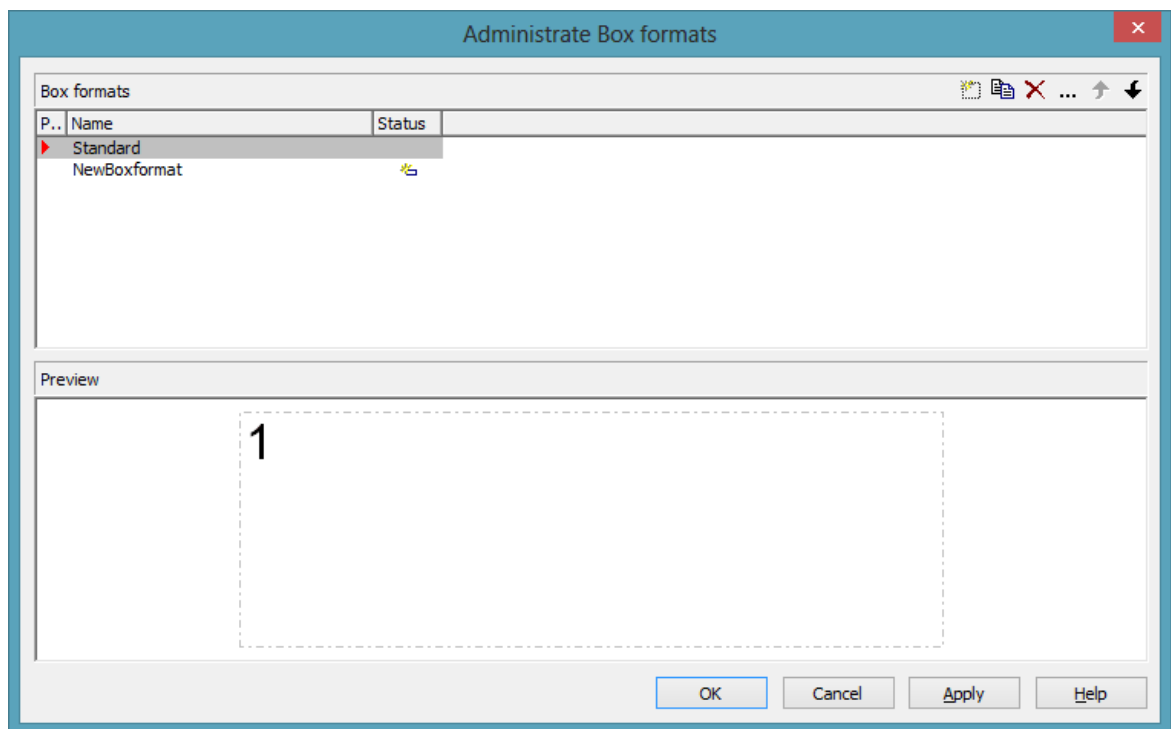
The **Field type** column displays the field types (text or graphics).

You can enter the text of the field or a graphics file name into the **Content** column. If a text field contains more than one line, you can use "\n" in the text string to separate two lines of the text field (Example: "Line1\nLine2"). Without the line feed symbol the lines will automatically be separated where blanks occur.

> **Box formats**

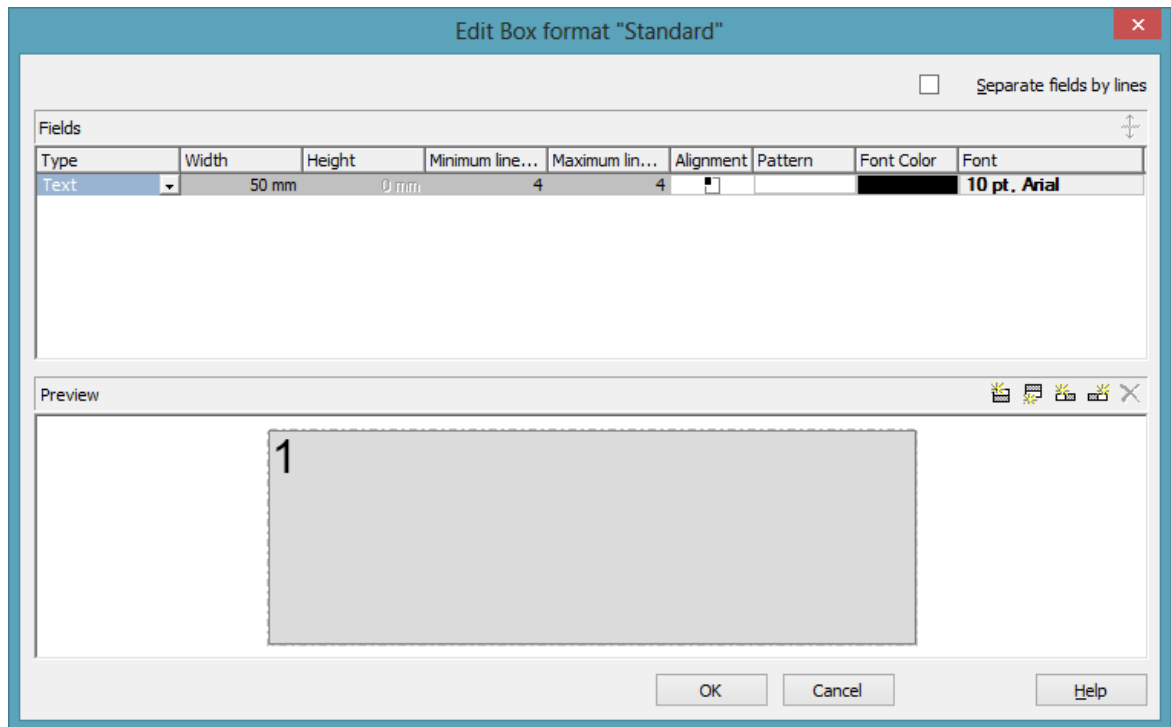
For each box you can select a box format, and you can specify the box formats.

In the **Administrate Box Formats** dialog box you can add, copy, delete or edit box formats. Click the corresponding button on the **Objects** property page to open this dialog.



In the **Edit Box Format** dialog box you can specify the box format. Click the **...** button in the **Administrate Box Formats** dialog box to open this dialog.

74 Important Concepts: Boxes



You can specify whether the box fields are to be separated by lines.

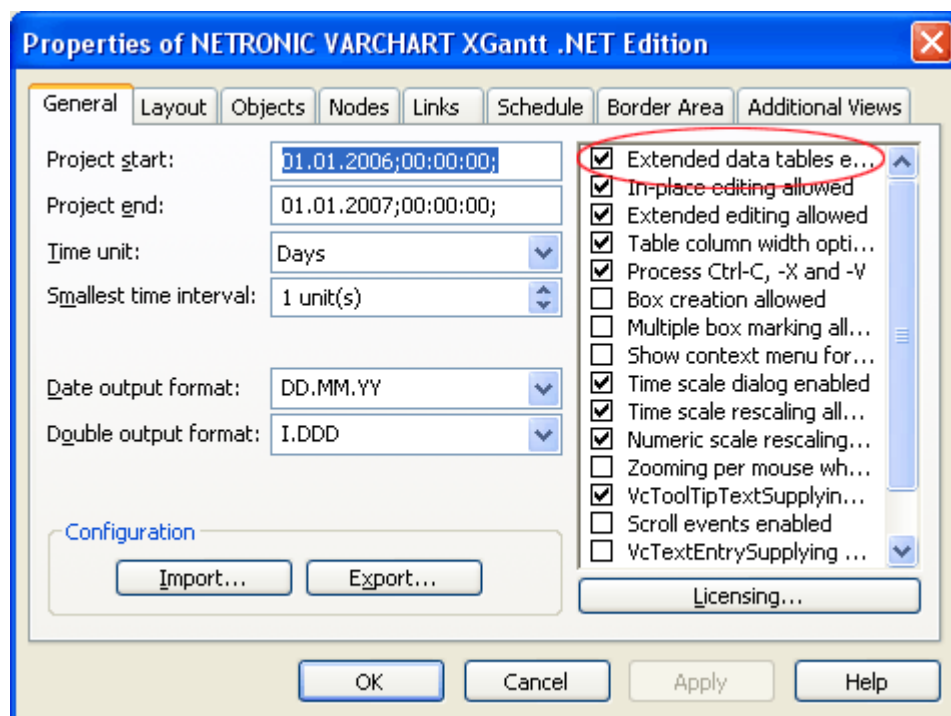
Furthermore, the following items can be specified for each box :

- field type (text or graphics)
- width and height
- how many lines of text can be displayed in the current field
- alignment
- background color and fill pattern
- font attributes

3.2 Data Tables

As a data base for the graphical display of Gantt charts VARCHART XGantt originally used two standard data tables for nodes and links, the fields of which can be individually defined. In version 4.0 this concept was extended. Up to 90 data tables can be defined and 1:n relations can be set up between the tables. This helps avoiding redundancies in many cases; it allows to access the main data record by the depending data record and supplies the data required by the resource scheduling module integrated in VARCHART XGantt.


For reasons of compatibility to existing applications VARCHART XGantt continues to operate in the previous mode. Only by activating the corresponding option at design time or at run time the extended data tables can be used. You can find the option **Extended data tables enabled** on the property page **General**:






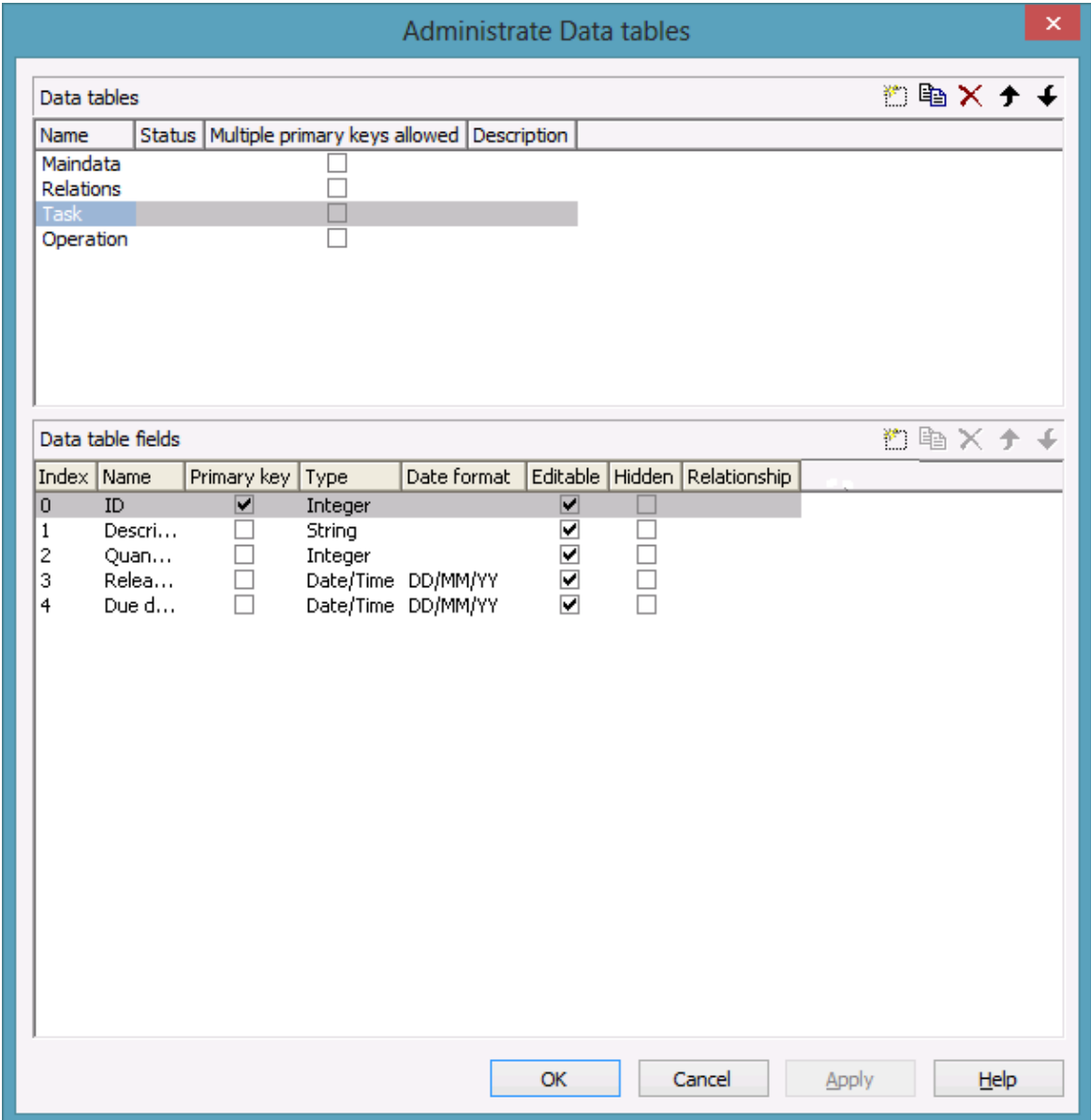
In the programming interface, the extended data tables are switched on at runtime by setting the VcGantt property **ExtendendDataTablesEnabled** to **True**.

> Handling Data Tables

By default, the data tables **Maindata** and **Relations** exist. On the property page **Objects** you can click on the button **Data tables...** to get to the dialog

Administrate Data Tables. Generating new data tables requires to have switched on the **Extended data tables** mode before. The data tables **Task** and **Operation** in the picture below were created by clicking on  in the section **Data Tables**.

In the section **Data Table Fields** you can edit the fields of the above selected table. You can generate new fields by , delete existing fields by  or copy fields by , as shown below.



Name	Status	Multiple primary keys allowed	Description
Maindata		<input type="checkbox"/>	
Relations		<input type="checkbox"/>	
Task		<input checked="" type="checkbox"/>	
Operation		<input type="checkbox"/>	

Index	Name	Primary key	Type	Date format	Editable	Hidden	Relationship
0	ID	<input checked="" type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
1	Descri...	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
2	Quan...	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
3	Relea...	<input type="checkbox"/>	Date/Time	DD/MM/YY	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
4	Due d...	<input type="checkbox"/>	Date/Time	DD/MM/YY	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

OK Cancel Apply Help

The column **Index** is essential when using the API, since the contents of the data fields can only be addressed via the index. If you modify the sequence of fields in this dialog, i.e. the index, after having produced programming code, you need to adapt the programming code that accesses the corresponding field.

If you modify the data type, you may accordingly have to adapt formats and layers already defined to ensure that the appropriate data type is used when the fields are accessed.

The primary key feature is to be set to a field if you want a data record to be unique and thus distinguishable. The primary key may also consist of more fields, but only up to three. For a detailed description of the use of composite primary keys see chapter **The Administrative Data Tables Dialog Box**.

For a data table referred to by a relation, selecting a field to be the primary key is compulsory.

Relating tables is useful if the content shows a 1:n relation and if a subordinated data record should directly refer to a data field of the main data record.

Between two tables A and B at the moment only a single 1:n relationship can be established; a second field of B is not allowed to relate to the primary key of A. Nevertheless, a field of a third table C is allowed to relate to the primary key of table A.

Note: If a data table with a composite primary key is used in a relationship, the relationship has to match the primary key. Otherwise a unique connection is not possible. If the relationship is not defined correctly - which is checked neither at the API nor in the **Administrative Data Tables** dialog, the data record will not be connected. This leads to the event **VcDataRecord-NotFound**.

In the sample below a relation is created between the tables **Operation** and **Task** by setting **Task:ID** in the column **Relationship**.

78 Important Concepts: Data Tables

Administrate Data Tables

Data Tables

Name	Status	Multiple primary keys allowed	Description
Maindata		<input type="checkbox"/>	
Relations		<input type="checkbox"/>	
Task		<input type="checkbox"/>	
Operation	!	<input type="checkbox"/>	

Data Table Fields

Index	Name	Primary key	Type	Date format	Editable	Hidden	Relationship
0	ID	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
1	TaskID	<input type="checkbox"/>	Integer		<input type="checkbox"/>	<input type="checkbox"/>	Task:ID
2	Description	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

OK Cancel Apply Help

Table Task:

ID	Description	Quantity	Release date	Due date
1	Task 1	10	12.05.13	20.05.13
2	Task 2	20	01.06.13	15.06.13

Table Operation:

ID	TaskID	Description	Start	End
1	1	Operation 1	12.05.13	14.05.13
2	1	Operation 2	15.05.13	19.05.13

ID	TaskID	Description	Start	End
3	2	Operation 3	01.06.13	05.06.13
4	2	Operation 4	05.06.13	11.06.13
5	2	Operation 5	11.06.13	15.06.13

Example Code VB.NET

```

Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcGantt1.DataTableCollection
dataTable = dataTableCltn.DataTableByName("Task")

dataTable.DataRecordCollection.Add("1;Task 1;10;12.05.2013;20.05.2013")
dataTable.DataRecordCollection.Add("2;Task 2;10;01.06.2013;15.06.2013")

dataTable = dataTableCltn.DataTableByName("Operation")
dataTable.DataRecordCollection.Add("1;1;Operation
1;12.05.2013;14.05.2013")
dataTable.DataRecordCollection.Add("2;1;Operation
2;15.05.2013;19.05.2013")
dataTable.DataRecordCollection.Add("3;2;Operation
3;01.06.2013;05.06.2013")
dataTable.DataRecordCollection.Add("4;2;Operation
4;05.06.2013;11.06.2013")
dataTable.DataRecordCollection.Add("5;2;Operation
5;11.06.2013;15.06.2013")

VcGantt1.EndLoading()

```

Example Code C#

```

VcDataTableCollection dataTableCltn = vcGantt1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.DataTableByName("Task");

dataTable.DataRecordCollection.Add("1;Task 1;10;12.05.2013;20.05.2013");
dataTable.DataRecordCollection.Add("2;Task 2;10;01.06.2013;15.06.2013");

dataTable = dataTableCltn.DataTableByName("Operation");
dataTable.DataRecordCollection.Add("1;1;Operation
1;12.05.2013;14.05.2013");
dataTable.DataRecordCollection.Add("2;1;Operation
2;15.05.2013;19.05.2013");
dataTable.DataRecordCollection.Add("3;2;Operation
3;01.06.2013;05.06.2013");
dataTable.DataRecordCollection.Add("4;2;Operation
4;05.06.2013;11.06.2013");
dataTable.DataRecordCollection.Add("5;2;Operation
5;11.06.2013;15.06.2013");

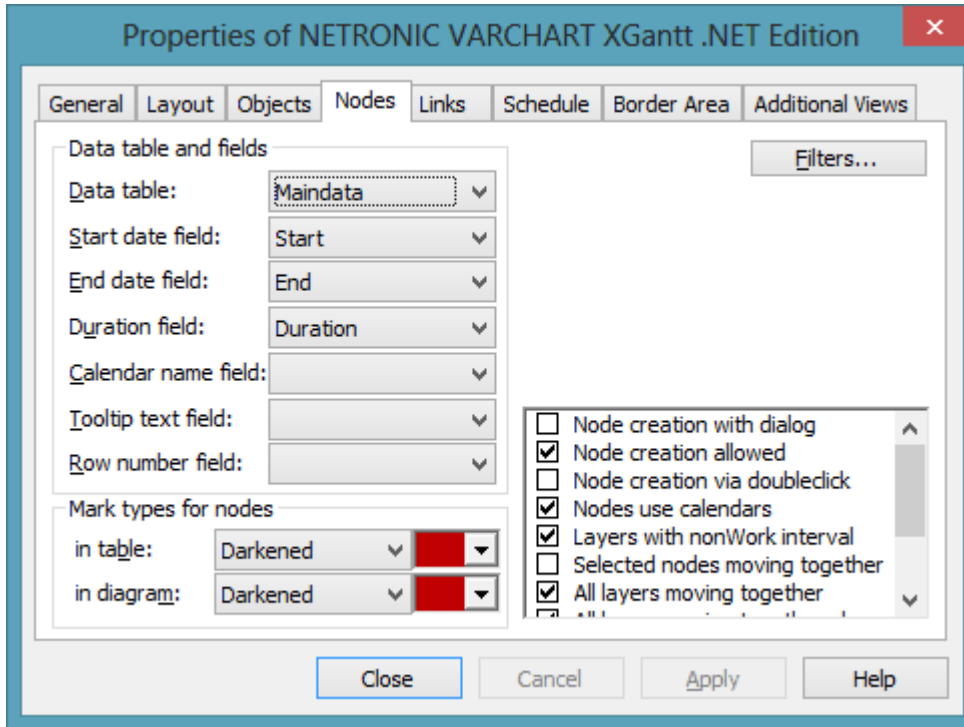
vcGantt1.EndLoading();

```

Depending on the data table selected on the property page **Nodes** in the **Data table and fields** section, the graphical display of the nodes may originate

80 Important Concepts: Data Tables

from different bases. When creating nodes interactively, the base is the table to which new data records are added automatically. The corresponding rows displayed by the visualization are influenced by the active node filter, by grouping and by display options.



This is the result in the table of the Gantt chart if the table **Operation** was taken as base. The entries for Description, Quantity and Due date originate from the main table **Task**.

Description	Quantity	Due date	Operation
Task1	10	20.05.13	Operation1
Task1	10	20.05.13	Operation2
Task2	20	15.06.13	Operation3
Task2	20	15.06.13	Operation4
Task2	20	15.06.13	Operation5

If the table **Task** instead of **Operation** is used, the visible table in XGantt will consist of two entries only.

ID	Description	Quantity	Due date	Operation
1	Task 1	10	20.05.13	
2	Task 2	20	15.06.13	

In version 4.0 of VARCHART XGantt new object types are available that will replace the former ones. For reasons of compatibility, the former object types have been preserved in the present version. In new applications and in updates of existing applications the new objects should be used only.

Former	Present from Version 4.0 Onward
VcDataDefinition	VcDataTableCollectionVcDataTable
VcDataDefinitionTable	VcDataTableFieldCollection
VcDefinitionField	VcDataTableField
	VcDataRecordCollection
	VcDataRecord

> Creating and modifying data records

After having defined the data table fields, you can add data records to a table by the API. There are two ways of adding data to your records. We recommend the common practice of defining an array of the type object with the number of its elements corresponding to the number of the data table fields.

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection

Dim dataRecVal() As Object
Dim dataRec1 As VcDataRecord
Dim dataRec2 As VcDataRecord

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection

ReDim dataRecVal(dataTable.DataTableFieldCollection.Count)

dataRecVal(Main_ID) = "1"
dataRecVal(Main_Name) = "Node 1"
dataRecVal(Main_Start) = DateSerial(2013, 1, 8)
dataRecVal(Main_Duration) = 8
```

Example Code C#

```
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

Object [] dataRecVal = new
object[dataTable.DataTableFieldCollection.Count];

VcDataRecord dataRec1;
VcDataRecord dataRec2;
```

82 Important Concepts: Data Tables

```
dataRecVal[Main_ID] = "1";  
dataRecVal[Main_Name] = "Node 1";  
dataRecVal[Main_Start] = "08.01.2013";  
dataRecVal[Main_Duration] = 8
```

A data record can be added by the method **Add()** of the object **DataRecordCollection**, the object array being passed as parameter.

Example Code VB.NET

```
dataRec1 = dataRecCltn.Add(dataRecVal)
```

Example Code C#

```
dataRec1 = dataRecCltn.Add(dataRecVal);
```

As a second method you can use a string consisting of data values which are separated by a semicolon.

Example Code VB.NET

```
dataRecCltn.Add("2;Node 2;15.01.13;;9")
```

Example Code C#

```
dataRec2.AllData = "2;Activity Y;15.01.13;;9";
```

If a data value contains a semicolon, the character string has to be enclosed in double quotes.

Example Code VB.NET

```
dataRec2 = dataRecCltn.Add("2;""Node 2;"";15.01.13;;9")
```

Example Code C#

```
dataRec2 = dataRecCltn.Add("2;\"Node 2;\";15.01.13;;9");
```

The reference to a data base object can be quickly found via the primary key by using the method **DataRecordByID ()**.

Example Code VB.NET

```
dataRec1 = dataRecCltn.DataRecordByID("1")  
dataRec2 = dataRecCltn.DataRecordByID("2")
```

Example Code C#

```
dataRec1 = dataRecCltn.DataRecordByID(1);
dataRec2 = dataRecCltn.DataRecordByID(2);
```

The contents of single data fields of a data record may be easily modified by using the indexed property **DataField()**. In order to replace all data field contents of a record you can use the property **AllData**.

Example Code VB.NET

```
dataRec1.DataField(Main_ID) = 1
dataRec1.DataField(Main_Name) = "Activity X"
dataRec1.DataField(Main_Start) = DateSerial(2013, 1, 4)
dataRec1.DataField(Main_Duration) = 12
dataRec1.Update()

dataRec2.AllData = "2;Activity Y;18.01.13;;5"
dataRec2.Update()
```

Example Code C#

```
dataRec1.set_DataField(Main_ID, 1);
dataRec1.set_DataField(Main_Name, "Activity X");
dataRec1.set_DataField(Main_Start, "04.01.2014");
dataRec1.set_DataField(Main_Duration, 12);
dataRec1.Update();

dataRec2.AllData = "2;Activity Y;18.01.14;;5";
dataRec2.Update();
```

A modification of a record can only be displayed after the method **Update()** of the object **DataRecord** was called.

Loading the values by using **Alldata** is suitable for quickly displaying all data values at design time and for transferring the data record contents to the record of a different table. You may also use this data format also for information exchange with OLE Drag & Drop.

Example Code VB.NET

```
Dim content As String
content = dataRec1.AllData & vbCr & dataRec2.AllData & vbCr &
dataRec1.DataField(Main_Name)
MsgBox(content)
```

Example Code C#

```
content = dataRec1.AllData + "\r\n" + dataRec2.AllData + "\r\n" +
dataRec1.get_DataField(Main_Name);
MessageBox.Show(content);
```


84 Important Concepts: Data Tables

Note: In order to improve the legibility for data field access, you can define global constants that have names rather than index numbers, which are more descriptive. Below please find the code in its context:

Example Code VB.NET

```
Const Main_ID = 0
Const Main_Name = 1
Const Main_Start = 2
Const Main_Duration = 4

'...

Dim dataRec1 As VcDataRecord
Dim dataRec2 As VcDataRecord

Dim content As String

VcGantt1.TimeScaleEnd = DateSerial(2014, 1, 1)
VcGantt1.TimeScaleStart = DateSerial(2013, 1, 1)

VcGantt1.ExtendedDataTablesEnabled = True
dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection

ReDim dataRecVal(dataTable.DataTableFieldCollection.Count)

dataRecVal(Main_ID) = "1"
dataRecVal(Main_Name) = "Node 1"
dataRecVal(Main_Start) = DateSerial(2013, 1, 8)
dataRecVal(Main_Duration) = 8
dataRec1 = dataRecCltn.Add(dataRecVal)

dataRecCltn.Add("2;Node 2;15.01.13;;9")

VcGantt1.EndLoading()

'...

dataRec1 = dataRecCltn.DataRecordByID("1")
dataRec2 = dataRecCltn.DataRecordByID("2")

dataRec1.DataField(Main_ID) = 1
dataRec1.DataField(Main_Name) = "Activity X"
dataRec1.DataField(Main_Start) = DateSerial(2013, 1, 4)
dataRec1.DataField(Main_Duration) = 12
dataRec1.Update()

dataRec2.AllData = "2;Activity Y;18.01.13;;5"
dataRec2.Update()

content = dataRec1.AllData & vbCr & dataRec2.AllData & vbCr &
dataRec1.DataField(Main_Name)
MsgBox(content)

'...

dataRec2.AllData = "2;""Activity Y;Z"";18.01.13;;5"
```

```
dataRec2.Update()
content = dataRec1.AllData & vbCr & dataRec2.AllData
MsgBox(content)
```

Example Code C#

```
const int Main_ID = 0;
const int Main_Name = 1;
const int Main_Start = 2;
const int Main_Duration = 4;

//...

VcDataRecord dataRec1;
VcDataRecord dataRec2;

string content;

vcGantt1.TimeScaleEnd = Convert.ToDateTime("01.01.2014");
vcGantt1.TimeScaleStart = Convert.ToDateTime("01.01.2013");

vcGantt1.ExtendedDataTablesEnabled = true;
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;
Object [] dataRecVal = new
object[dataTable.DataTableFieldCollection.Count];

dataRecVal[Main_ID] = "1";
dataRecVal[Main_Name] = "Node 1";
dataRecVal[Main_Start] = "08.01.2013";
dataRecVal[Main_Duration] = 8;
dataRec1 = dataRecCltn.Add(dataRecVal);

dataRecCltn.Add("2;Node 2;15.01.13;;9");

vcGantt1.EndLoading();

//...

dataRec1 = dataRecCltn.DataRecordByID(1);
dataRec2 = dataRecCltn.DataRecordByID(2);

dataRec1.set_DataField(Main_ID, 1);
dataRec1.set_DataField(Main_Name, "Activity X");
dataRec1.set_DataField(Main_Start, "04.01.2013");
dataRec1.set_DataField(Main_Duration, 12);
dataRec1.Update();

dataRec2.AllData = "2;Activity Y;18.01.13;;5";
dataRec2.Update();

content = dataRec1.AllData + "\r\n" + dataRec2.AllData + "\r\n" +
dataRec1.get_DataField(Main_Name);
MessageBox.Show(content);

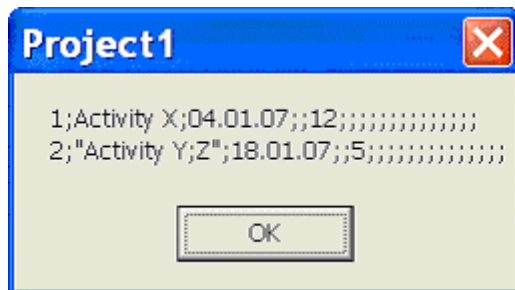
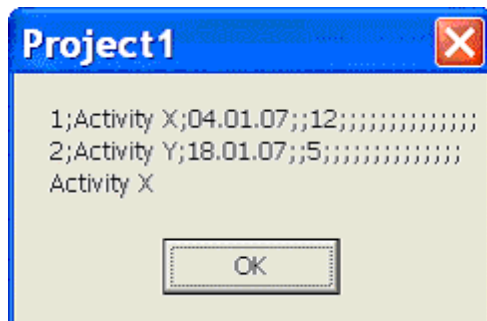
//...

dataRec2.AllData = "2;Activity Y;Z;18.01.13;;5";
dataRec2.Update();
```

86 Important Concepts: Data Tables

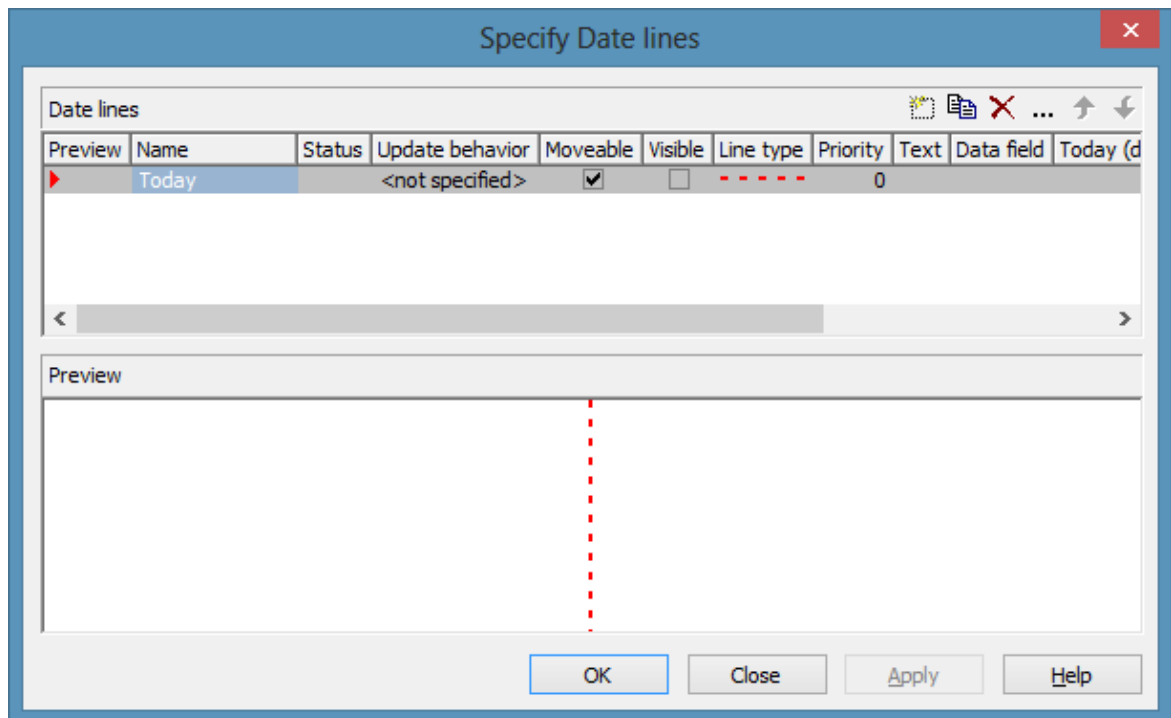
```
content = dataRec1.AllData + "\r\n" + dataRec2.AllData;  
MessageBox.Show(content);
```

The following output will be created:

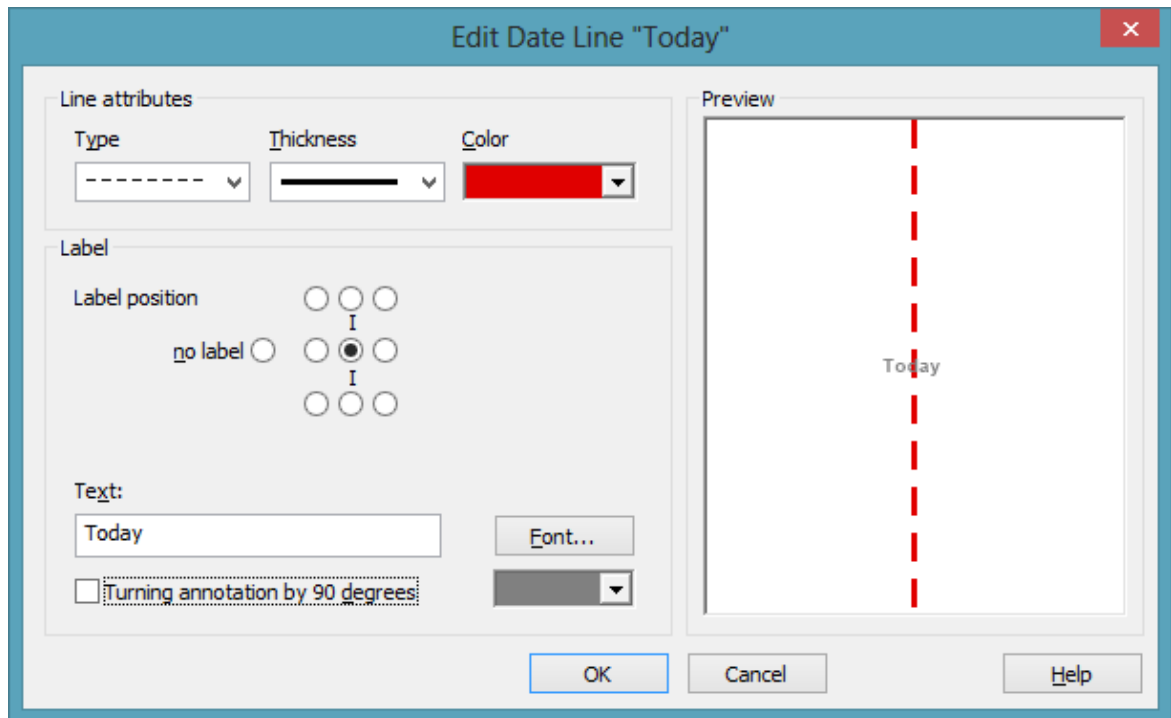


3.3 Date Lines

Date lines (vertical lines in the diagram) can highlight certain dates. Their attributes, such as the date, the line type, the priority and whether they are moveable and visible can be set in the corresponding "Specify..." and "Edit..." dialogs. You can get to them if you click on **Date lines** button on the **Objects** property page:



By clicking on  you can open the **Edit Date Line** dialog:

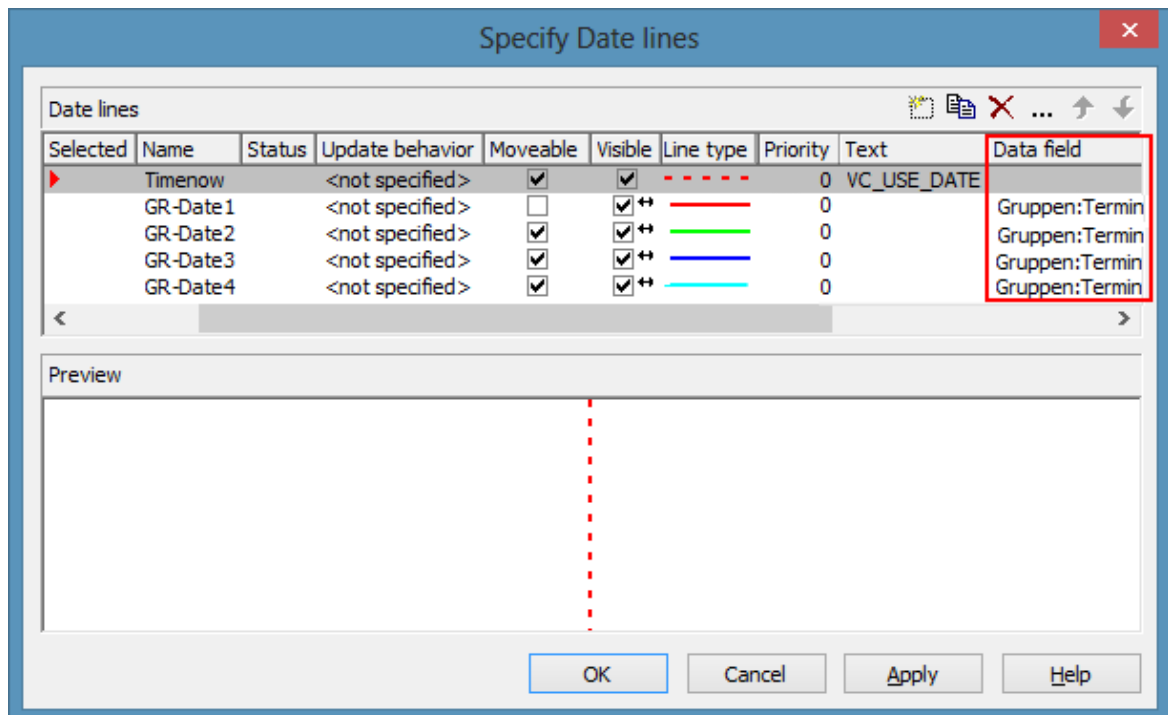


Individual, data-based date lines

Besides the fixed date, date lines can also use a date from a node or group record. This means that for each node or group record an individual date line as graphic copy can be created, using the properties (color etc.), except date, from the underlying date line of the DateLine Collection, date and position in the plan being individual, however. Such date lines are only drawn within the ribbons of nodes or groups by using **NodeLevelLayout** or **GroupLevelLayout**, resp., (see picture below: four date lines have been created and placed for three groups individually; four symbol layers of the activated group node use the same dates as the date lines).

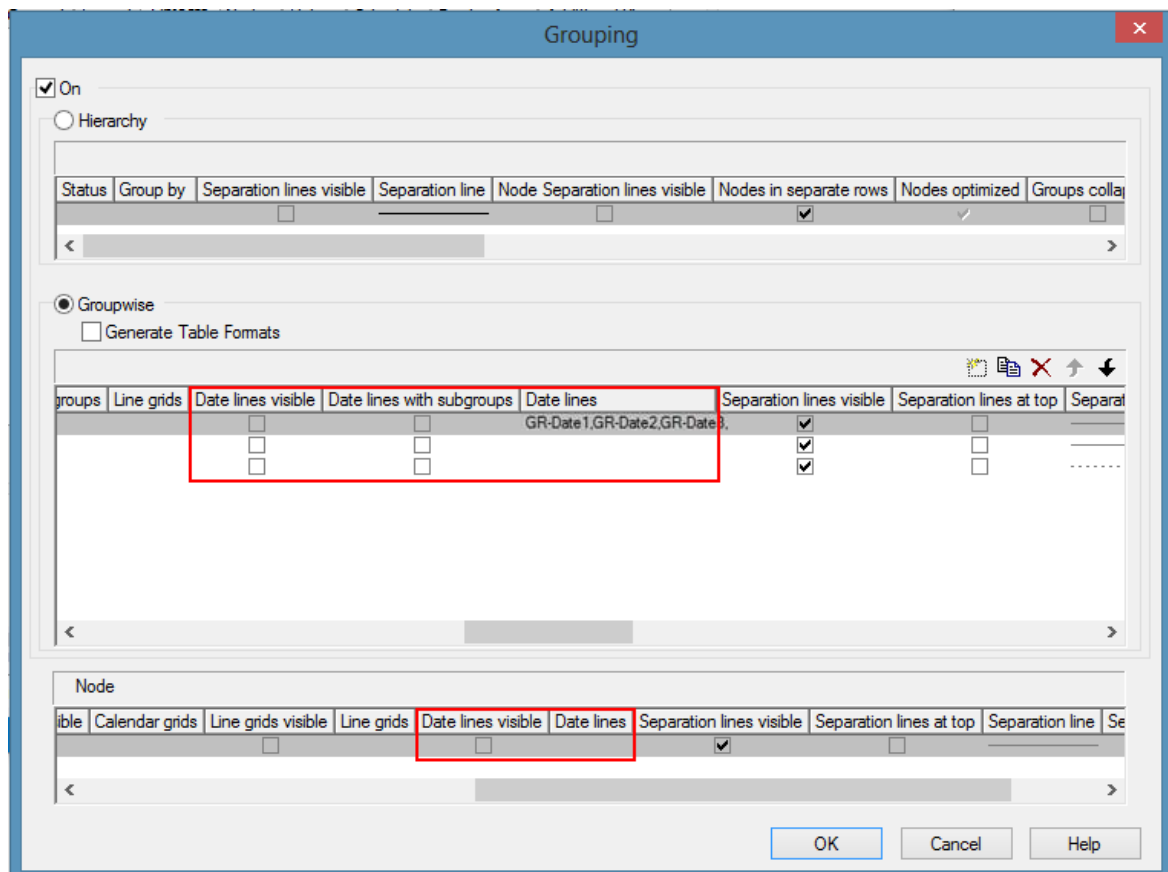


For this, a data field has to be specified for the date:



Note: When a data field has been individually specified, the date from the record has priority over the fixed date (**VcDateLine.Date**). When no date could be identified, e.g. because the data field is empty in the record, the date line has to be linked to a data record. This is done by the according settings in the **Grouping** dialog:

90 Important Concepts: Date Lines



The corresponding API properties:

VcGroupLevelLayout.DateLinesVisible

VcGroupLevelLayout.DateLinesWithChildGroups

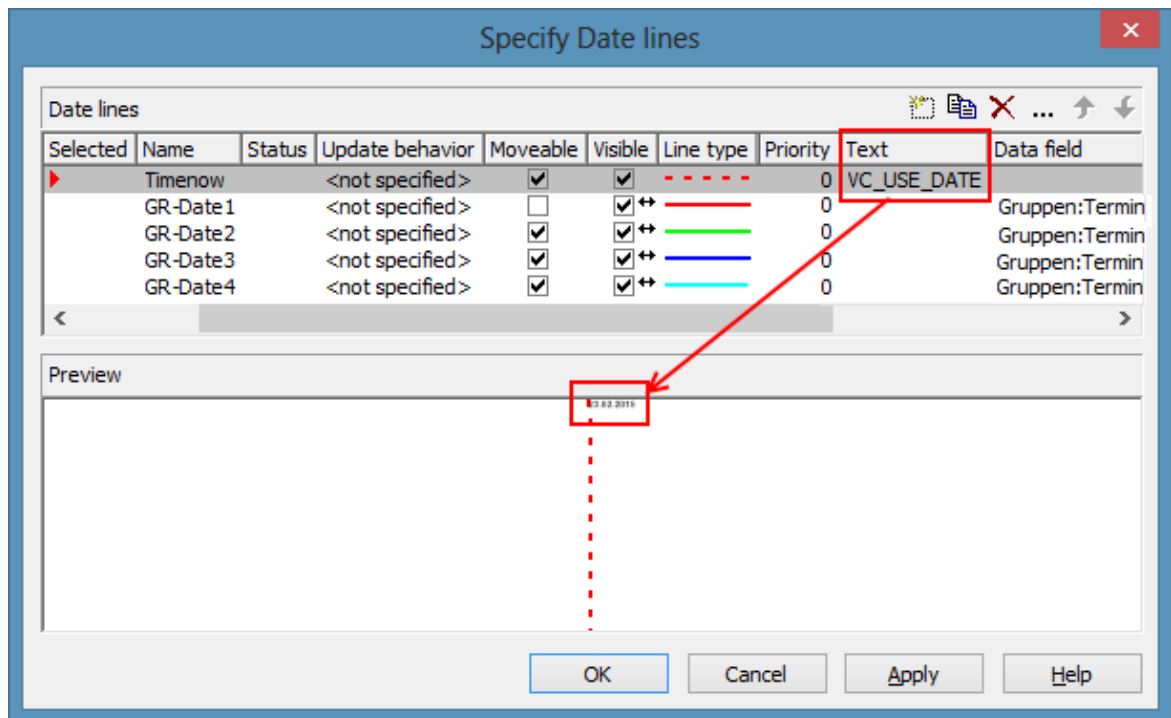
VcGroupLevelLayout.DateLineName

VcNodeLevelLayout.DateLinesVisible

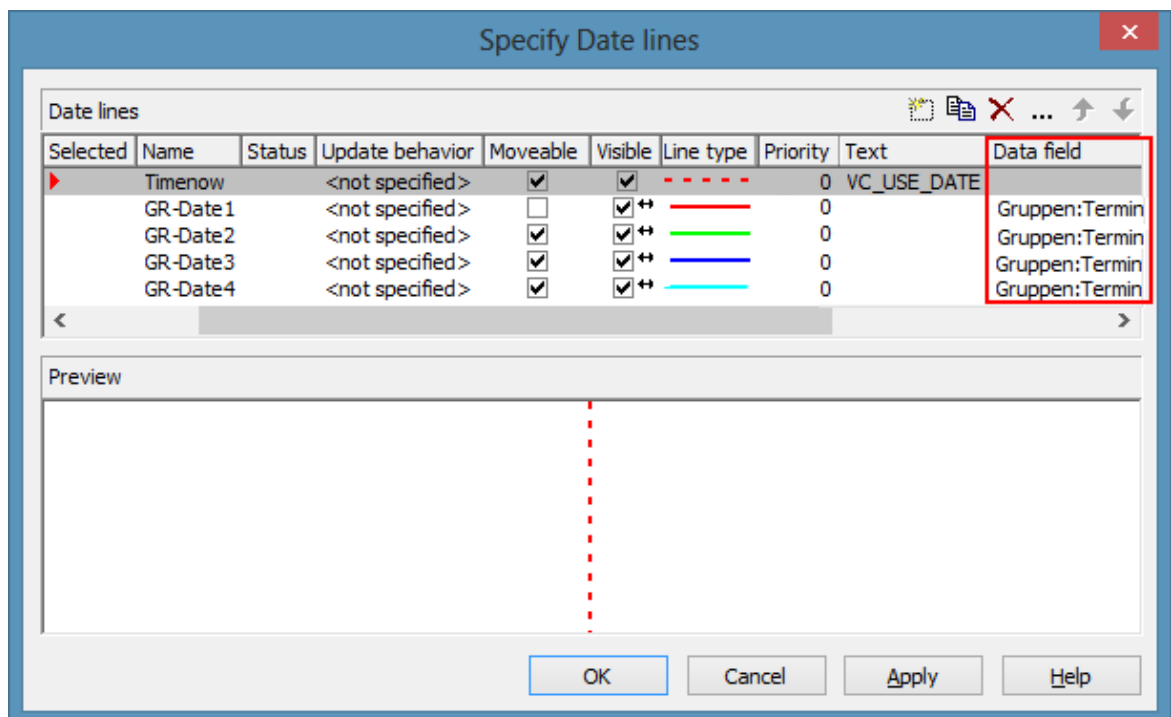
VcGroupLevelLayout.DateLineName

Labeling Date Lines

Date lines can be labeled. As a rule, this is done by a fixed text, Displaying the individual date might in some cases be wished for at all, but especially at individual date lines. The key word **VC_USE_DATE** manages to display the corresponding date at the specified place of the date line (**VcDateLine.LabelPosition**) in the specified date format (**VcGantt.DateOutputFormat**).



To make date lines individually visible the **Visible** option can be mapped and thus be set individually.



The corresponding API properties:

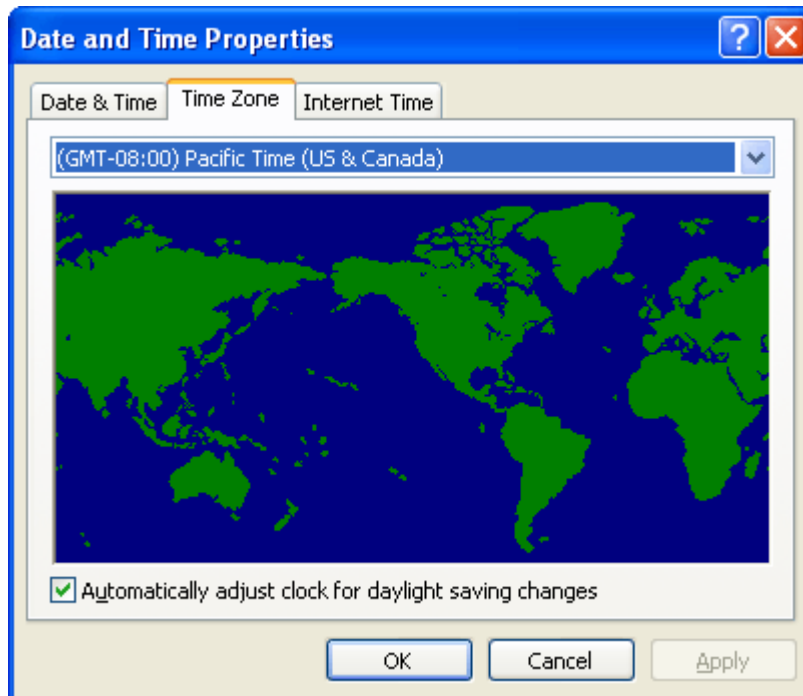
VcDateLine.VisibleDataFieldIndex

VcDateLine.VisibleMapName.

3.4 Dates and Daylight Saving Time

Dates in VARCHART components always refer to the time zone set in the system that the program is running on. It is not possible to set dates from different time zones; the dates have to be converted into dates of the time zone set to the system that VARCHART XGantt is running on before they are passed to the VARCHART component. The latter automatically refers to the information on the beginning and the end of daylight saving time which is present in the system.

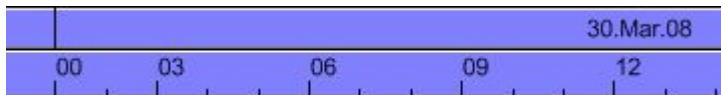
To make the switching times known to a VARCHART component, the check box in the time zone dialog **Automatically adjust clock for daylight saving changes** needs to be ticked, as shown in the picture. You can find the dialog in the Windows system by clicking on the button **Start**, then on the menu item **Control Panel**, then on the icon **Date and Time**.



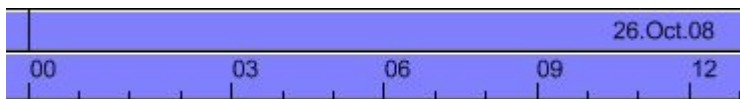
When switching to or back from daylight saving time, a VARCHART component uses the start date and the end date including hour, month and day of daylight saving time that usually are communicated by the system. This implies that the DST times of the years before and after the current year are extrapolated and true deviations probably existing of those years are ignored, since they are also unknown to the system. For example, a couple of years ago daylight saving time was prolonged for some weeks at the beginning and end. Since the system only knows the current rules, dates in those periods consequently will be interpreted in the wrong way.

At present, VARCHART components can only take into account a DST time offset of exactly one hour. Besides, the switch can only take place at full hour. Since the VARCHART components always receives and displays the date values of local time, at the beginning of the DST period there is an hour missing and at the end there are two hours of the same number. At present, the identical numbers are not discriminated when passed, returned or displayed.

The switching becomes visible in the time scale if its resolution is hours.



Switching between 0 and 3 o'clock in spring (1 hour missing)



Switching between 0 and 3 o'clock in spring (1 hour twice)

> New Default Date From Version 4.3 Onward

If in a VARCHART component a date is retrieved that does not exist, up to version 4.3 the date **31.12.1899 00:00:00** was returned. From version 4.3 onward, a different date **01.01.0001 00:00:00** will be returned.

In certain situations this can lead to an argument-out-of-range exception which you can intercept by treating the exception.

If within your application program, for example a date is handled by DateTimePicker controls of .NET, and if you try to display an “empty” date, up to version 4.3 the date **31.12.1899 00:00:00** was displayed. The new default though, which is **01.01.0001 00:00:00** cannot be displayed by using the default settings of the DateTimePicker, so it will throw an **ArgumentOutOfRangeException** exception.

Your program should react to this; in any case you should write some treatment to this exception, otherwise an untreated exception could occur and could entail an unexpected end of program.

3.5 Drag & Drop

Apart from moving or copying nodes within an instance of the VARCHART XGantt component, a user can also move or copy activities beyond the limits of an instance (source component) to a different instance (target component). This chapter introduces subjects that are important to the developer to program the latter type of interaction.

Whereas shifting a node within the same instance entails an alteration of the node's data, its dates do not change if the node is shifted between different instances (they certainly could by a subsequent shift within the target instance).

Shifting a node between different instances splits into two steps: leaving the source component and entering the target component. Each step requires a permission from the corresponding component.

VARCHART XGantt allows to move or copy several nodes by a single interaction. If a user presses the left mouse button while the cursor is on a node, internally an object of the type **System.Windows.Forms.DataObject** is generated and filled with the data of the node in CSV format (i.e. by text or by the data type **System.String**). After that, the event **VcDragStarting** is triggered immediately so that the application can control permitted actions (copy and/or move) by itself. By default, both actions are possible, depending on the status of the <Ctrl> key: by pressing it while releasing the mouse button, the object will be copied, otherwise it will be moved.

After this, the event **VcDragCompleting** is triggered to inform the application of the action taken (copy, move or cancellation) and to enable it to probably react.

Then, in the source component the events **Control.GiveFeedback** and **Control.QueryContinueDrag** are triggered. In the target component the events **Control.DragEnter**, **Control.DragOver** and **Control.DragLeave** are triggered.

For further information about the .NET drag&drop routines please refer to the description of the .NET framework. In addition, five more properties exist that influence the behavior of drag&drop:

> **Control.AllowDrop**

This Boolean property of the base class **Control** allows to set whether objects that were dragged onto the control can be dropped. The property applies only to objects from the outside; objects dragged within the VARCHART control are not affected (i.e., they can always be dropped).

> **VcGantt.LeavingControlWhileDraggingAllowed**

This Boolean property of the VcGantt object allows to set whether nodes can be dragged beyond the limits of the source control. This allows to move or copy nodes between two different VARCHART controls, to different controls of the same application or even to controls of different applications.

> **VcGantt.NodeCreationAtDroppingEnabled**

This Boolean property of the VcGantt object allows to set whether the target component automatically should generate a node after an object was dropped on it.

> **VcGantt.PhantomDrawingWhileDraggingEnabled**

This Boolean property lets you set to the target component whether the default phantom of the VARCHART component should be generated.

> **VcGantt.InbuiltMouseCursorWhileDraggingEnabled**

This Boolean property lets you set to the target component whether the mouse cursor typical of the VARCHART component should be displayed. If it is not displayed, the drag&drop mouse cursor (arrow and a little square or prohibitory sign) will be displayed, or even a cursor specific of the application.

3.6 Dragging Tools

Gantt charts enable the planner to easily re-plan orders, tasks or resources by shifting them back and forth. However, positioning a node at a certain point of the timeline or directly after another node can be tricky because a certain spot in the Gantt has to be exactly hit by mouse.

Besides, in many Gantt charts, multi-level groups are used. In large plans dragging a node from one group or its subgroup to another one by mouse can at times get a bit inconvenient and confusing if the target group is located quite far away.

Snap Tools: Support for horizontal dragging

Many dragging applications or design tools already offer the so-called snap-grids as help for exactly positioning objects by means of a predefined grid, usually pixel-spaced. VARCHART XGantt now offers a similar functionality. The moved objects are not adjusted to a fixed grid but to other objects in the graphic, these objects thus defining a snap grid with irregular distances.

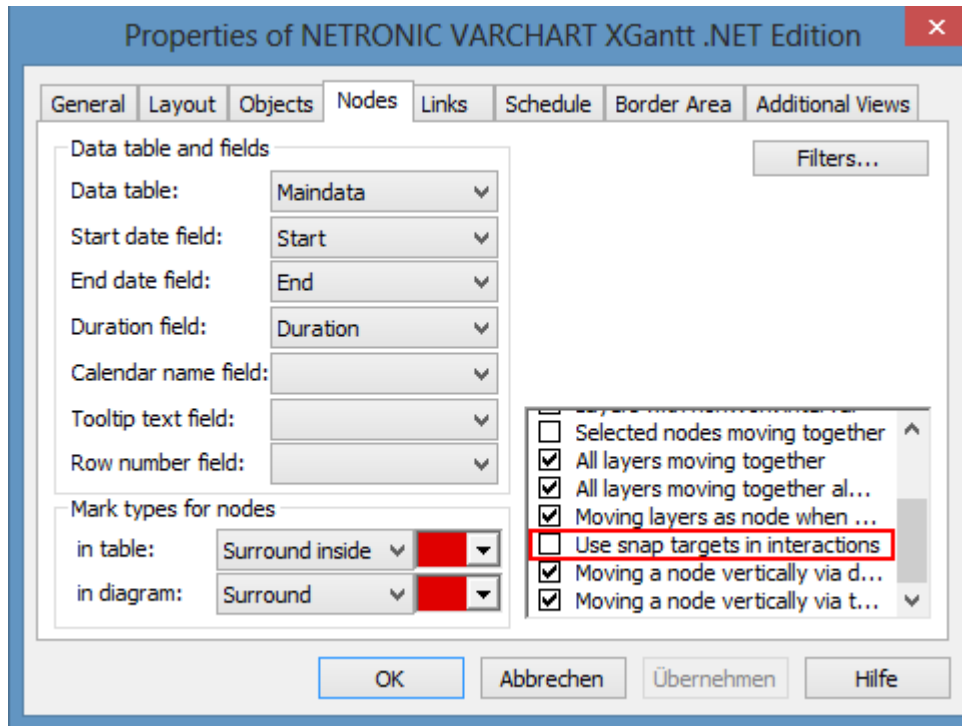
Nodes (or their layers), date lines, line grids and calendar grids allow to define so-called snap targets. That means that these objects define certain places at themselves serving as targets of a snap action of other objects. When moving a node horizontally or modifying the size of a node or a layer, start or end date of this node or layer will be chronologically adjusted to the defined snap tools of the other objects. The start or end date will move towards the snap target within 5 pixels next to it thus taking over the exact date of the target.

Special behaviors have been defined for each node layout (ungrouped, grouped, hierarchical arrangement; given that the according objects define snap tools):

- All node layouts: the layer-to-be-moved is adjusted to date lines, line grids and calendar grids.
- Ungrouped layout: The layer-to-be moved is adjusted to the layers of all nodes.
- Grouped layout: The layer-to-be-moved is adjusted to the layers of the nodes of one group (without subgroups). If the group is changed during the interaction, the layer will be adjusted to the objects of the new group.
- Hierarchical arrangement: The-layer-to-be-moved will be adjusted to the layers of the nodes of the same branch (with sub-branches). If the branch

is changed during the interaction, the layer will be adjusted to the objects of the new branch.

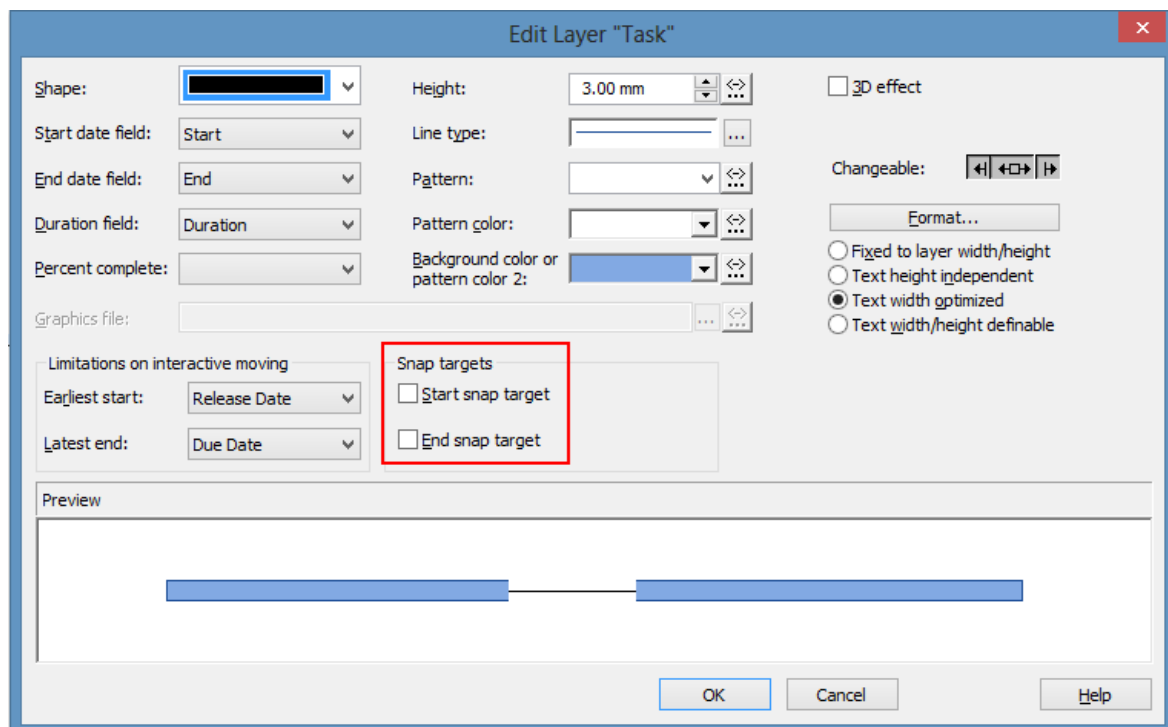
For the snap tools to take effect, they have to be enabled on the **Nodes** property page



API call: **vcGantt.UseSnapTargetsInInteractions = true/false**

Layers can be defined as snap targets in the Edit Layer dialog. Ticking the checkboxes Start snap target and End snap target sets the layer's position (i.e. its dates) as snap targets for dragging a node or layer.

98 Important Concepts: Dragging Tools

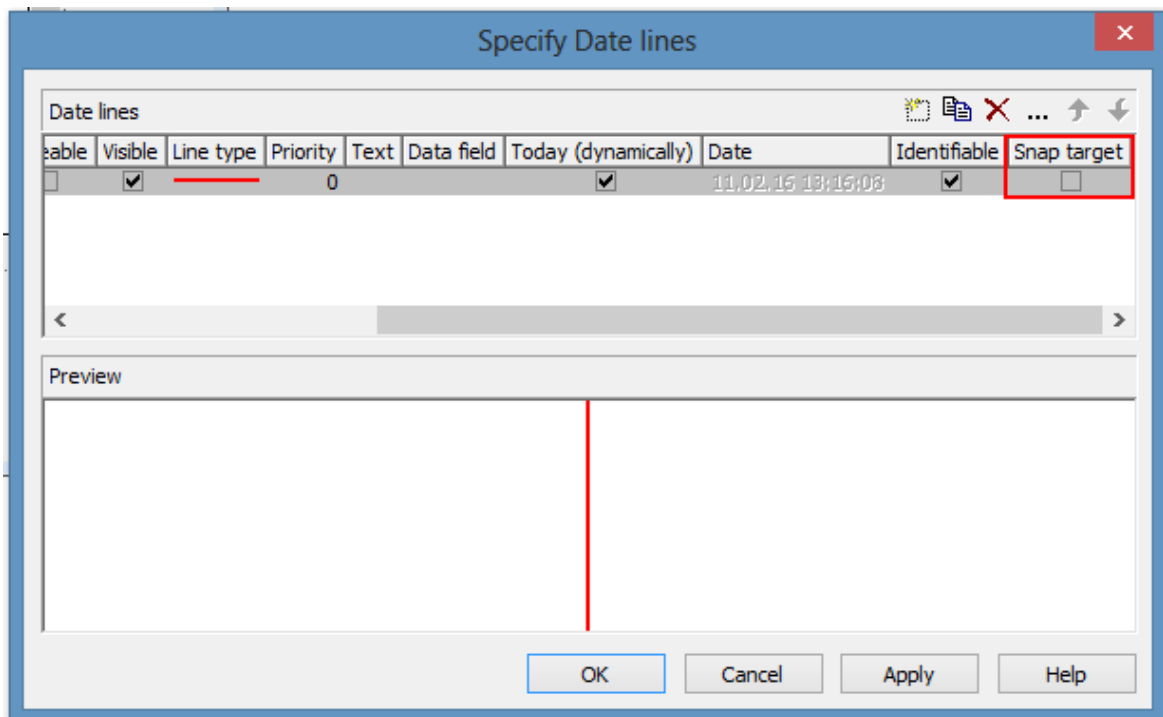


API calls:

VcLayer.StartSnapTarget = true/false

VcLayer.EndSnapTarget = true/false

Date lines can be defined as snap targets in the Specify Date Lines dialog. Ticking the checkbox Snap target sets the date line's position (i.e. its dates) as snap target for dragging a node or layer.



API call: **VcDateLine.SnapTarget = true/false**

> **Snap target LINE GRIDS/CALENDAR GRIDS**

Line grids and calendar grids can be defined as snap targets at two different places:

- In the Edit time scale section for not individual objects
- Below the Grouping dialog for individual, group- or node-related objects.

Ticking the according checkboxes in the Edit time scale section dialog sets the related objects' position (i.e. their dates) as snap targets for dragging a node or layer.

100 Important Concepts: Dragging Tools

Width per time unit: ☐ Collapse workfree periods

Time ribbons with 3D effect: ☐

Index	Type	Position	Minimum height	Major ticks	Minor ticks	Pattern	Font	Date format	Unit separation	Tick position	Tick color	Alignment
0	Months	Top	5.2 mm	1	1		8 pt, ...	TM YYYY	Full line	Below		Center
1	Days	Top	5.2 mm	1	1		8 pt, ...	DD	Full line	Below		Center

Line grids

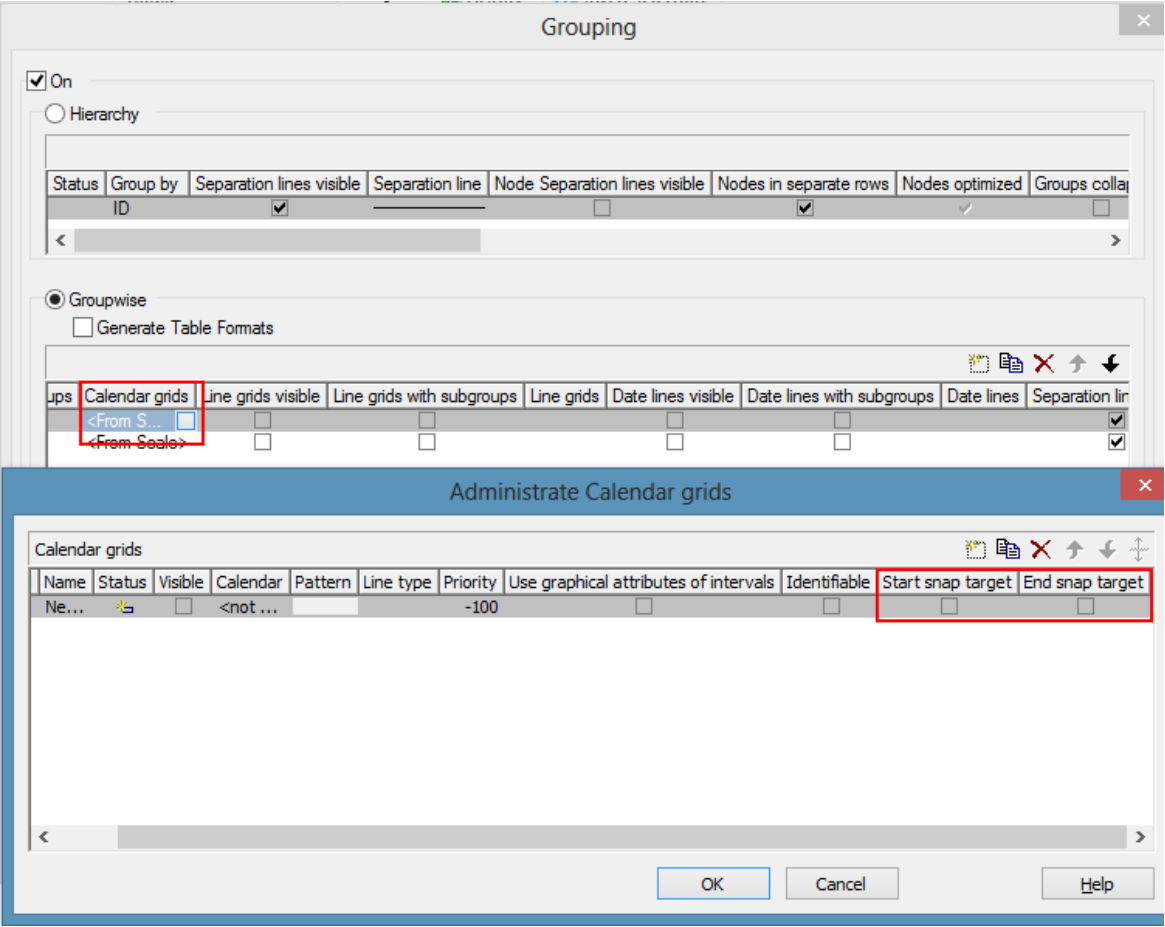
Interval	Line type	Priority	Use reference date	Reference date	Adjust to reference date	Observe DST	Snap target	Line format	Turn
xs 1 Weeks	-----	-20	<input type="checkbox"/>		<input type="checkbox"/>	not specified	<input type="checkbox"/>	<not specified>	<input type="checkbox"/>
hs 1 Months	-----	-10	<input type="checkbox"/>		<input type="checkbox"/>	not specified	<input type="checkbox"/>	<not specified>	<input type="checkbox"/>

Calendar grids

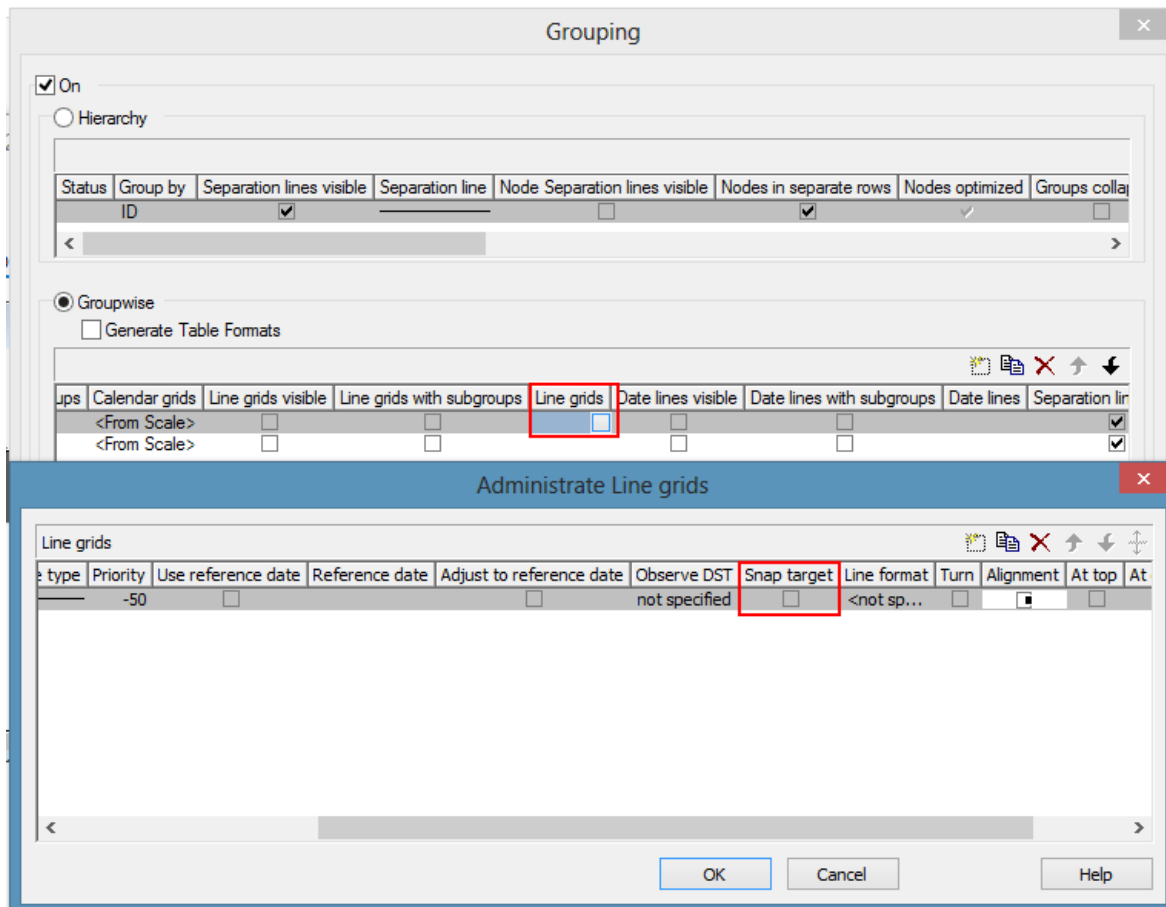
Index	Visible	Calendar	Pattern	Line type	Priority	Use graphical attributes of intervals	Identifiable	Start snap target	End snap target
0	<input checked="" type="checkbox"/>	<not specified>			-25	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

OK Cancel Help

In the **Grouping** dialog you can access the dialogs **Administrate Calendar Grids** and **Administrate Line Grids**, where ticking the according checkboxes sets the related objects' position (i.e. their dates) as snap targets for dragging a node or layer.



102 Important Concepts: Dragging Tools



API calls:

VcDateLineGrid.SnapTarget = true/false

VcCalendarGrid.StartSnapTarget = true/false

VcCalendarGrid.StartSnapTarget = true/false

Please note: Since it makes no sense to mix the snap targets of all objects (i.e. the objects from several ribbons) when moving several nodes, snap targets of individual objects are only taken into account if a single node is moved. A separate snapping of a node to the snap target of the ribbon it is situated in is not provided for.

> Moving a node by arrow keys

Nodes can not only be moved interactively by mouse but also by the mouse keys on the keyboard. To do this, the following setting is needed:

vcGantt1.ArrowKeyMode = VcArrowKeyMode.vcResizeOrMoveNode

The value **vcNodeJumpToSnapTarget** was added to the enumeration **VcArrowKeyMode**. If this value is set, pressing CTRL + left or right arrow key causes a marked node to snap to the next or the last snap target, this

being s a cyclical operation: If the end is reached, everything starts at the beginning again.

Auto collapse/expand: Support for vertical dragging

Everybody has already moved files in the Windows explorer and knows the automatical expanding of the folder structure: You move the file onto a collapsed folder, pause the mouse shortly, the folder is opened and you can move further until you have reached the desired folder.

> Behavior in older versions

Up to now, when moving a node vertically to another group in VARCHART XGantt, searching for the target group could take quite a bit of time, if the chart had many nodes in many expanded groups. In most cases, automatic vertical scrolling was needed to reach the target group, this sometimes being tedious and therefore uncomfortable.

> New: Easy orientation and fast vertical dragging

The new functionality considerably shortens the search for the target group. The combination and setting options being quite manifold, we'd like to confine ourselves to introducing one possible configuration here.

Example: Collapse all groups except the current one

One possible configuration of VARCHART XGantt might be that when moving a node, all groups but the one having just been touched get collapsed. The status of this group will be maintained, in case the node is to be moved within the same group only. By collapsing the other groups, the vertical extension of the plan is reduced to a fraction of its original size, thus allowing to show considerably more groups than before and ideally, the target group will be already visible by now. If not, VARCHART XGantt can automatically scroll over the collapsed groups so that the target group can be found much faster than before. On reaching the target group, one pauses a moment, the target group is expanded and the movement can go on. The group having been touched before gets collapsed so that the plan size remains minimized. The dragging goes on, perhaps to another group that is expanded, the group having been expanded before being collapsed again etc. until reaching the target. On releasing the node in the target group, the interaction is finished and, if desired, VARCHART XGantt can restore the original condition, scrolling to the new position of the moved node.

> **Many combination options**

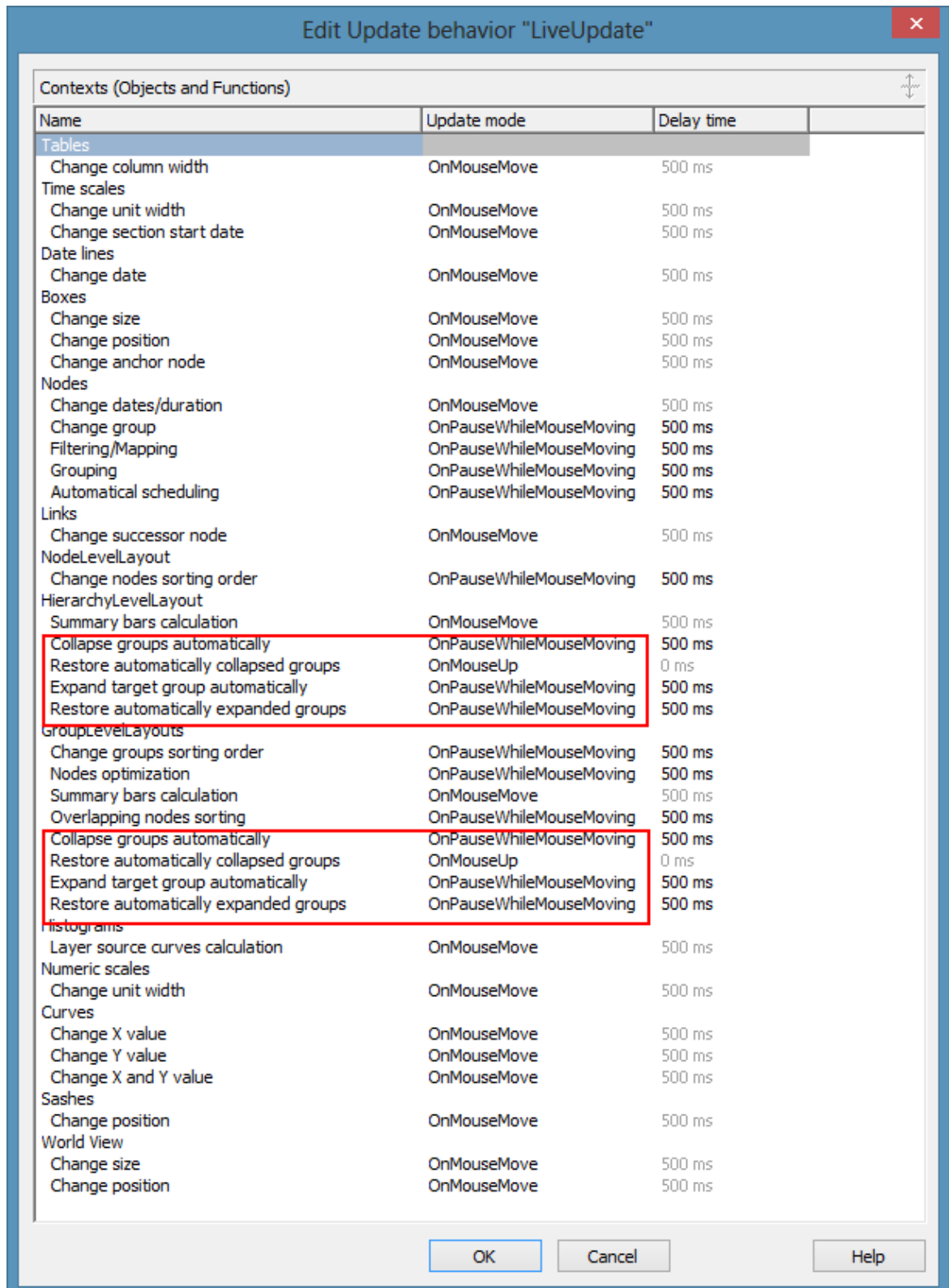
This was only one example of the new functionality. There are further options available for:

- Automatic collapsing of groups
- Automatic expanding of groups
- Automatic restoring of automatically collapsed or expanded groups, an update behavior allowing for a precise temporal control of this option.

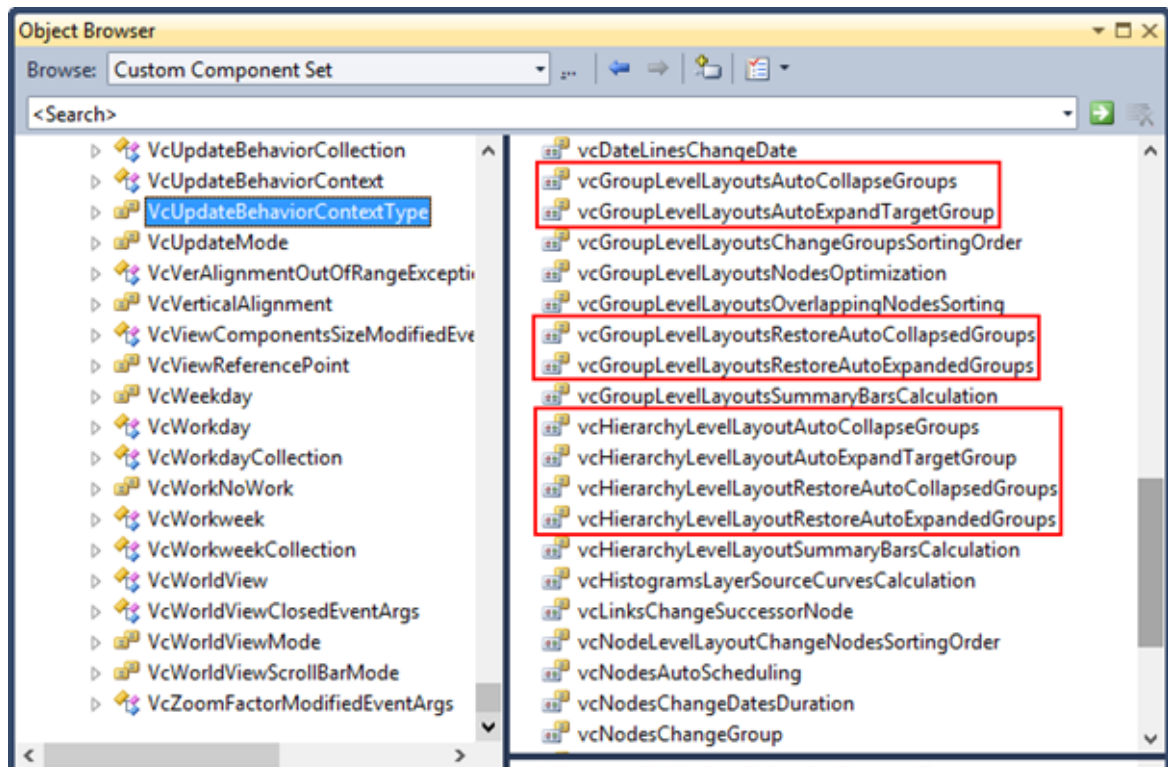
These settings can be made per grouping level and also for the hierarchical arrangement of the nodes, allowing for very detailed dragging operations.

> **New properties and API calls**

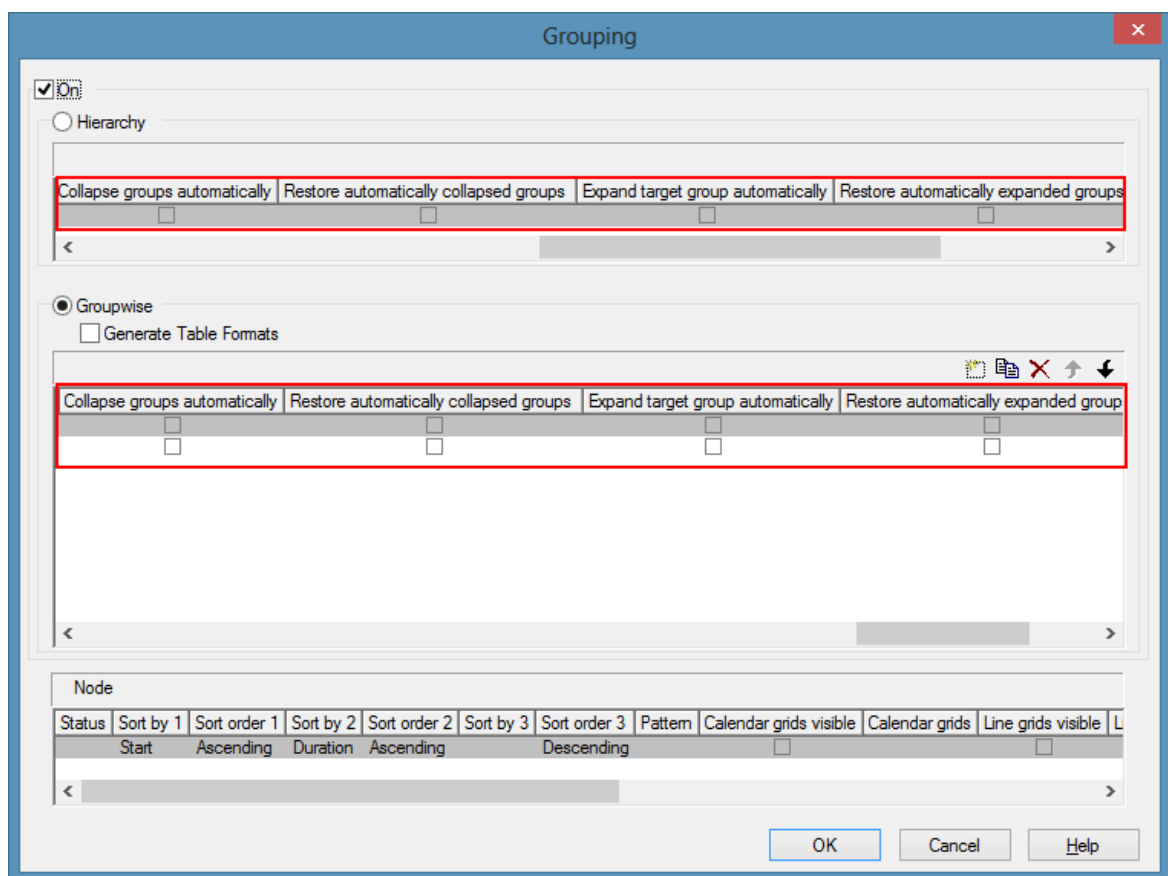
The **Edit Update behavior** dialog offers eight related contexts, four each in Grouping Line Layouts and Hierarchy Layout:



The enumeration `VcUpdateBehaviorContextType` has also got 8 new values so that the new contexts can also be set at runtime.



The functionalities that are activated by this contexts by way of timer can be enabled or disabled in the **Grouping** dialog.



API calls:

VcGroupLevelLayout.AutoCollapseGroups = **true/false**
VcGroupLevelLayout.AutoExpandTargetGroup = **true/false**
VcGroupLevelLayout.RestoreAutoCollapsedGroups = **true/false**
VcGroupLevelLayout.RestoreAutoExpandedGroups =
true/false

VcHierarchyLevelLayout.AutoCollapseGroups= **true/false**
VcHierarchyLevelLayout.AutoExpandTargetGroup = **true/false**
VcHierarchyLevelLayout.RestoreAutoCollapsedGroups =
true/false
VcHierarchyLevelLayout.RestoreAutoExpandedGroups =
true/false

3.7 Events

Events are the elements that pass information on the user's interactions with the VARCHART control to the application. Each time a user interacts with the VARCHART control, for example by modifying data or by clicking on somewhere in the control, a corresponding event is invoked. You can react to these events in the program code of your application.

In all programming environments, functions which already contain the parameters provided by the control are supplied for the various events. Each event is described in detail by the API Reference.

Note: By means of the events, via the **returnStatus** parameter you can deactivate all context menus offered in VARCHART control (and replace them by your own, if you want) plus you can control all interactions and revoke them where required.

> Return Status

The below table shows the return status values of VARCHART events:

Constant	value	description
vcRetStatDefault	2	default value
vcRetStatFalse	0	revoking the action
vcRetStatNoPopup	4	revoking the popup menu

3.8 Filters

A filter consists of conditions that are to be fulfilled by layers, histogram curves, links or table formats. Filters let you select layers, curves, links or table formats that fulfil the criteria defined, e.g. in order to highlight them in the diagram.

When applying a filter, the data of the record is compared to the criteria of the filter. Those layers, curves, links or table formats that fulfil the filter criteria will be selected.

For example, you can define a filter that selects "All activities starting after January 2012".

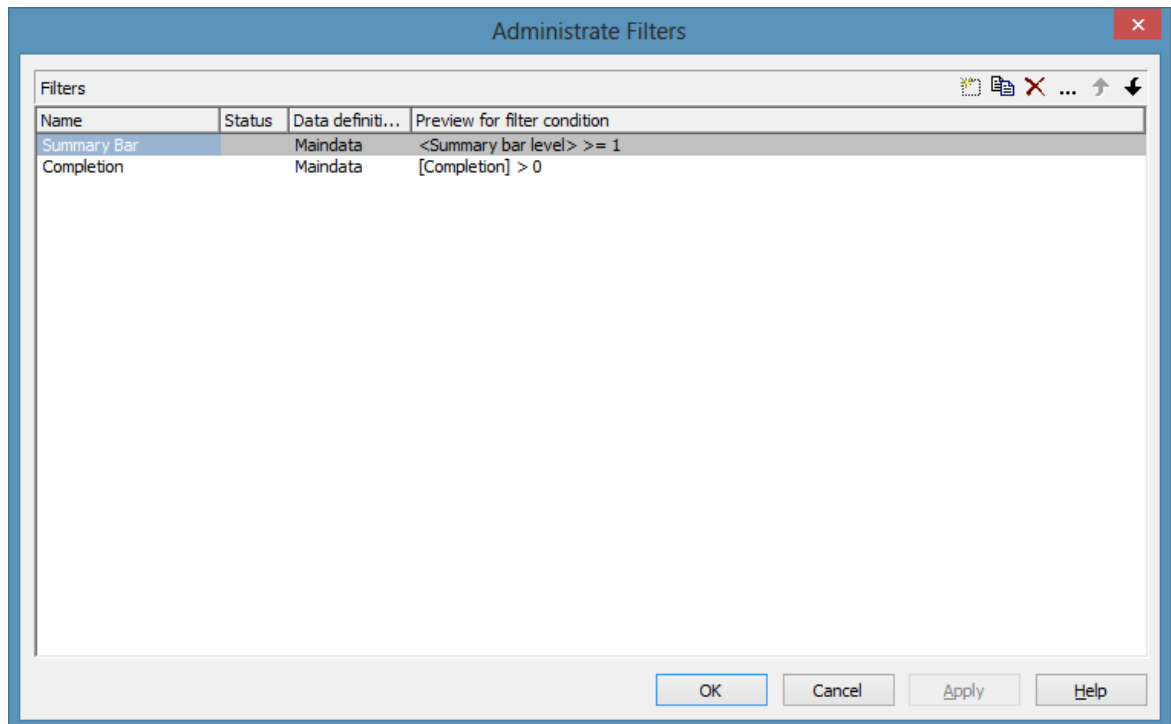
Filters can only be handled in design mode.

You can reach the **Administrate Filters** dialog box:

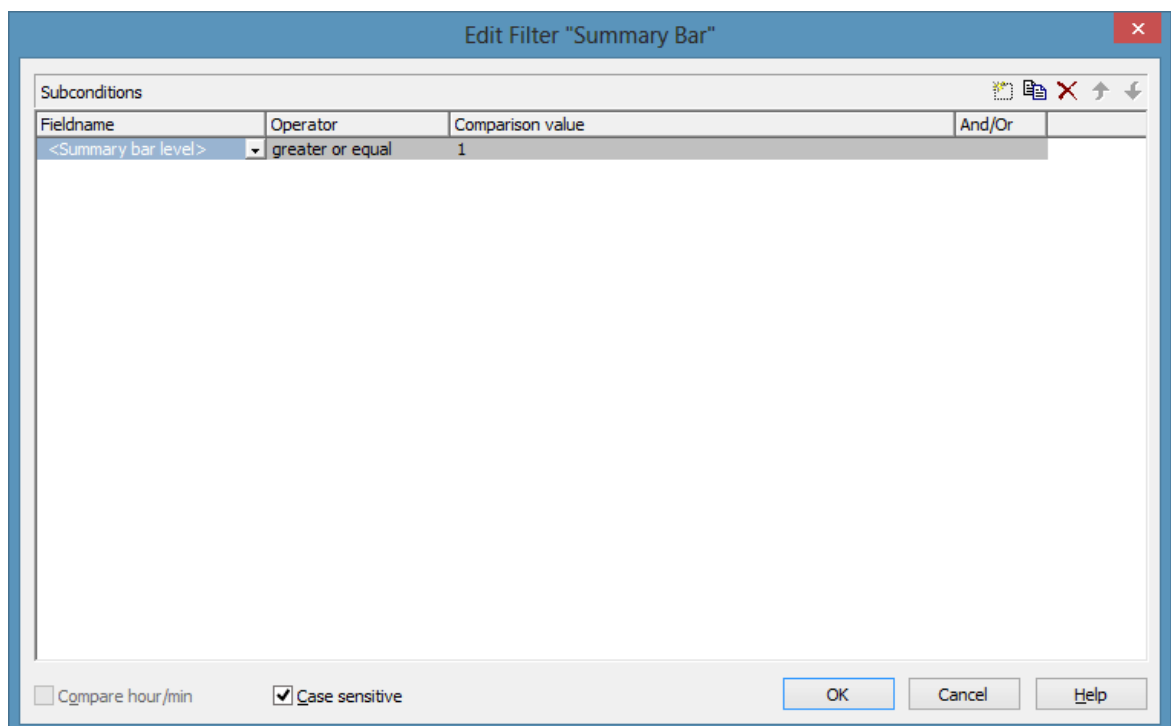
- via the **Objects** property page
- for layers: via the **Specify Bar Appearance** dialog box
- for table formats: via the **Edit Table** dialog box
- for links: via the **Filter** button of the **Link** property page
- for histogram curves: via the **Filter** combo box of the **Edit Histogram** dialog
- for nodes: via the **Filter** button of the **Nodes** property page.

Use the **Administrate Filters** dialog box to rename, create, copy, delete or edit filters.

110 Important Concepts: Filters



To edit a filter, please click on the **Edit filter** button of the **Administrate Filters** dialog box. Then the **Edit Filter** dialog box will open.



3.9 Graphics Formats

VARCHART supports the below graphics formats, which is important to exporting charts, affecting mainly the calls **VcGantt1.ShowGraphics-ExportDialog** and **VcGantt1.ExportGraphics**.

The XGantt control supports both the import of graphics files e.g. for displaying in nodes or in boxes and the export of complete charts to graphics files. There is a connection between the chosen (supported) graphics format and the graphic's display quality in the control (after the import) or in an external viewer program (after the export). Please find below a description of the advantages and restrictions of the individual graphics formats. Basically there are two different types:

Vector graphics formats store single geometrical figures such as lines, ellipses or rectangles as descriptions of the figure with corresponding parameters as start coordinates, dimension and color. Thus they are resolution-independent and lines are still displayed precisely, regardless of the zoom level. There is just one restriction concerning the size of the available coordinate space, especially with the WMF format. In general, the vector graphics formats' great advantage lies in their resolution independence and also often in the resulting file size. Unfortunately a platform-independent, standardized format has not established itself.

Bitmap graphics formats store pixels together with their color in a preset dimension. If the graphics are heavily zoomed in they automatically get "pixelly". To limit the file size, bitmap graphics are often compressed lossless or lossy even. A loss, however, can only be accepted with photos, not with diagrams. The only advantage that the bitmap graphics formats offer is the fact that they have become widely accepted via digital cameras and the internet and are widespread platform-independent.

> WMF (Windows Metafile Format)

This vector graphics format has been in existence since Windows 3.0. It internally consists of command data sets that correspond to the GDI commands of the Windows API. By them, the GDI commands can be persisted to all intents and purposes. Nevertheless, this format was incomplete already when it was developed. It had and today still has a limited coordinate space. Beside, it lacks clipping, transforming coordinates and filling complex polygons. The problem of the missing option to transform the "real" coordinates into inches and centimeters was encountered by the Aldus company already at an early stage. They developed the "Aldus Placeable Header" which for long has been recognized and used by virtually all

programs that display and use WMF files, except for the Windows API itself, which up to now is unable to generate or process the header, although it is mentioned and explained in the Microsoft documentation.

When Microsoft released Windows NT and 95, the WMF format became dispensable and its successor called EMF entered the market. Still, WMF is quite popular up to now, especially with ClipArt graphics that do not require the extended options of the successor format. The innovations of Windows 95 and NT have not been not transferred to the format, it has remained unchanged since.

In WMF, a comment data set is available which can be used to place EMF commands. If a display program discovers those kinds of comments, i.e. if it can display EMF files, it automatically will discard the WMF command data sets and will display the EMF command data sets instead. Thus a single file can contain a WMF graphics as well as an EMF graphics. Presumably, this was implemented for reasons of compatibility, but it inflates the file size considerably.

For the description of the format please see:

<http://msdn.microsoft.com/en-us/library/cc215212.aspx>

On the limitations of the format see:

<http://support.microsoft.com/kb/81497/en-us>.

> **EMF (Enhanced Metafile Format)**

This vector graphics format was introduced simultaneously with the 32bit operation systems Windows NT and 95. It suspends the limitations imposed by the WMF format and internally consists of graphics commands that correspond to the GDI32 commands of the Windows API. The coordinates' space is 32 bits large, transformation and clipping are supported. The commands of masking and alpha-blending equipped blitting of storage bitmaps added to GDI32 later on are not supported though.

In spite of its advantages that it features compared to WMF, the format has remained largely unknown, although all display programs and Office packages can handle EMF.

A disadvantage when using GDI+ is that some of the new GDI+ graphical features such as color gradients and transparencies are not fully supported. In addition, when exporting the chart into an EMF file, discontinuous lines (for example dashed) are stored as a set of short, continued lines, which on one hand increases storage demand and on the other hand consumes more time when the file is loaded.

EMF also offers a comment data set that can be used to place EMF+ commands. If a display program discovers those kinds of comments, i.e. if it can display EMF+ files, it automatically will discard the EMF command data sets and will display the EMF+ command data sets instead. Thus a single file can contain a EMF graphics as well as an EMF+ graphics. Presumably, this was implemented for reasons of compatibility, but it inflates the file size considerably.

By the way, if required, printing jobs in Windows internally are cached as EMF data streams and passed to the printer driver.

For the format description please see:

<http://msdn.microsoft.com/en-us/library/cc204166.aspx>

> **EMF+ (Enhanced Metafile Format)**

Although the name suggests this format to be an extension of EMF, it is a vector graphics format of its own which was introduced simultaneously with the GDI+ Windows API. Internally, it consists of graphics command data sets that correspond to the GDI+ commands. By the way, GDI+ is not an extension of the GDI API, but a graphics library of its own. In addition to EMF also transparencies and color gradients are completely supported.

Up to now the format has remained quite unknown and quite often is not supported by the common display programs, except by Microsoft Office from 2003 onward. Microsoft has published the structure of the EMF+ format only in 2003.

For the format description please see:

<http://msdn.microsoft.com/en-us/library/cc204376.aspx>

> **GIF (Graphics Interchange Format)**

This bitmap format was developed by CompuServe for a lossless, compressed storage of graphics files before the World Wide Web came into existence. It can only display 256 colors simultaneously and is therefore unable to store today's graphics files reasonably. This format is only supported for reasons of compatibility.

The subformat "Animated GIF" is not supported at all.

> **JPEG (Joint Photographic Experts Group)**

This bitmap format was developed by the JPEG for compressed storage of photographs, accepting loss. Storing charts and diagrams requires a precise

storage of lines, so using this format does not make much sense. This format is only supported for reasons of compatibility.

> **BMP (Windows Bitmap)**

This bitmap format was developed by Microsoft for a lossless, uncompressed storage of graphics files. Internally, the format is used directly in the memory of the Windows API GDI. A restraint is given by this format not supporting the alpha channel, so merely 24 bits per pixel can be stored. Due to its high memory demand this format should be abandoned. This format is only supported for reasons of compatibility.

> **TIFF (Tagged Image File Format)**

This bitmap format was developed by Aldus (merged into ADOBE) for a lossless, uncompressed storage of graphics files. Graphics files can be stored with or without loss. The format has not been enhanced for quite some time. This format is only supported for reasons of compatibility.

> **PNG (Portable Network Graphics)**

This bitmap format was developed by the World Wide Web Consortium (W3C) for a lossless, compressed storage of graphics files to replace the copyright-afflicted and limited GIF format. PNG is brilliantly qualified to store VARCHART charts; transparent elements are actually drawn as such. It is universally used by virtually every display program and internet browser. The format itself is free of copyrights and completely documented.

From version 4.2 onward the free library **libpng** is used, in order to set a resolution and thus store bitmaps of any size. It has to be taken into account though that very large PNG files may cause problems when loaded, since usually PNG files get completely unpacked in the memory and then are displayed.

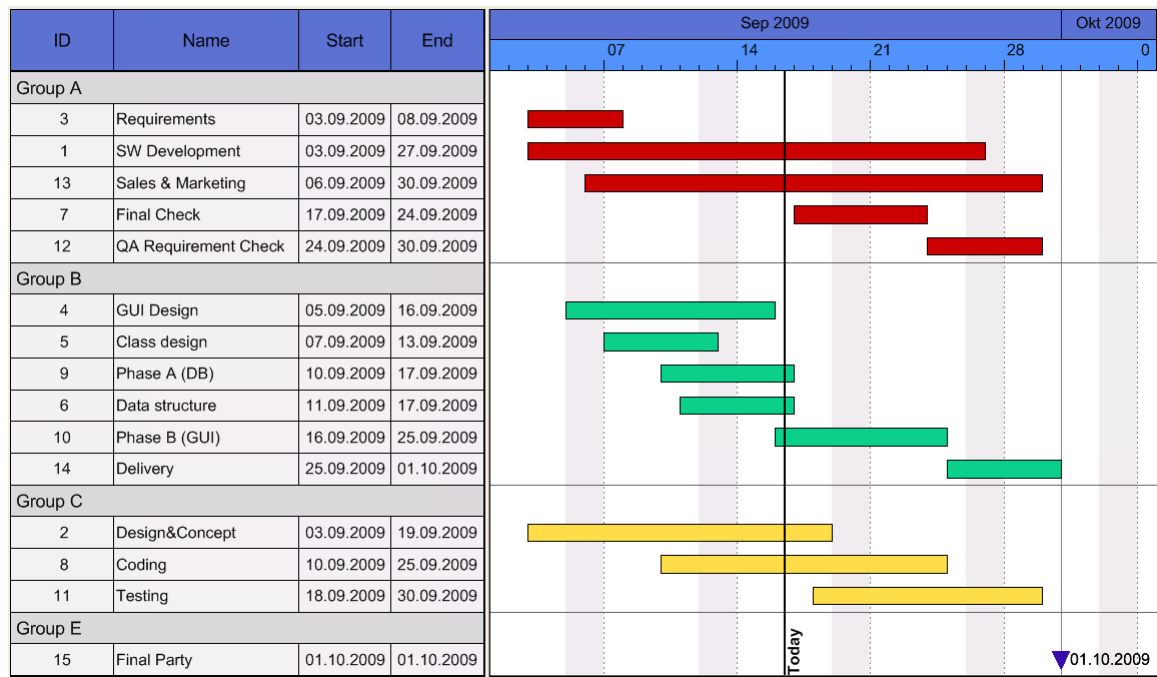
For the format description please see:

<http://www.libpng.org/pub/png/spec/1.1/PNG-Contents.html>

3.10 Grouping

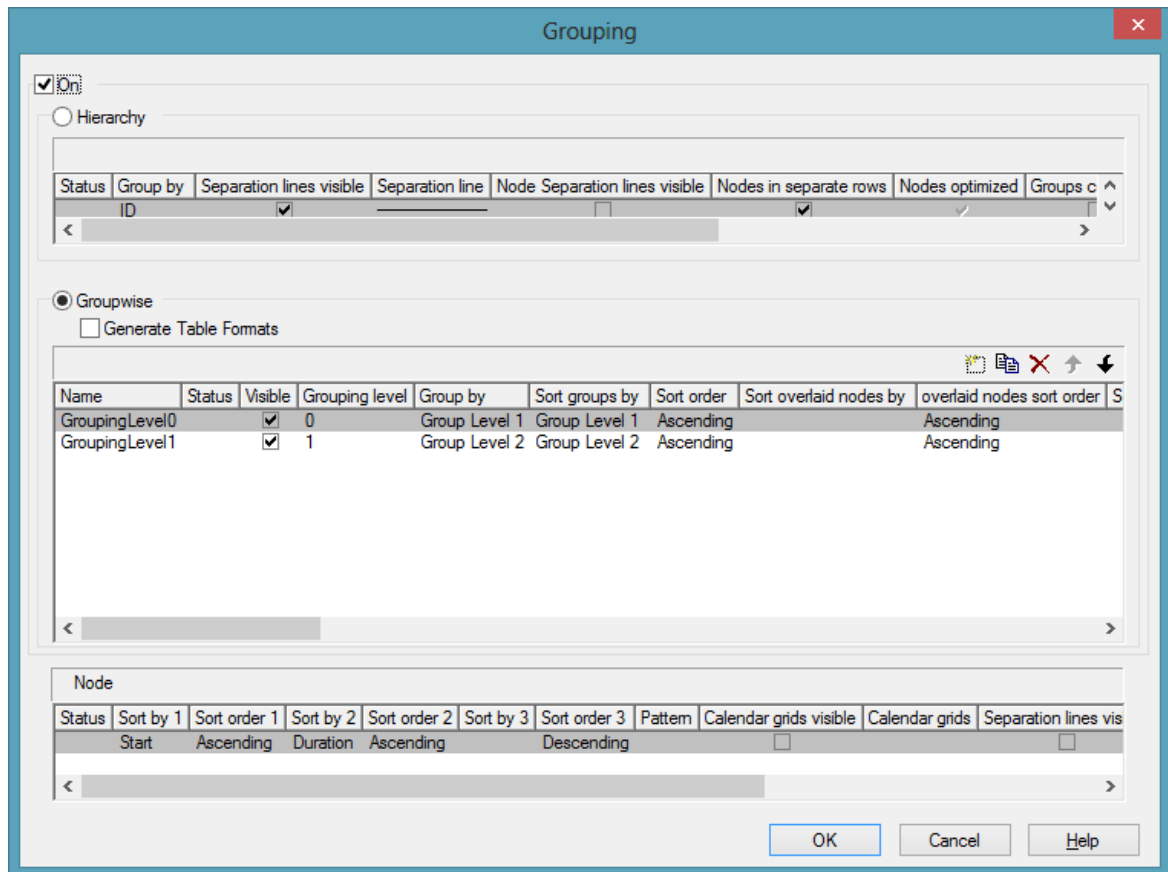
It often is necessary to split activities into groups and then visually emphasize the groups in your diagram. For example, activities are frequently grouped by project phases (e.g. planning, construction, manufacturing, etc.) or by departments (Construction Dept., Accounts Dept., etc.).

A grouped diagram could look something like this:



Groups are formed by a value, that all members of a group have in common. Nodes that show the same entry in their grouping data field belong to the same group. The grouping field and all other grouping criteria can be set in the corresponding dialog which you can open by clicking the **Grouping** button on the **Objects** property page.

116 Important Concepts: Grouping



Activities that have the same value in the **Group by** data field will be allocated to the same group.

In the diagram, an extra row is displayed above a group that contains the group title. The appearance of the group title in the table can be individually defined in the **Edit Table Format** dialog box depending on whether the groups are expanded or collapsed (table formats **Subtitle** and **Collapsed**), e.g. by using different colors or data fields.

The small plus or minus sign next to the group headings indicates whether the associated group is collapsed or expanded. By clicking on the sign, you can switch from the collapsed state to the expanded one and vice versa. To enable the feature, the **Modifications allowed** check box in the **Grouping** dialog has to be ticked.

You can use the **Sort groups by** and the **Sort order** options to set the order of the groups.

More options can be selected for groups:

- whether **table formats** are to be generated
- a **pattern** for the title row of the group (only in the diagram)
- display and style of **calendar** and **line grids**

- whether all activities of a group should be displayed in a single row or not (switching on/off the option **Nodes in separate rows**) and, if so, whether the node layout should be optimized automatically (**Optimized**)
- whether the groups should be collapsed when starting the program (**Groups collapsed**)
- display and style of **Separation lines**
- whether the collapse/expand function (**Modifications Allowed**) should be available to the user
- whether summary bars are to be displayed (**Summary Bars**)
- whether **Group nodes** are to be displayed
- whether the **order of groups** can be changed by drag interactions in the diagram and/or the table
- whether **page breaks** are to be carried out after each group

> **Creating Groups Interactively**

As soon as you create a new node in the empty chart interactively, a group node will be created automatically. In the **Edit Data** dialog you can enter a group name into the data field that has been selected for **Group by** in the **Grouping** dialog.

If you want to create a new group, please proceed as follows: Create a node in an existing group. Double-click on the node to open the **Edit Data** dialog box. Then enter a group name into the data field that has been selected for **Group by** in the **Grouping** dialog. Then the new group will be created.

> **Regrouping Nodes Interactively**

If the user moves an activity from one group to another with the help of the mouse, the value in the grouping field is automatically adjusted.

> **Empty Groups**

If you delete all nodes of a group, the title of this group in the table will still remain. If you remove the grouping and apply it again, or if you finish the program and restart it, the titles of all empty groups will disappear.

> **Resorting subgroups interactively**

You can change the sorting order of subgroups interactively. To do so, mark the summary bar of the subgroup which you want to move. Then move the phantom of this subgroup up or down in the diagram. As soon as you place the phantom onto another summary bar of the same grouping level, an arrow

will indicate whether you can insert the summary bar above or below the other one. As soon as you release the mouse button, the group will all nodes will be inserted at the place selected.

> **All nodes of all groups in one line/in separate lines/expanded/collapsed**

With a few lines of code you can specify how the nodes of all groups are to be displayed. In the following example the nodes of all groups (two grouping levels) are displayed in one line.

Example Code VB.NET

```
Private Sub mnuAllNodesOneRow_Click()

    Dim groupCltn As VcGroupCollection
    Dim group As VcGroup
    Dim subGroupCltn As VcGroupCollection
    Dim subGroup As VcGroup

    groupCltn = VcGantt1.GroupCollection

    For Each group In groupCltn
        subGroupCltn = group.SubGroups
        group.NodesArrangedInOneRow = True
        For Each subGroup In subGroupCltn
            subGroup.NodesArrangedInOneRow = True
        Next
    Next

End Sub
```

Example Code C#

```
private void mnuAllNodesOneRow_Click(object sender, System.EventArgs e)
{
    VcGroupCollection groupCltn = VcGantt1.GroupCollection;
    VcGroupCollection subGroupCltn;

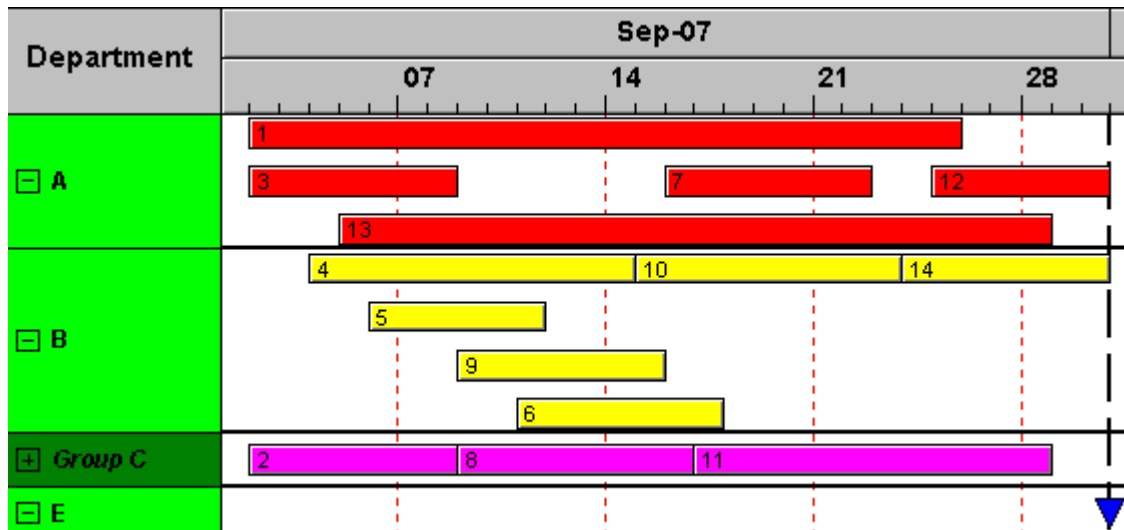
    foreach (VcGroup group in groupCltn)
    {
        subGroupCltn = group.SubGroups;
        group.NodesArrangedInOneRow = true;
        foreach (VcGroup subGroup in subGroupCltn)
        {
            subGroup.NodesArrangedInOneRow = true;
        }
    }
}
```

In the same way you can display all nodes of all groups in separate rows (group.NodesArrangedInOneRow = False), expand them (group.Collapsed = False) or collapse them (group.Collapsed = True).

> Diagram with Grouping Option "Nodes in One Line"

This section gives a brief description of the **Nodes in separate rows** option for the group layout of the activities.

A diagram with this option enabled looks something like this:



The grouping procedure is the same as previously described, where each activity was displayed in a separate line. If the **Nodes in separate rows** option of the **Grouping** dialog was not set, a whole group is displayed in one row. Naturally, the activities may overlap within the row. In order to make overlays visible, the group can be expanded, which means that, strictly speaking, the option should be called "In as few lines as possible". In their expanded state, you are free to move overlapping activities until all overlays have gone. Thus an expanded diagram ensures that overlapping activities (even if they do so for only a second) can instantly be recognized.

When a group is collapsed (as is Group C in the example), it shows that it comprises several activities, but there is no way to recognize whether there are overlays.

Naturally, with this type of diagram, it makes no sense to arrange the activities in a table format. Therefore, we recommend to display annotations on layers instead or to use tooltips for their identification.

> Displaying Overlaying nodes

If the **Nodes in separate rows** mode was not selected, you can specify via the sorting order which nodes lie above the others. The nodes are sorted according to their sorting order, that means that the last node in the sorting order lies above all others and is completely visible.

> **Summary bars**

Summary bars can be displayed in the grouping lines. You can specify whether summary bars are to be displayed and for which grouping levels.

To display summary bars at grouping levels defined by **Grouping level**, in the **Grouping** dialog, the check box **Summary Bar** needs to be ticked for the corresponding level.

The VcGantt property **SummaryBarsVisible** lets you specify/enquire at run time whether summary bars are visible or not. On condition that the grouping is not hierarchically, you can switch on or off the summary bars for each level separately with the help of the parameter **GroupingLevel**.

On the **Layer** property page you can specify the appearance of the summary bars by creating appropriate layers which visualize the summary bars. You may define one layer for all or several levels as well as different layers for each level, e. g. the layer "Summary bar 1" for the first level, "Summary bar 2" for the second level etc.

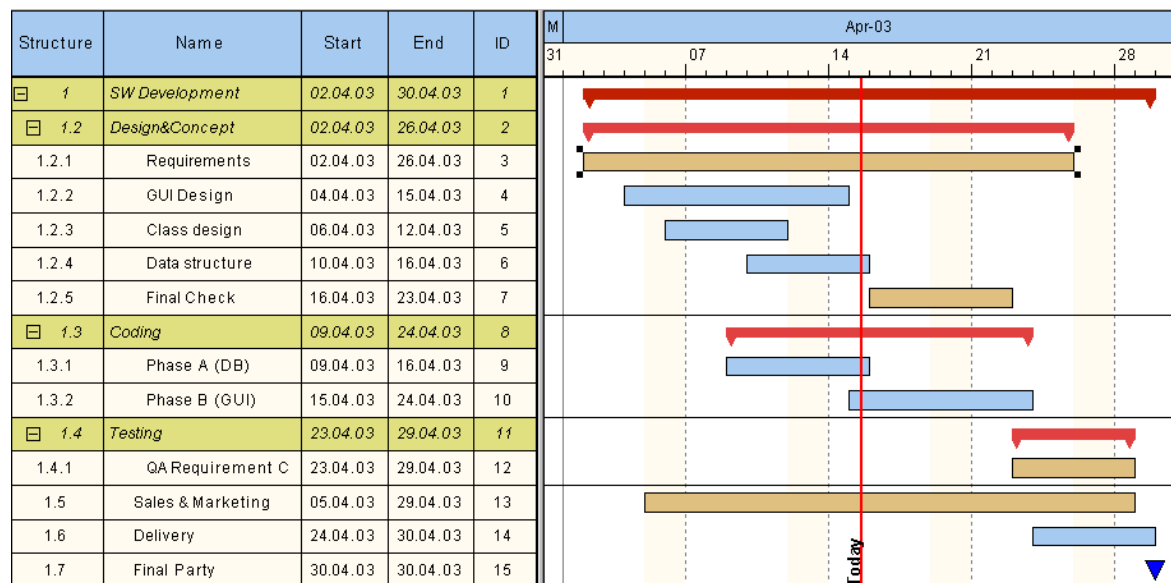
Now you have to assign corresponding filters to the summary bars so that the visualization is carried out at all. Filters can be created in the **Administrate Filters** dialog, e.g. the filter "Summary bar 1" for the first level. In order to specify the appropriate level, in the **Edit Filter** dialog select "<summary bar-level>" under **Field name**, select the right **Operator** (equal, greater or equal, greater than, etc.) and enter the desired level number in the **Comparison** field.

3.11 Hierarchical Order

An alternative way of arranging activities by levels is to use a hierarchy. For a hierarchical order the project data has to contain a hierarchy code of the format:

1., 1.1, 1.1.1, 1.2, 1.2.1, ...

A hierarchical layout could look something like this:



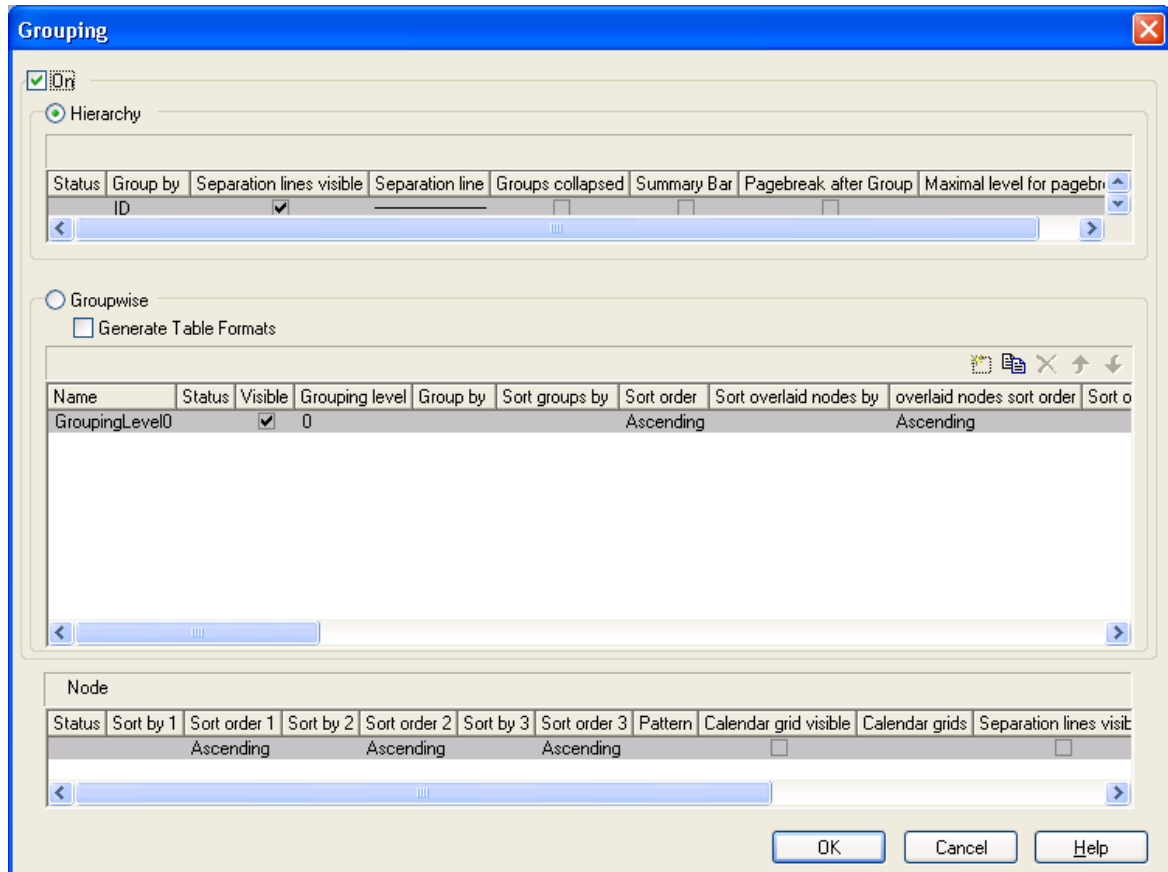
The symbols + and – are automatically displayed in front of the superordinate activities. Sublevels are indented automatically. By clicking on the - symbol, the structure of subordinate activities will fold (collapse); by clicking on the + symbol it will unfold (expand).

The program does not check whether the dates of the superordinate activities comprehend the dates of the subordinate ones, i.e. the program does not verify or set activity durations.

If the hierarchical order is selected, no other grouping or sorting option can be set.

A hierarchical arrangement can be set in the **Grouping** dialog:

122 Important Concepts: Hierarchical Order



To apply a hierarchical order, the check box **Hierarchy** needs to be ticked. After this, a data field that contains the structure code has to be selected from the combo box (**Group by**).

In addition, the below hierarchy features can be set:

- Display and style of **Separation lines**
- whether the activities should be collapsed on the start of the program (**Groups collapsed**)
- whether summary bars are to be displayed (**Summary Bar**)
- whether **page breaks** are to be carried out after each group and up to which level they are to be carried out

The table formats **Hierarchy** and **HierarchyCollapsed** are used to display the summary activities. They can be modified in the **Edit Table Format** dialog.

> Moving nodes interactively

You can move nodes interactively. A node moved will be inserted above or below the reference node, even in collapsed groups.

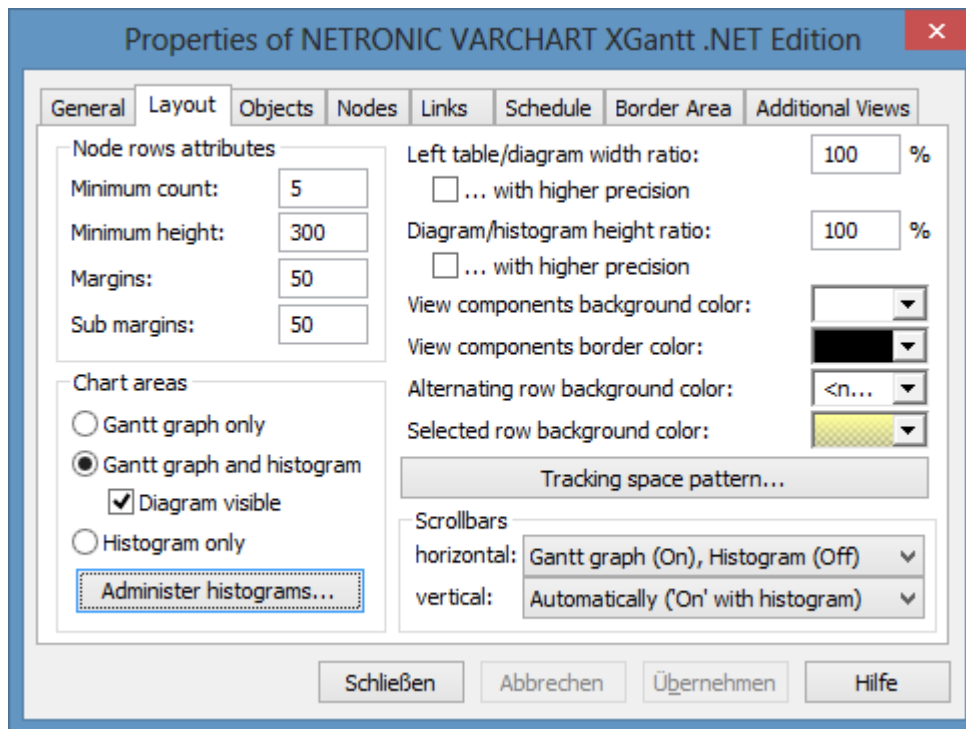
> **Moving summary bars interactively**

You can move summary bars interactively in the same way as nodes. Subordinated nodes are automatically moved together with their higher level bars.

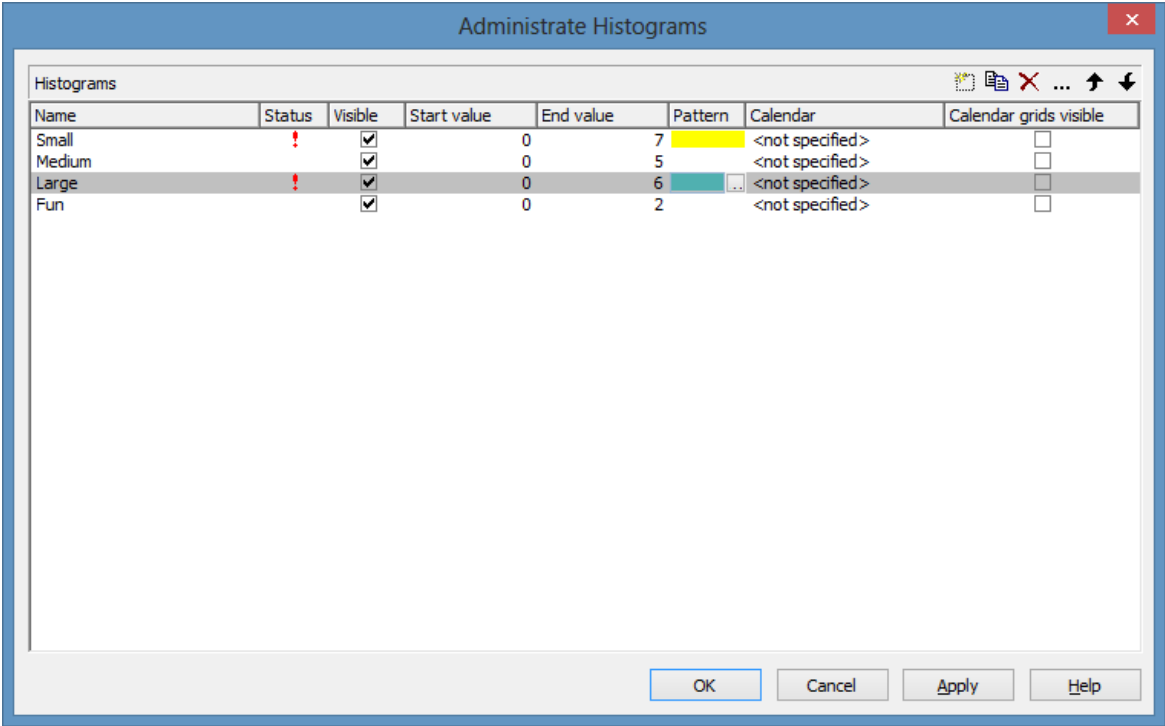
3.12 Histograms

Histograms are used to summarize activities to curves, with the activities fulfilling certain criteria.

On the **Layout** property page you can specify whether the Gantt chart only, the histogram only or both, the Gantt chart and the histogram should be displayed.



To select the histogram(s) to be displayed and for editing histograms, please click on the button **Administer histograms**.

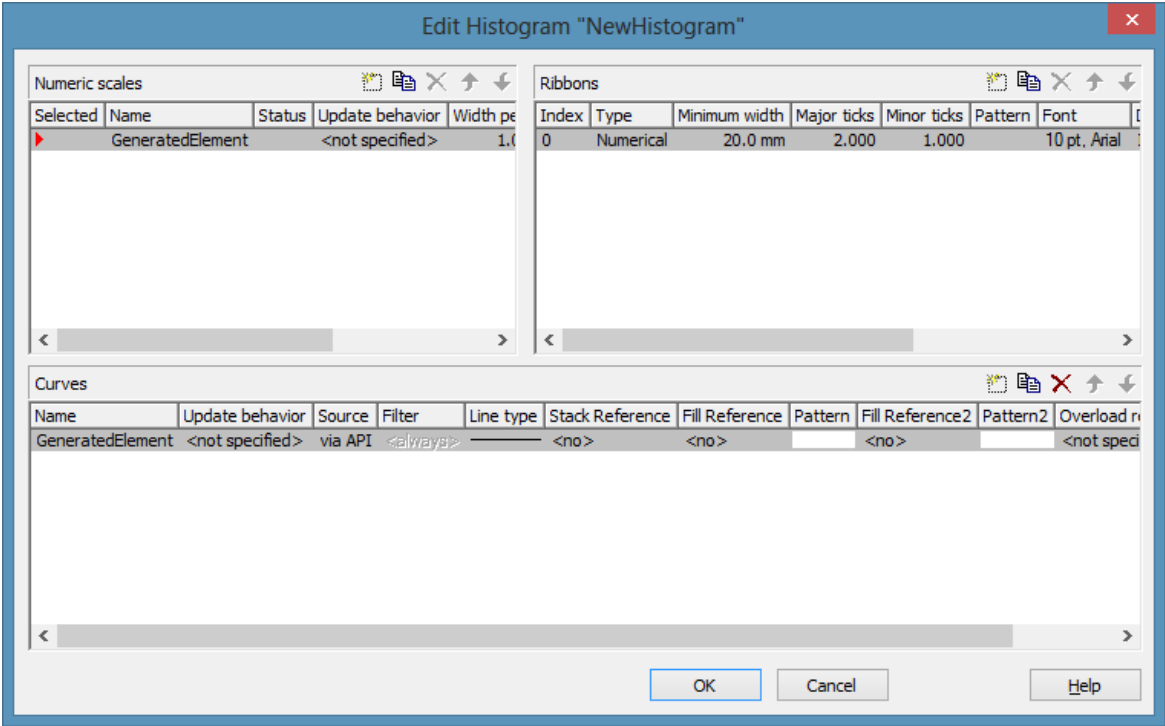


The dialog allows to select one or more histograms to be displayed.

A histogram has a numeric scale (y axis) and curves. Ist x axis is scaled by the Gantt chart time scale.

For each histogram you can define the start and the end values of the numeric scale separately.

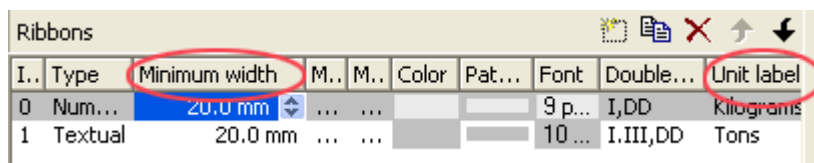
To edit a histogram, please mark it and click on the **Edit** button (...).



> Numeric scales

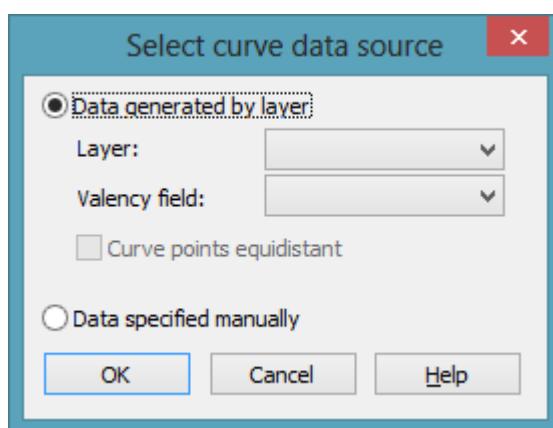
In the above dialog you can define different numeric scales and select the one to apply to the histogram. You can define the grading of a numeric scale in y direction (**Width per unit**). Beside, you can decide whether a line grid is to be displayed and you can define its features.

In the **Ribbons** area you can assign one or more ribbons to the numeric scale being edited. To each ribbon you can set a **Type**, a **Minimum width**, a number to define after how many units a **Major** or a **Minor tick** should occur, you can assign a background **Color**, **Font** features and a **Double format**. Furthermore you can tick the option **Object draw events** if you want to design the contents of the ribbon by yourself and you can specify a **Unit label** to designate the units used in the ribbon. For the unit label, please ensure that sufficient space is provided by the minimum width of the ribbon; otherwise the label cannot be displayed and remains invisible.



> Histogram curves

A histogram may contain several capacity curves, for each of which you can individually define a number of parameters. A **Name** and a **Line type** are the most simple ones. For a curve to be generated, a **Source** needs to be specified to supply the data, i.e. the values of the points. For this, please click on the **Source** field and then on the **Edit** button (...). The below dialog will appear:



You can choose between two alternatives:

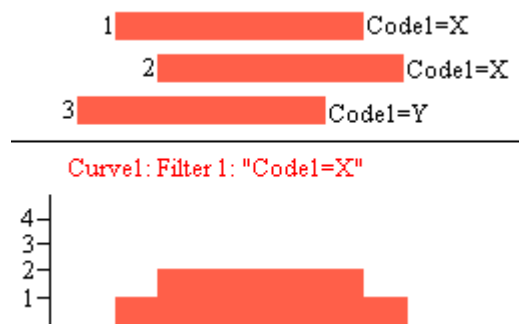
> 1. Data generated by layers

The curves are generated from data of the activities. When summing up the activities to a curve, the start and end dates of each of the selected layers (e.g. named "Start-End") are used.

When generating data from layers, you can once more filter out certain layers that shall add to the curve. For this, in the upstream **Edit Histogram** dialog you can choose a **Filter** for each curve for the selection of activities.

Example:

Only those activities that fulfil the conditions of Filter1 contribute to Curve1. Filter1 contains the condition "Code1 = X", i.e. only the activities 1 and 2 for which "Code1 = X" applies contribute to Curve 1.



For curves generated by layers you can select a data field of the activity that provides the fraction on the scale, by which the curve is to rise on the numeric scale when an activity is added (e.g. by 5 units).

> 2. Data defined by the API

This option allows to set curve values by the API. The latter offers the VcCurve method **SetValues** by which the points can be freely defined.

For curves generated by the API you can set in the **Select curve data source** dialog whether the curve points are to be created with a regular spacing (**Curve points equidistant**), where the curve points cannot be modified interactively, or in arbitrary positions that allow for interactive modifications.

Curve points equidistant: Please specify the start value (**startDate**) and the y values of the histogram curve. The curve points are calculated from the start value and the values set to **Time Unit** and **Smallest time interval** (on the property page **General**).

Set Values X, Y1, Y2, Y3, ...

Curves generated in this way cannot be edited interactively.

Curve points not equidistant: Specify pairs of x and y values:

Set Values X1, Y1

Set Values X2, Y2

Set Values X3, Y3...

The fields **Time Unit** and **Smallest time interval** do not apply. A curve generated this way can be edited interactively.

> **Reference curve**

A typical way to use curves defined by the **SetValues** method of the API is the capacity curve. It mostly serves as a reference curve, that forms areas with other curves, to which colors and patterns can be assigned.

The **Fill Reference** field allows you to specify the curve to limit the opposite end of the area (starting at the curve being edited). If you select <Flatline>, the area will reach down to the x axis, possibly hiding other curves on its way (which depends on the drawing priorities of the curves). In applications often the capacity curve is assigned here.

The curve line and fill pattern of the area you can set in the **Line type** and **Pattern** fields.

If you click on the entry in the **Line type** field, the **Line attributes** dialog box will appear where you can define the color, thickness and type of each curve line. If you click on the **Pattern** field, the **Pattern Attributes** dialog box will appear where you can define a pattern and the foreground/background colors for the fill pattern below a curve.

In addition to the first reference curve, you can specify a second one.

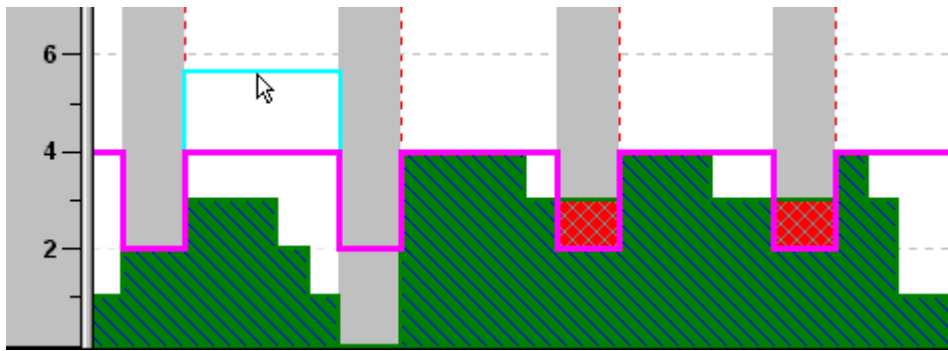
For this, please select a curve in the **Fill Reference2** field. The area between the curves is displayed only if the y values of the curve being edited are higher than the y values of the second reference curve, i.e. if the area expands below the curve being edited.

In the corresponding **Pattern** field you can specify the pattern and the color of the area.

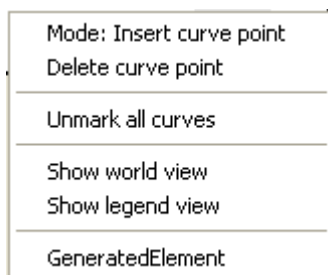
Examples of handling histograms you can find in "Tutorial: Displaying Histograms" and in "Tutorial: Displaying Capacity Bottlenecks".

> **Modifying capacity curves interactively**

A user can modify curves defined by the API (which mostly are capacity curves) interactively, for example when capacities have changed. This is only possible if the curves were not generated from equidistant curve points. The horizontal parts of the curve can be moved up or down by mouse. A phantom indicates the new position of the availability curve.



Beside, you can add or delete single curve points interactively. To do so, use the right mouse button to click in the histogram area. The below context menu will appear:



If several capacity curves were defined, their names will be indicated in the context menu. Clicking on a curve name will mark the curve.

Select **Mode: Insert curve point** and click on the capacity curve. Each click will add a curve point.

To delete a curve point, use the right mouse button to pop up its context menu and select the option **Delete curve point**.

> Displaying curve points

If you click on a curve that was generated by API (but that was not generated from equidistant curve points), the curve points will be displayed as small black squares. By clicking again on the histogram curve you can make the squares disappear again.

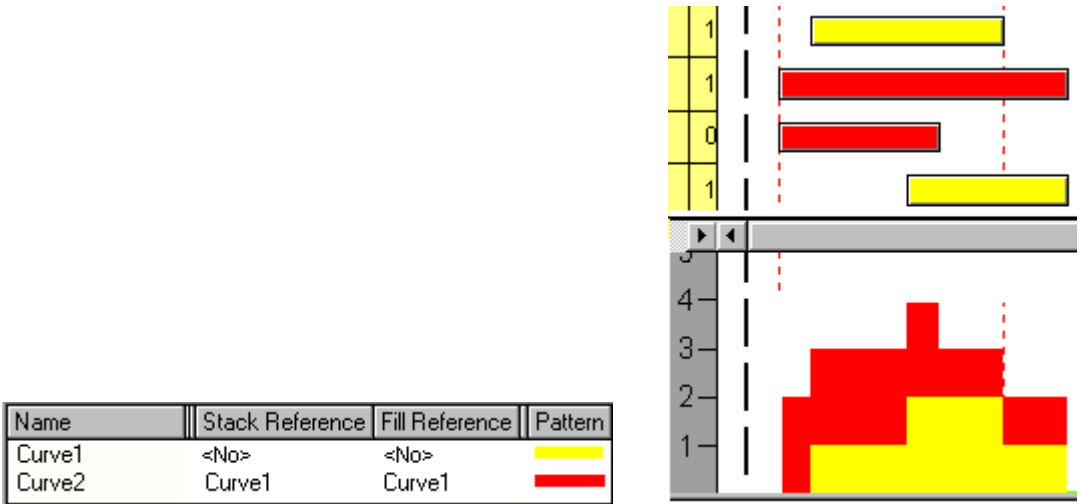
> Stacking Curves

Stacking curves is useful, for instance, if a histogram displays different curves that visualize the workload of single resources but in addition shall indicate the total workload.

In the example below, there are red and yellow activities indicating that they occupy different resources. Filters of corresponding conditions collect them to form a curve each. Being stacked the curves indicate the total workload of the system.

130 Important Concepts: Histograms

To display the red urve on top of the yellow one, please select the yellow curve (Curve 1) to be the **Stack Reference** of the red curve (Curve 2).
(If you select <No> in the **Stack Reference** for all curves, the curves will overlap visually and may hide each other).



Curve2 is stacked on Curve1.

3.13 How to Use a Calendar

A calendar represents a gapless sequence of working and non-working times. In a calendar that has a variable profile (shift calendar) different periods succeed repeatedly, such as morning, late or night shifts. A calendar itself has no visual appearance, it merely is the logic differentiation of working and non-working times. A calendar can become visible only if assigned to a **CalendarGrid** object.

In VARCHART XGantt a calendar also serves to derive start and end dates of nodes from durations. If no other option is set, a pre-defined base calendar named **BaseCalendar** is used for all calculations. In the base calendar the days Monday to Friday are defined as working periods, while Sunday and Saturday are free of work. The base calendar can be modified if required.

Defining a Calendar

A calendar can be defined at design time by the property pages or at runtime by the application programming interface (API). In this chapter we explain the basic handling of calendars from a developer's point of view and give some programming samples in C#. Defining a calendar by property pages is described in detail by the chapter **Property Pages and Dialog Fields**.

In the **VcGantt** control, an object **VcCalendarCollection** exists which takes care of the administration of all calendars. It has similar administrative functions as other collections have in VARCHART XGantt. The pre-defined **BaseCalendar** and any other calendar created at design time automatically form a part of the collection.

A new calendar can be created by the method **Add** of the **CalendarCollection** object. The method requires a unique name for a calendar to be identified. Initially, a new calendar merely consists of working time.

Please note: A calendar must contain at least a single time interval, since a calendar containing but non-working time cannot exist.

To make the results of our programming samples verifiable in the pictures of the Gantt diagrams, a constant time period is defined from 1.1.2011 to 31.12.2011 for the time scale in the programming samples. A calendar can only become visible in the background of a Gantt diagram if it was activated in the collection:

Example Code C#

```
// Creating and activating a new calendar
vcGantt1.TimeScaleEnd = new DateTime(2012, 1, 1);
vcGantt1.TimeScaleStart = new DateTime(2011, 1, 1);
```


132 Important Concepts: How to Use a Calendar

```
VcCalendar calendar =  
vcGantt1.CalendarCollection.Add("CompanyCalendar1");  
vcGantt1.CalendarCollection.Active = calendar;
```

Example Code VB.NET

```
'Creating and activating a new calendar  
vcGantt1.TimeScaleEnd = New DateTime(2012, 1, 1)  
vcGantt1.TimeScaleStart = New DateTime(2011, 1, 1)  
Dim calendar As VcCalendar =  
vcGantt1.CalendarCollection.Add("CompanyCalendar1")  
vcGantt1.CalendarCollection.Active = calendar
```

January 2011																																	
01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30				

If you now wish to re-activate the default base calendar, you can do this by the below settings:

Example Code C#

```
// Re-activating the default calendar  
VcCalendar calendar =  
vcGantt1.CalendarCollection.CalendarByName("BaseCalendar");  
vcGantt1.CalendarCollection.Active = calendar;
```

Example Code VB.NET

```
' Re-activating the default calendar  
Dim calendar As VcCalendar =  
vcGantt1.CalendarCollection.CalendarByName("BaseCalendar")  
vcGantt1.CalendarCollection.Active = calendar
```

January 2011																															
01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	

In the below example we will show how to define a working time profile by **intervals**. An irregular pattern of non-working days is to be defined: January 1st of 2011 and the period from January 6th to January 20th 2011, except for the two days of the 10th and 11th:

Example Code C#

```
// Defining non-working times
vcGantt1.TimeScaleEnd = new DateTime(2012, 1, 1);
vcGantt1.TimeScaleStart = new DateTime(2011, 1, 1);
VcCalendar calendar =
vcGantt1.CalendarCollection.Add("CompanyCalendar1");
vcGantt1.CalendarCollection.Active = calendar;

VcInterval interval = calendar.IntervalCollection.Add("NewYear");
interval.CalendarProfileName = "<NONWORK>";
interval.StartDateTime = new DateTime(2011, 1, 1);
interval.EndDateTime = new DateTime(2011, 1, 2);

interval = calendar.IntervalCollection.Add("NonworkPeriod");
interval.CalendarProfileName = "<NONWORK>";
interval.StartDateTime = new DateTime(2011, 1, 6);
interval.EndDateTime = new DateTime(2011, 1, 21);

interval = calendar.IntervalCollection.Add("WorkPeriod");
interval.CalendarProfileName = "<WORK>";
interval.StartDateTime = new DateTime(2011, 1, 11);
interval.EndDateTime = new DateTime(2011, 1, 13);
vcGantt1.CalendarCollection.Update();
```

Example Code VB.NET

```
' Defining non-working times
vcGantt1.TimeScaleEnd = New DateTime(2012, 1, 1)
vcGantt1.TimeScaleStart = New DateTime(2011, 1, 1)
Dim calendar As VcCalendar =
vcGantt1.CalendarCollection.Add("CompanyCalendar1")
vcGantt1.CalendarCollection.Active = calendar

Dim interval As VcInterval = calendar.IntervalCollection.Add("NewYear")
interval.CalendarProfileName = "<NONWORK>"
interval.StartDateTime = New DateTime(2011, 1, 1)
interval.EndDateTime = New DateTime(2011, 1, 2)

interval = calendar.IntervalCollection.Add("NonworkPeriod")
interval.CalendarProfileName = "<NONWORK>"
interval.StartDateTime = New DateTime(2011, 1, 6)
interval.EndDateTime = New DateTime(2011, 1, 21)

interval = calendar.IntervalCollection.Add("WorkPeriod")
interval.CalendarProfileName = "<WORK>"
interval.StartDateTime = New DateTime(2011, 1, 11)
interval.EndDateTime = New DateTime(2011, 1, 13)
vcGantt1.CalendarCollection.Update()
```

January 2011																															
01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	

134 Important Concepts: How to Use a Calendar

Visually, non-working times can be identified by the light gray shade. Since working times by default do not have a color, the white background of the diagram remains visible in them. In the next step, we want working times to appear in a light yellow color and non-working times in light blue. The colors are produced by graphical attributes that can be defined at the intervals.

Example Code C#

```
// Assigning colors to intervals
vcGantt1.TimeScaleEnd = new DateTime(2012, 1, 1);
vcGantt1.TimeScaleStart = new DateTime(2011, 1, 1);
VcCalendar calendar =
vcGantt1.CalendarCollection.Add("CompanyCalendar1");
vcGantt1.CalendarCollection.Active = calendar;
vcGantt1.TimeScaleCollection.FirstTimeScale().get_Section(0).
get_CalendarGrid(0).UseGraphicalAttributesOfIntervals = true;

VcInterval interval = calendar.IntervalCollection.Add("Work");
interval.CalendarProfileName = "<WORK>";
interval.BackgroundColor = Color.LightYellow;
interval.UseGraphicalAttributes = true;

VcInterval interval = calendar.IntervalCollection.Add("NewYear");
interval.CalendarProfileName = "<NONWORK>";
interval.StartDateTime = new DateTime(2011, 1, 1);
interval.EndDateTime = new DateTime(2011, 1, 2);
interval.BackgroundColor = Color.FromArgb(212,227,245);
interval.UseGraphicalAttributes = true;

interval = calendar.IntervalCollection.Add("NonworkPeriod");
interval.CalendarProfileName = "<NONWORK>";
interval.StartDateTime = new DateTime(2011, 1, 6);
interval.EndDateTime = new DateTime(2011, 1, 21);
interval.BackgroundColor = Color.FromArgb(212,227,245);
interval.UseGraphicalAttributes = true;

interval = calendar.IntervalCollection.Add("WorkPeriod");
interval.CalendarProfileName = "<WORK>";
interval.StartDateTime = new DateTime(2011, 1, 11);
interval.EndDateTime = new DateTime(2011, 1, 13);
interval.BackgroundColor = Color.LightYellow;
interval.UseGraphicalAttributes = true;

vcGantt1.CalendarCollection.Update();
```

Example Code VB.NET

```
' Assigning colors to intervals
vcGantt1.TimeScaleEnd = New DateTime(2012, 1, 1)
vcGantt1.TimeScaleStart = New DateTime(2011, 1, 1)
Dim calendar As VcCalendar =
vcGantt1.CalendarCollection.Add("CompanyCalendar1")
vcGantt1.CalendarCollection.Active = calendar
Dim get_CalendarGrid(0).UseGraphicalAttributesOfIntervals As
vcGantt1.TimeScaleCollection.FirstTimeScale().get_Section(0). = True

Dim interval As VcInterval = calendar.IntervalCollection.Add("Work")
interval.CalendarProfileName = "<WORK>"
```

```

interval.BackgroundColor = Color.LightYellow
interval.UseGraphicalAttributes = True

interval = calendar.IntervalCollection.Add("NewYear")
interval.CalendarProfileName = "<NONWORK>"
interval.StartDateTime = New DateTime(2011, 1, 1)
interval.EndDateTime = New DateTime(2011, 1, 2)
interval.BackgroundColor = Color.FromArgb(212,227,245)
interval.UseGraphicalAttributes = True

interval = calendar.IntervalCollection.Add("NonworkPeriod")
interval.CalendarProfileName = "<NONWORK>"
interval.StartDateTime = New DateTime(2011, 1, 6)
interval.EndDateTime = New DateTime(2011, 1, 21)
interval.BackgroundColor = Color.FromArgb(212,227,245)
interval.UseGraphicalAttributes = True

interval = calendar.IntervalCollection.Add("WorkPeriod")
interval.CalendarProfileName = "<WORK>"
interval.StartDateTime = New DateTime(2011, 1, 11)
interval.EndDateTime = New DateTime(2011, 1, 13)
interval.BackgroundColor = Color.LightYellow
interval.UseGraphicalAttributes = True

vcGantt1.CalendarCollection.Update()

```

January 2011																															
01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	

The below sample shows how to define a week where Monday to Friday are a working time while the weekend is free of work. The options introduced so far do not suffice for this; an object of the type **VcCalendarProfile** is required.

Please note: In VARCHART XGantt, VcCalendarProfile objects can be defined on a global or on a local level. Local calendar profile objects can only be used in the calendar in which they were defined, while global objects simultaneously can be used in different calendars. In our programming samples, merely local calendar profile objects are used. In terms of functions, local calendars do not differ from global ones. If a local and a global profile of identical names were created, within the corresponding calendar only the local profile is addressed; the global profile cannot be accessed.

A calendar profile of the type **vcWeekProfile** allows to describe working and non-working times of the days of a week. A week profile becomes effective only after it was added to the interval collection of the calendar. Setting **StartDateTime** and **EndDateTime** can be omitted, since we want our

136 Important Concepts: How to Use a Calendar

settings to be valid for the complete period of the calendar without any restriction. The calendar profiles of the pre-set names **<WORK>** and **<NONWORK>** have a defined meaning: they are used to allocate working and nonworking times.

Example Code C#

```
// Defining a week profile
VcCalendarProfile calendarProfile =
calendar.CalendarProfileCollection.Add("WeekProfile");
calendarProfile.Type = VcCalendarProfileType.vcWeekProfile;

VcInterval interval = calendarProfile.IntervalCollection.Add("Mo-Fr");
interval.CalendarProfileName = "<WORK>";
interval.StartWeekday = VcWeekday.vcMonday;
interval.EndWeekday = VcWeekday.vcFriday;

interval = calendarProfile.IntervalCollection.Add("Sa");
interval.CalendarProfileName = "<NONWORK>";
interval.BackgroundColor = Color.FromArgb(255, 246, 159);
interval.StartWeekday = VcWeekday.vcSaturday;
interval.EndWeekday = VcWeekday.vcSaturday;

interval = calendarProfile.IntervalCollection.Add("So");
interval.CalendarProfileName = "<NONWORK>";

interval.BackgroundColor = Color.FromArgb(251, 211, 170);
interval.StartWeekday = VcWeekday.vcSunday;
interval.EndWeekday = VcWeekday.vcSunday;
interval = calendar.IntervalCollection.Add("StandardWeek");
interval.CalendarProfileName = "WeekProfile";
```

Example Code VB.NET

```
' Defining a week profile
dim calendarProfile as VcCalendarProfile
Set calendar.Profile =
VcGantt1.CalendarProfileCollection.Add("WeekProfile")
calendarProfile.Type = VcCalendarProfileType.vcWeekProfile

VcInterval interval = calendarProfile.IntervalCollection.Add("Mo-Fr")
interval.CalendarProfileName = "<WORK>"
interval.StartWeekday = VcWeekday.vcMonday
interval.EndWeekday = VcWeekday.vcFriday

interval = calendarProfile.IntervalCollection.Add("Sa")
interval.CalendarProfileName = "<NONWORK>"
interval.BackgroundColor = Color.FromArgb(255, 246, 159)
interval.StartWeekday = VcWeekday.vcSaturday
interval.EndWeekday = VcWeekday.vcSaturday

interval = calendarProfile.IntervalCollection.Add("Su")
interval.CalendarProfileName = "<NONWORK>"
interval.BackgroundColor = Color.FromArgb(251, 211, 170)
interval.StartWeekday = VcWeekday.vcSunday
interval.EndWeekday = VcWeekday.vcSunday

interval = calendar.IntervalCollection.Add("StandardWeek")
```

```
interval.CalendarProfileName = "WeekProfile"
```

Distinguishing working and non-working times within a single day requires a day profile that allows to specify a precise clock time, for example from 8.00 h to 12.00 h am and from 1.00 h to 5.00 h pm. Since a day profile newly created consists of working time only, any interruption is to be defined as a non-working interval.

Example Code C#

```
// Defining a day profile
VcCalendarProfile calendarProfile =
calendar.CalendarProfileCollection.Add("DayProfile");
calendarProfile.Type = VcCalendarProfileType.vcDayProfile;

VcInterval interval =
calendarProfile.IntervalCollection.Add("Interval_1");
// 00:00-8:00
interval.CalendarProfileName = "<NONWORK>";
interval.StartTime = new DateTime(2011, 1, 1, 0, 0, 0);
interval.EndTime = new DateTime(2011, 1, 1, 8, 0, 0);

interval = calendarProfile.IntervalCollection.Add("Interval_2");
// 12:00-13:00
interval.CalendarProfileName = "<NONWORK>";
interval.StartTime = new DateTime(2011, 1, 1, 12, 0, 0);
interval.EndTime = new DateTime(2011, 1, 1, 13, 0, 0);

interval = calendarProfile.IntervalCollection.Add("Interval_3");
// 17:00-24:00
interval.CalendarProfileName = "<NONWORK>";
interval.StartTime = new DateTime(2011, 1, 1, 17, 0, 0);
interval.EndTime = new DateTime(2011, 1, 1, 0, 0, 0);
```

Example Code VB.NET

```
' Defining a day profile
Dim calendarProfile As VcCalendarProfile =
calendar.CalendarProfileCollection.Add("DayProfile")
calendarProfile.Type = VcCalendarProfileType.vcDayProfile

Dim interval As VcInterval =
calendarProfile.IntervalCollection.Add("Interval_1")
' 00:00-8:00
interval.CalendarProfileName = "<NONWORK>"
interval.StartTime = New DateTime(2011, 1, 1, 0, 0, 0)
interval.EndTime = New DateTime(2011, 1, 1, 8, 0, 0)

interval = calendarProfile.IntervalCollection.Add("Interval_2")
' 12:00-13:00
interval.CalendarProfileName = "<NONWORK>"
interval.StartTime = New DateTime(2011, 1, 1, 12, 0, 0)
interval.EndTime = New DateTime(2011, 1, 1, 13, 0, 0)

interval = calendarProfile.IntervalCollection.Add("Interval_3")
' 17:00-24:00
interval.CalendarProfileName = "<NONWORK>"
interval.StartTime = New DateTime(2011, 1, 1, 17, 0, 0)
```

138 Important Concepts: How to Use a Calendar

```
interval.EndTime = New DateTime(2011, 1, 1, 0, 0, 0)
```

The clock time is set by the object **DateTime**. The date fraction is ignored since it is meaningless in this context. The date only needs to be set in the constructor, to set a value to all parameters required by the constructor. In **Interval_3** it is important to specify 0 h instead of 24 h, since the latter is not accepted in the **DateTime** object.

Recurring days of a year, such as **New Year's Eve** on the 1st of January or **Christmas** and **Boxing Day** on the 25th and 26th of December are defined by a calendar profile which covers a whole year.

Example Code C#

```
// Setting a profile of fixed annual holidays
VcCalendarProfile calendarProfile =
calendar.CalendarProfileCollection.Add("YearProfile");
calendarProfile.Type = VcCalendarProfileType.vcYearProfile;

VcInterval interval = calendarProfile.IntervalCollection.Add("New
Year");
interval.CalendarProfileName = "<NONWORK>";
interval.DayInStartMonth = 1 ;
interval.StartMonth = VcMonth.vcJanuary;
interval.DayInEndMonth = 1;
interval.EndMonth = VcMonth.vcJanuary;
SetAppearanceForHolidays(interval);

interval = calendarProfile.IntervalCollection.Add("Christmas");
interval.CalendarProfileName = "<NONWORK>";
interval.DayInStartMonth = 25 ;
interval.StartMonth = VcMonth.vcDecember;
interval.DayInEndMonth = 26;
interval.EndMonth = VcMonth.vcDecember;
SetAppearanceForHolidays(interval);
```

Example Code VB.NET

```
' Setting a profile of fixed annual holidays
Dim calendarProfile As VcCalendarProfile =
calendar.CalendarProfileCollection.Add("YearProfile")
calendarProfile.Type = VcCalendarProfileType.vcYearProfile

Dim interval As VcInterval = calendarProfile.IntervalCollection.Add("New
Year")
interval.CalendarProfileName = "<NONWORK>"
interval.DayInStartMonth = 1
interval.StartMonth = VcMonth.vcJanuary
interval.DayInEndMonth = 1
interval.EndMonth = VcMonth.vcJanuary
SetAppearanceForHolidays(interval)

interval = calendarProfile.IntervalCollection.Add("Christmas")
interval.CalendarProfileName = "<NONWORK>"
interval.DayInStartMonth = 25
interval.StartMonth = VcMonth.vcDecember
interval.DayInEndMonth = 26
```

```
interval.EndMonth = VcMonth.vcDecember
SetAppearanceForHolidays(interval)
```

To avoid repeated settings that produce identical appearances of holidays, we collect the calls in a method named **SetAppearanceForHolidays**:

Example Code C#

```
// Method to set the visual appearance of holidays
void SetAppearanceForHolidays(VcInterval interval)
{
    interval.BackgroundColor = Color.FromArgb(255, 255, 164, 164);
    interval.Pattern = VcFillPattern.vcWeavePattern;
    interval.PatternColor = Color.FromArgb(255, 64, 64, 64);
    interval.LineColor = Color.FromArgb(255, 128, 128, 128);
    interval.LineThickness = 1;
    interval.LineType = VcLineType.vcSolid;
    interval.UseGraphicalAttributes = true;
}
```

Example Code VB.NET

```
' Method to set the visual appearance of holidays
Private Sub SetAppearanceForHolidays(ByVal interval As VcInterval)
    interval.BackgroundColor = Color.FromArgb(255, 255, 164, 164)
    interval.Pattern = VcFillPattern.vcWeavePattern
    interval.PatternColor = Color.FromArgb(255, 64, 64, 64)
    interval.LineColor = Color.FromArgb(255, 128, 128, 128)
    interval.LineThickness = 1
    interval.LineType = VcLineType.vcSolid
    interval.UseGraphicalAttributes = True
End Sub
```

Please note: The color properties become effective only in those intervals, the `CalendarProfileName` of which was set either to **<WORK>** or to **<NONWORK>**. In addition, the interval property **UseGraphicalAttribute** needs to be set to **true**. The same is valid for the calendarGrid property **UseGraphicalAttributesOfIntervals**.

Floating holidays such as Easter, and other holidays that depend on them have to be calculated for each year and need to be assigned to the calendar as fixed dates. The below method is very useful for this:

Example Code C#

```
// Method to find floating holidays
public enum Anniversary
{
    AshWednesday,
    GoodFriday,
    EasterSunday,
    EasterMonday,
    FeastOfCorpusChristi,
    AscensionOfChrist,
    WhitSunday,
    WhitMonday,
```


140 Important Concepts: How to Use a Calendar

```
        CentralEuropeanSummerTimeStart,  
        CentralEuropeanSummerTimeEnd  
    }  
  
private DateTime calculateAnniversaryForYear(int year, Anniversary  
specialDay)  
{  
    int g = year % 19;  
    int c = year / 100;  
    int h = (c - c / 4 - (8 * c + 13) / 25 + 19 * g + 15) % 30;  
    int i = h - (h / 28) * (1 - (29 / (h + 1)) * ((21 - g) / 11));  
    int j = (year + year / 4 + i + 2 - c + c / 4) % 7;  
    int month = 3 + (i - j + 40) / 44;  
    int day = i - j + 28 - 31 * (month / 4);  
  
    int dayOffset = 0;  
    switch (specialDay)  
    {  
        case Anniversary.AshWednesday:  
            dayOffset = -40;  
            break;  
        case Anniversary.GoodFriday:  
            dayOffset = -2;  
            break;  
        case Anniversary.EasterSunday:  
            break;  
        case Anniversary.EasterMonday:  
            dayOffset = 1;  
            break;  
        case Anniversary.AscensionOfChrist:  
            dayOffset = 39;  
            break;  
        case Anniversary.WhitSunday:  
            dayOffset = 49;  
            break;  
        case Anniversary.WhitMonday:  
            dayOffset = 50;  
            break;  
        case Anniversary.FeastOfCorpusChristi:  
            dayOffset = 60;  
            break;  
        case Anniversary.CentralEuropeanSummerTimeStart:  
            month = 3;  
            day = 31 - Convert.ToInt32(new DateTime(year, 3,  
31).DayOfWeek);  
            break;  
        case Anniversary.CentralEuropeanSummerTimeEnd:  
            month = 10;  
            day = 31 - Convert.ToInt32(new DateTime(year, 10,  
31).DayOfWeek);  
            break;  
    }  
    return new DateTime(year, month, day).AddDays(dayOffset);  
}
```

Example Code VB.NET

```
' Method to find floating holidays  
Public Enum Anniversary  
    AshWednesday
```

```

    GoodFriday
    EasterSunday
    EasterMonday
    FeastOfCorpusChristi
    AscensionOfChrist
    WhitSunday
    WhitMonday
    CentralEuropeanSummerTimeStart
    CentralEuropeanSummerTimeEnd
End Enum

Private Function calculateAnniversaryForYear(ByVal year As Integer,
ByVal specialDay As Anniversary) As DateTime
    Dim g As Integer = Decimal.Remainder( year , 19 )
    Dim c As Integer = year / 100
    Dim h As Integer = (c - c / 4 - (8 * c + 13) / 25 + 19 * g + 15) % 30
    Dim i As Integer = h - (h / 28) * (1 - (29 / (h + 1)) * ((21 - g) / 11))
    Dim j As Integer = (year + year / 4 + i + 2 - c + c / 4) % 7
    Dim month As Integer = 3 + (i - j + 40) / 44
    Dim day As Integer = i - j + 28 - 31 * (month / 4)

    Dim dayOffset As Integer = 0
    Select Case specialDay
        Case Anniversary.AshWednesday
            dayOffset = -40
        Case Anniversary.GoodFriday
            dayOffset = -2
        Case Anniversary.EasterSunday
            Exit Function
        Case Anniversary.EasterMonday
            dayOffset = 1
        Case Anniversary.AscensionOfChrist
            dayOffset = 39
        Case Anniversary.WhitSunday
            dayOffset = 49
        Case Anniversary.WhitMonday
            dayOffset = 50
        Case Anniversary.FeastOfCorpusChristi
            dayOffset = 60
        Case Anniversary.CentralEuropeanSummerTimeStart
            month = 3
            day = 31 - Convert.ToInt32(New DateTime(year, 3, 31).DayOfWeek)
        Case Anniversary.CentralEuropeanSummerTimeEnd
            month = 10
            day = 31 - Convert.ToInt32(New DateTime(year, 10,
31).DayOfWeek)
    End Select
    Return New DateTime(year, month, day).AddDays(dayOffset)
End Function

```

In the next step, the week profile and the holiday profile are assigned to the calendar as intervals. Then the floating holidays are calculated and assigned to the calendar in the same way:

Example Code C#

```

// Assembling the week profile, the holiday profile and the floating
holidays into an interval
interval = calendar.IntervalCollection.Add("Weekly_Pattern");

```

142 Important Concepts: How to Use a Calendar

```
interval.CalendarProfileName = "WeekProfile";

interval = calendar.IntervalCollection.Add("Yearly_Pattern");
interval.CalendarProfileName = "YearProfile";

int startYear = vcGantt1.TimeScaleStart.Year;
int endYear   = vcGantt1.TimeScaleEnd.Year;

for (int i=startYear; i<=endYear; i++)
{
    interval = calendar.IntervalCollection.Add("GoodFriday_" +
i.ToString());
    interval.CalendarProfileName = "<NONWORK>";
    interval.StartDateTime = calculateAnniversaryForYear(i,
Anniversary.GoodFriday);
    interval.EndDateTime = interval.StartDateTime;
    SetAppearanceForHolidays(interval);

    interval = calendar.IntervalCollection.Add("EasterMonday_" +
i.ToString());
    interval.CalendarProfileName = "<NONWORK>";
    interval.StartDateTime = calculateAnniversaryForYear(i,
Anniversary.EasterMonday);
    interval.EndDateTime = interval.StartDateTime;
    SetAppearanceForHolidays(interval);

    interval = calendar.IntervalCollection.Add("FeastOfCorpusChristi_" +
i.ToString());
    interval.CalendarProfileName = "<NONWORK>";
    interval.StartDateTime = calculateAnniversaryForYear
(i, Anniversary.FeastOfCorpusChristi);
    interval.EndDateTime = interval.StartDateTime;
    SetAppearanceForHolidays(interval);

    interval = calendar.IntervalCollection.Add("AscensionOfChrist_" +
i.ToString());
    interval.CalendarProfileName = "<NONWORK>";
    interval.StartDateTime = calculateAnniversaryForYear(i,
Anniversary.AscensionOfChrist);
    interval.EndDateTime = interval.StartDateTime;
    SetAppearanceForHolidays(interval);

    interval = calendar.IntervalCollection.Add("WhitMonday_" +
i.ToString());
    interval.CalendarProfileName = "<NONWORK>";
    interval.StartDateTime = calculateAnniversaryForYear(i,
Anniversary.WhitMonday);
    interval.EndDateTime = interval.StartDateTime;
    SetAppearanceForHolidays(interval);
}

vcGantt1.CalendarCollection.Update();
```

Example Code VB.NET

```
' Assembling the week profile, the holiday profile and the floating
holidays into an interval
interval = calendar.IntervalCollection.Add("Weekly_Pattern")
interval.CalendarProfileName = "WeekProfile"
```

```

interval = calendar.IntervalCollection.Add("Yearly_Pattern")
interval.CalendarProfileName = "YearProfile"

Dim startYear As Integer = vcGantt1.TimeScaleStart.Year
Dim endYear As Integer = vcGantt1.TimeScaleEnd.Year

Dim i As Integer
For i = startYear To endYear Step i + 1
    interval = calendar.IntervalCollection.Add("GoodFriday_" +
i.ToString())
    interval.CalendarProfileName = "<NONWORK>"
    interval.StartDateTime = calculateAnniversaryForYear(i,
Anniversary.GoodFriday)
    interval.EndDateTime = interval.StartDateTime
    SetAppearanceForHolidays(interval)

    interval = calendar.IntervalCollection.Add("EasterMonday_" +
i.ToString())
    interval.CalendarProfileName = "<NONWORK>"
    interval.StartDateTime = calculateAnniversaryForYear(i,
Anniversary.EasterMonday)
    interval.EndDateTime = interval.StartDateTime
    SetAppearanceForHolidays(interval)

    interval = calendar.IntervalCollection.Add("FeastOfCorpusChristi_" +
i.ToString())
    interval.CalendarProfileName = "<NONWORK>"
    interval.StartDateTime = calculateAnniversaryForYear
(i, Anniversary.FeastOfCorpusChristi)
    interval.EndDateTime = interval.StartDateTime
    SetAppearanceForHolidays(interval)

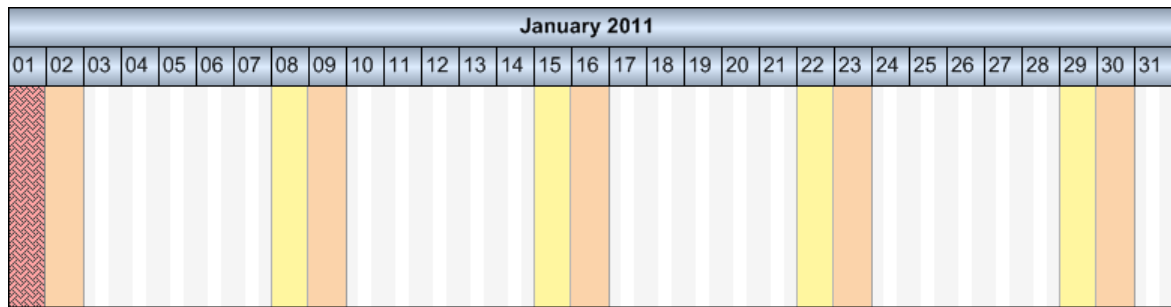
    interval = calendar.IntervalCollection.Add("AscensionOfChrist_" +
i.ToString())
    interval.CalendarProfileName = "<NONWORK>"
    interval.StartDateTime = calculateAnniversaryForYear(i,
Anniversary.AscensionOfChrist)
    interval.EndDateTime = interval.StartDateTime
    SetAppearanceForHolidays(interval)

    interval = calendar.IntervalCollection.Add("WhitMonday_" +
i.ToString())
    interval.CalendarProfileName = "<NONWORK>"
    interval.StartDateTime = calculateAnniversaryForYear(i,
Anniversary.WhitMonday)
    interval.EndDateTime = interval.StartDateTime
    SetAppearanceForHolidays(interval)
Next

vcGantt1.CalendarCollection.Update()

```

144 Important Concepts: How to Use a Calendar



These are the steps in summary that are required to put assemble a calendar. Depending on the requirements single steps may be omitted:

1. Creating day profiles of different working days
2. Assembling a week profile by using the day profiles
3. Defining a holiday profile
4. Assigning the week profile and the holiday profile to the interval collection of the calendar
5. Assigning additional dates (e.g. floating holidays) to the interval collection

The interval object allows to define periods that can be interpreted as working time or as non-working time. The periods are distinguished to be **<WORK>** or **<NONWORK>** by the **CalendarProfileName** property. By this property, a calendar can also refer to other existing profiles and adopt their settings. When setting this property please take into account that only certain profile types can be assigned, depending on the interval type. The interval type implicitly is selected by the chosen profile type. The pre-set default value of the **calendar profile**, which is **vcDayProfile**, can be modified by a corresponding setting initially, that is, before defining intervals.

Object	Profile Type Chosen	Interval Type Assigned
VcCalendar		vcCalendarInterval
VcCalendarProfile	vcYearProfile	vcYearProfileInterval
	vcWeekProfile	vcWeekProfileInterval
	vcDayProfile	vcDayProfileInterval
	vcVariableProfile	vcVariableProfileInterval

The profile type suggests the allowed interval type. For example, a day profile always requires intervals of the type **vcDayProfileInterval**.

Interval Type	<WORK>	<NONWORK>	vcDayProfile	vcWeekProfile	vcYearProfile	vcVariableProfile
vcCalendarInterval	■	■	■	■	■	■
vcVariableProfileInterval	■	■	■	■	■	
vcYearProfileInterval	■	■	■	■		
vcWeekProfileInterval	■	■	■			
vcDayProfileInterval	■	■				

Calendar profiles can show the types **day profile**, **week profile**, **year profile** and **variable profile**. In a day profile, intervals can only be defined by clock times that range within the limits of a day. A week profile holds day profiles to apply on certain days. A year profile assigns selected day profiles that apply to a single recurring day or to a couple of recurring days. A variable profile contains a sequence of different working times. Depending on the interval types **vcCalendarInterval**, **vcDayProfileInterval**, **vcWeekProfileInterval**, **vcYearProfileInterval** and **vcVariableProfileInterval** only some properties of the object are of relevance. The below table maps profile types and relevant properties.

vcCalendar-Interval	vcYearProfile-Interval	vcWeekProfile-Interval	vcDayProfile-Interval	vcVariable-Interval
StartDateTime	StartMonth	StartWeekday	StartTime	Duration
EndDateTime	EndMonth	EndWeekday	EndTime	TimeUnit
	DayInEndMonth			
	DayInStartMonth			

A **CalendarInterval** describes a unique time span in a precisely defined interval. Example: May 5th, 2010 from 11:30 h to September, 15th 2010 17:00 h.

A **YearProfileInterval** allows to define days or a time spans that recur once a year. Example: May 1st or December, 24th - 26th.

A **WeekProfileInterval** handles a single or several days of a week. Example: Saturday, or Monday - Friday.

A **DayProfileInterval** deals with time specifications that range within a day. Example: 8.00 h to 17.00 h.

A **VariableProfile** describes a time span without referring to a defined date or time. The unit of the time span may be days, hours, minutes or seconds and is specified by the property `TimeUnit` of the interval object. Example: 4 hours.

How to Calculate with Calendars

Calculations in a calendar are not necessarily visible in the time scale. The method **AddDuration** of the object **Calendar** calculates the final date from the start date and the specified number of working time units while taking into account non-working periods. Passing time units of negative signs will result in calculating the start date from a given end date. The method **CalcDuration** being a complement of the method **AddDuration** calculates the number of working time units (duration) from a given start and an end date.

> How the Calculating Methods Work

Please note: Working time units specified as days, hours, minutes or seconds need to correspond to what was defined by the property `TimeUnit` of the `VcGantt` object.

The method **AddDuration** ensures, that the dates calculated always are located in a working time interval. At the same time, a backward calculation does not necessarily provide a result equal to the source value of the forward calculation, if the source value had been situated in a non-working time.

> Limited Reversibility of Calculations

When activities are interactively created or modified, `VARCHART XGantt` automatically cares that activities cannot start or finish within non-working times. If you wish the behavior to be consistent while creating or modifying nodes by the API, you need to ensure this by manually correcting the start or end date. For this, a start date being situated in a non-working time needs to be moved to the beginning of the succeeding working time interval, and an end date correspondingly to the end of the previous working time interval. There are methods to identify the limits of intervals. They are discussed in detail in the below chapter.

Example Code C#

```
if (calendar.IsWorkTime(startDate) == false)
    startDate = calendar.GetNextIntervalBorder(startDate);

if (calendar.IsWorkTime(endDate) == false)
    endDate = calendar.GetStartOfInterval(endDate);
```

Example Code VB.NET

```

If calendar.IsWorkTime(startDate) = False Then
    startDate = calendar.GetNextIntervalBorder(startDate)
End If

If calendar.IsWorkTime(endDate) = False Then
    endDate = calendar.GetStartOfInterval(endDate)
End If

```

> Daylight Saving Time

VARCHART XGantt automatically supports daylight saving time. In central Europe, DST starts on the last Sunday in the month of March and finishes on the last Sunday in the month of October. On the start of DST the clocks are put forward from 2:00 h to 3:00 h and at its end they are put back from 3:00 h to 2:00 h.

Start of daylight saving time:

00:00 h	01:00 h	03:00 h	04:00 h	05:00 h	06:00 h	...
---------	---------	---------	---------	---------	---------	-----

End of daylight saving time:

00:00 h	01:00 h	02:00 h	02:00 h	03:00 h	04:00 h	...
---------	---------	---------	---------	---------	---------	-----

On the start day of daylight saving time, the method **calcDuration** retrieves a time span of 23 hours while on its final day, 25 hours are returned, if **TimeUnit** is set to hours. If set to days, the time span in both cases will be exactly 1 day.

Retrieving the Limits of Time Intervals

The methods of the **Calendar** object to retrieve the limits of a time interval **GetStartOfInterval**, **GetNextIntervalBorder** and **GetPreviousIntervalBorder** allow to iterate over working time intervals and non-working time intervals. The results returned are relative and refer to a reference date which is passed by the methods as a parameter.

A date can be checked for being located in a working time or in a non-working time by the method **IsWorkTime** of the Calendar object. Although the start date of a new interval equals the end date of the previous one, the start date always belongs to the new interval (open to the right).

The methods **GetEndOfPreviousWorkTime** and **GetStartOfNextWorkTime** do not provide new options but merely simplify the handling of working time intervals.

In the below programming sample, the time intervals of the calendar are retrieved and written to a file. Beside, the working time available in the given period is calculated:

Example Code C#

```
void writeCalendarIntervalsToFile (string filename, VcCalendar calendar,
DateTime startDate, DateTime endDate,
bool listWorkIntervals, bool listNonWorkIntervals)
{
    TextWriter tw = new
StreamWriter(Path.GetDirectoryName(Application.ExecutablePath) +
filename);
    tw.WriteLine("Time Intervals of {0} between \r\n{1} - {2}\r\n",
calendar.Name, startDate, endDate);
    DateTime tmpStartDate = startDate;
    while (tmpStartDate < endDate)
    {
        DateTime nextStartDate =
calendar.GetNextIntervalBorder(tmpStartDate);
        if (tmpStartDate == nextStartDate)
            nextStartDate = endDate;
        if (nextStartDate > endDate)
            nextStartDate = endDate;
        if (calendar.IsWorktime(tmpStartDate))
            if (listWorkIntervals)
                tw.WriteLine("{0} - {1} WorkInterval", tmpStartDate,
nextStartDate);
            else
                if (listNonWorkIntervals)
                    tw.WriteLine("{0} - {1} NonWorkInterval", tmpStartDate,
nextStartDate);
                tmpStartDate = nextStartDate;
        }
        int totalWorkTime = calendar.CalcDuration(startDate, endDate);
        tw.WriteLine("Total work time: {0} Units", totalWorkTime);
        tw.Close();
    }
}
```

Example Code VB.NET

```
Private Sub writeCalendarIntervalsToFile(ByVal filename As String,
ByVal calendar As VcCalendar, ByVal startDate As DateTime, ByVal endDate
As DateTime, ByVal listWorkIntervals As Boolean, ByVal
listNonWorkIntervals As Boolean)
    Dim tw As TextWriter = New
StreamWriter(Path.GetDirectoryName(Application.ExecutablePath) +
filename)
    tw.WriteLine("Time Intervals of {0} between \r\n{1} - {2}\r\n",
calendar.Name, startDate, endDate)
    Dim tmpStartDate As DateTime = startDate
    While tmpStartDate < endDate
        Dim nextStartDate As DateTime =
calendar.GetNextIntervalBorder(tmpStartDate)
        if (tmpStartDate = nextStartDate)
            nextStartDate = endDate
        if (nextStartDate > endDate)
            nextStartDate = endDate
        if (calendar.IsWorktime(tmpStartDate))
            if (listWorkIntervals)
                tw.WriteLine("{0} - {1} WorkInterval", tmpStartDate,
nextStartDate)
            else
                if (listNonWorkIntervals)
                    tw.WriteLine("{0} - {1} NonWorkInterval", tmpStartDate,
nextStartDate)
                tmpStartDate = nextStartDate
            }
        int totalWorkTime = calendar.CalcDuration(startDate, endDate)
        tw.WriteLine("Total work time: {0} Units", totalWorkTime)
        tw.Close()
    }
}
```

```

        if (listWorkIntervals)
            tw.WriteLine("{0} - {1} WorkInterval", tmpStartDate,
nextStartDate)
        else
            if (listNonWorkIntervals)
                tw.WriteLine("{0} - {1} NonWorkInterval", tmpStartDate,
nextStartDate)
            tmpStartDate = nextStartDate
        End While
        Dim totalWorkTime As Integer =
calendar.CalcDuration(startDate,endDate)
        tw.WriteLine("Total work time: {0} Units", totalWorkTime)
        tw.Close()
    End Sub

```

Please note: Intervals in the calendar can be specified as exactly as by seconds and may comprise an interval of 137 years (ulong in seconds) at maximum.

> Code to Write Intervals to File

Example Code C#

```

writeCalendarIntervalsToFile("CalenderIntervals.txt", calendar,
vcGantt1.TimeScaleStart, vcGantt1.TimeScaleEnd, true, true);

```

```

Time Intervals of CompanyCalendar_1 between
01.01.2011 00:00:00 - 01.01.2012 00:00:00

```

```

01.01.2011 00:00:00 - 02.01.2011 00:00:00 non-work time
02.01.2011 00:00:00 - 03.01.2011 00:00:00 non-work time
03.01.2011 00:00:00 - 03.01.2011 08:00:00 non-work time
03.01.2011 08:00:00 - 03.01.2011 12:00:00 work time
03.01.2011 12:00:00 - 03.01.2011 13:00:00 non-work time
03.01.2011 13:00:00 - 03.01.2011 17:00:00 work time
03.01.2011 17:00:00 - 04.01.2011 00:00:00 non-work time
04.01.2011 00:00:00 - 04.01.2011 08:00:00 non-work time
04.01.2011 08:00:00 - 04.01.2011 12:00:00 work time
04.01.2011 12:00:00 - 04.01.2011 13:00:00 non-work time
04.01.2011 13:00:00 - 04.01.2011 17:00:00 work time
04.01.2011 17:00:00 - 05.01.2011 00:00:00 non-work time
...
30.12.2011 00:00:00 - 30.12.2011 08:00:00 non-work time
30.12.2011 08:00:00 - 30.12.2011 12:00:00 work time
30.12.2011 12:00:00 - 30.12.2011 13:00:00 non-work time
30.12.2011 13:00:00 - 30.12.2011 17:00:00 work time
30.12.2011 17:00:00 - 31.12.2011 00:00:00 non-work time
31.12.2011 00:00:00 - 01.01.2012 00:00:00 non-work time

```

```

Total work time: 2064 Units

```

> Code to Write Intervals to File

Example Code VB.NET

```

writeCalendarIntervalsToFile("CalenderIntervals.txt", calendar,
vcGantt1.TimeScaleStart, vcGantt1.TimeScaleEnd, True, True)

```

150 Important Concepts: How to Use a Calendar

```
Time Intervals of CompanyCalendar_1 between
01.01.2011 00:00:00 - 01.01.2012 00:00:00

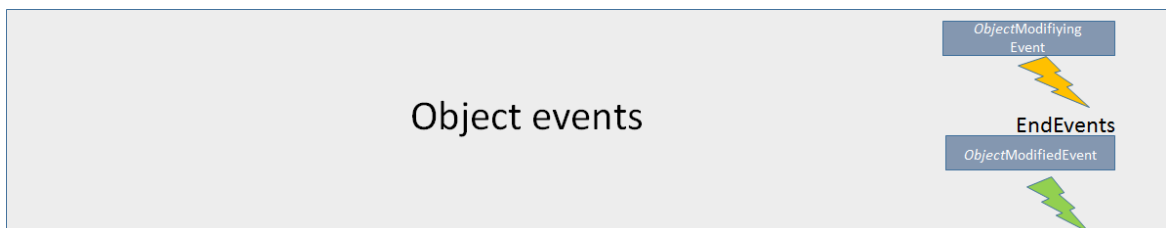
01.01.2011 00:00:00 - 02.01.2011 00:00:00 non-work time
02.01.2011 00:00:00 - 03.01.2011 00:00:00 non-work time
03.01.2011 00:00:00 - 03.01.2011 08:00:00 non-work time
03.01.2011 08:00:00 - 03.01.2011 12:00:00 work time
03.01.2011 12:00:00 - 03.01.2011 13:00:00 non-work time
03.01.2011 13:00:00 - 03.01.2011 17:00:00 work time
03.01.2011 17:00:00 - 04.01.2011 00:00:00 non-work time
04.01.2011 00:00:00 - 04.01.2011 08:00:00 non-work time
04.01.2011 08:00:00 - 04.01.2011 12:00:00 work time
04.01.2011 12:00:00 - 04.01.2011 13:00:00 non-work time
04.01.2011 13:00:00 - 04.01.2011 17:00:00 work time
04.01.2011 17:00:00 - 05.01.2011 00:00:00 non-work time
...
30.12.2011 00:00:00 - 30.12.2011 08:00:00 non-work time
30.12.2011 08:00:00 - 30.12.2011 12:00:00 work time
30.12.2011 12:00:00 - 30.12.2011 13:00:00 non-work time
30.12.2011 13:00:00 - 30.12.2011 17:00:00 work time
30.12.2011 17:00:00 - 31.12.2011 00:00:00 non-work time
31.12.2011 00:00:00 - 01.01.2012 00:00:00 non-work time

Total work time: 2064 Units
```

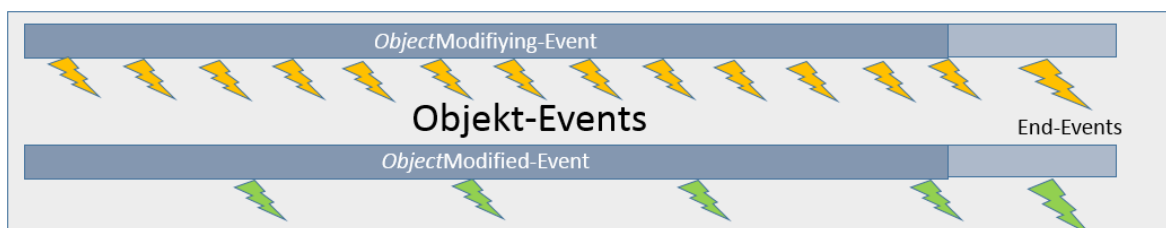
3.14 Interaction Events

During drag & drop interactions with the live update being enabled, receiving and processing information on the object would be quite useful.

In the default behavior, no feedback is given as to the status of the concerned object. Only when the mouse key is released, information on the old (before pressing the mouse key) and the new (after having released the mouse key) status is given by an **ObjectModifying** event. In addition, an **ObjectModified** event indicates that the operation is finished internally.



To solve this problem of not receiving information during mouse interactions, use the Interaction events that accompany and describe the interaction. Moreover, the object events' time of calling and frequency were modified as of XGantt version 5.0.



Interactions involved

We will explain events that describe the process of an interaction in VARCHART XGantt and the objects involved in greater detail, i.e. "Drag(Drop)" events during interactions that

- start with pressing the left mouse key at an object
- carry out movements with the mouse key being pressed
- end with releasing the left mouse key
- are treated in the course of "Live Update"

Terminology

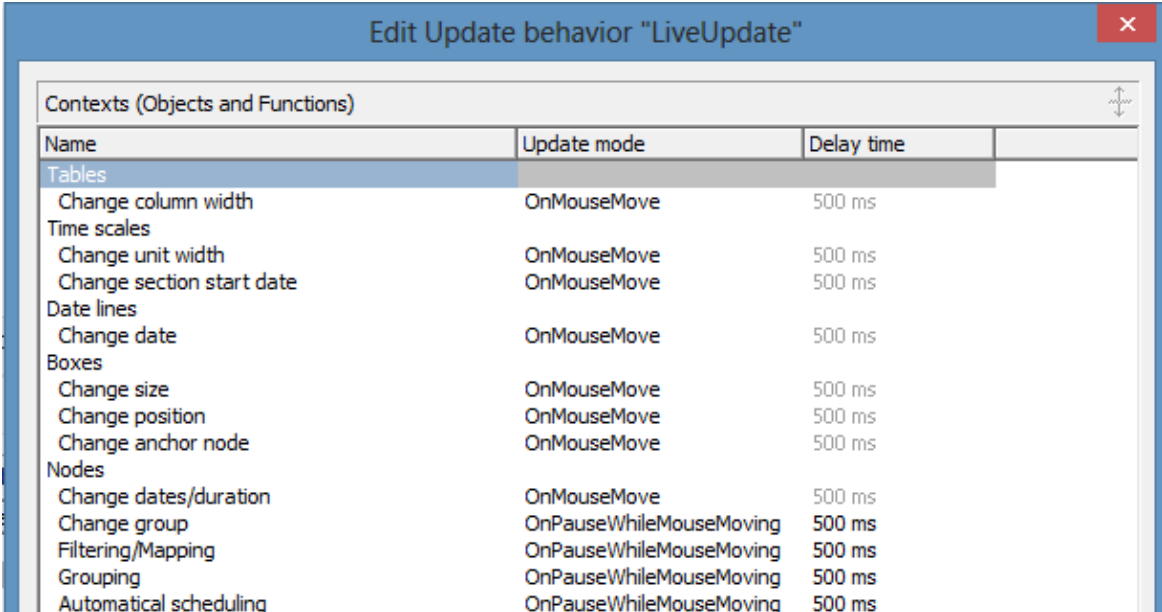
For a better understanding we'd like to further explain some terms that are used in the text.

> Object Events

Object events, such as **VcDateLineModifying**, **VcDateLineModified**, **VcNode-Modifying**, **VcNode-Modified** etc., are events, that, according to the practice already known up to now, are thrown at the end of an action during the addressed interactions.

> Live Update

Live update means that a "Drag Drop" action causes a "What if the object was updated here?" scenario to be shown permanently, this resulting in processing different contexts, such as direct or dependent functionalities during an interaction, at different times. If, for instance, a node is being moved, this results in modifying various data and the node's position, this in turn resulting in modifying the histogram curves or the summary bars, for instance. Depending on the settings in the Live Update dialog, the modifications will either come into effect at once or after hovering with the mouse a time span to be specified or at the end of the action on releasing the mouse key.



Name	Update mode	Delay time
Tables		
Change column width	OnMouseMove	500 ms
Time scales		
Change unit width	OnMouseMove	500 ms
Change section start date	OnMouseMove	500 ms
Date lines		
Change date	OnMouseMove	500 ms
Boxes		
Change size	OnMouseMove	500 ms
Change position	OnMouseMove	500 ms
Change anchor node	OnMouseMove	500 ms
Nodes		
Change dates/duration	OnMouseMove	500 ms
Change group	OnPauseWhileMouseMoving	500 ms
Filtering/Mapping	OnPauseWhileMouseMoving	500 ms
Grouping	OnPauseWhileMouseMoving	500 ms
Automatic scheduling	OnPauseWhileMouseMoving	500 ms

Example: What does the updates look like if the update behavior "OnMouseMove" is selected for the moving of nodes?

Immediate effects on the node:

- every date value of the node
- filters are evaluated, thus causing other colors, e.g., to appear in the table area

- osummary bars
- histogram curves

Modifications after a waiting period (500 ms)

- positioning the node in a group, for instance
- optimization with corresponding layout of the node order

Only updates that are necessary and meaningful in the total context of the action should be carried out, because otherwise the chart would become too restless.

InInteraction Events

From VARCHART XGantt 5.0 SR3 onward, object events can be processed already while the interaction is running, this objects being called InInteraction events.

Important: Be sure **to enable** the InInteraction events beforehand, either by the property **VcGantt.InInteractionEventsEnabled = true** or on the **General** property page.

Please note that when talking about interactions with nodes in the real mode, we will call the display object **Real (node)** and the data element in the chart **Chart** node. The chart node is not visible during the live interaction in the chart area because it will be replaced temporarily by the real node there, its presence, however, affecting the diagram in terms of ribbon height, optimization, colors in the table area etc.

This way, according information on the normal objects are delivered during the interaction matching with the displayed phantom or real node.

When a node is moved, every snapping into place of the node (depending on its time unit and increment) causes a **VcNodeModifying** to be thrown (yellow lightnings). The real node shows the possible position and the possible layout and describes this status by the **VcNodeModifying** event. The node (e.Node) being passed in the event args, represents the real node's status.

Important: This is why queries for properties of the chart nodes don't make sense or are not possible. Only the properties **get/setDataField**, **AllData**, **ID** can be retrieved or set.

If, depending on the selected updating context, e.g. "On pause while mouse moving", the real object is updated, this will be indicated by the **Modified**

154 Important Concepts: Interaction Events

event (green lightning). This can but doesn't have to happen at the same time as the Modifying events.

If a node is moved while the updating behavior "On mouse move" is selected, both events will appear at the same time.

To sum up the facts:

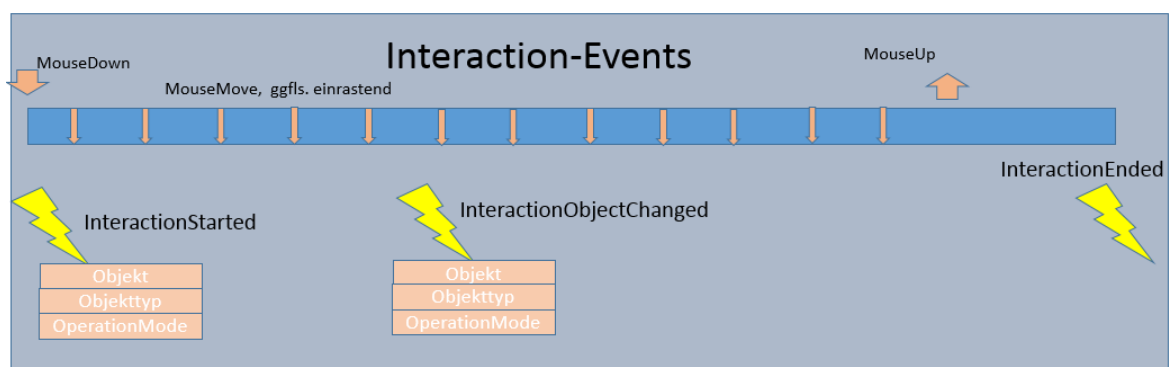
- If a node is moved, its modification, indicated by the real node, will be permanently described by the **VcNodeModifying** event.
- Modifications of the chart node are indicated by the **VcNodeModified** event.
- When the interaction is finished, upon releasing the mouse key, the concluding event pair, consisting of the **VcNodeModifying** and the **VcNodeModified** event are provided.

The concerned objects in events that use real nodes are the real objects.

In the last **VcNodeModifying** event, the chart node (as opposed to the previous **VcNodeModifying** events) with the values that were last set during the interaction is provided, i.e. the status at the time of the last small green lightning. **e.OldNode** of the **EventArgs** describes the status at the beginning of the action. This way, the start and end status of the interaction can be compared.

As always, the chart node is available in the last **VcNodeModified** event and all internal processes are finished.

Interaction Events



As described above, the object events are now thrown during and at the end of an interaction. The signature of the event handler, e.g. of the **VcNodeModifying** event don't differ there. But how to recognize whether the event has been thrown during or at the end of an interaction?

This could be important, because not every modification resulting from a mouse movement, for instance, is to be stored to a data base: This would

cause too much time-consuming effort. Of course, the data shall only be stored after the action was finished.

This problem can be solved now by some new events that accompany and describe the interaction and can be evaluated in the object events during the interaction.

As soon as the left mouse key is pressed, the **VcInteractionStarted** event delivers information on the object the mouse key is standing on (object and object type) and on what is happening with the object. Everything that is needed for the interaction can be prepared.

Tip: The update behavior can also be switched object- and context-specific here. In an extreme case, one could have one node react completely dynamical and another one with a blue phantom frame. Moreover, an according setting (**InInteractionEventsEnabled**) allows for an individual decision about whether the object events are to come also during the interaction or not.

Example: Node

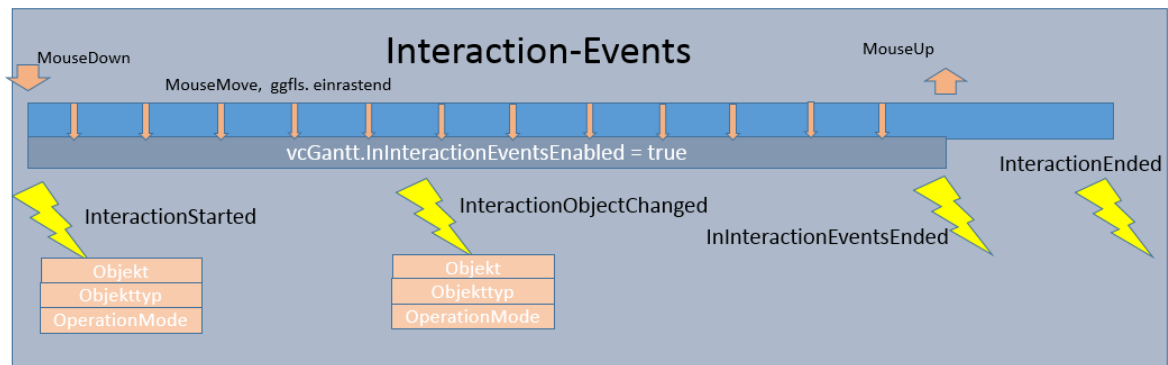
By

- Object: NodeObject
- Type: vcObjTypeNodeInDiagram
- OperationMode: vcIIMMoveNode
- upon pressing the left mouse key, the **VcInteractionStarted** event shows that the moving of a node in the chart has started.
- Information or elements that ought to accompany the interaction can be initialized here.
- **Creating Objects**
- In some interactions, there's no object available initially, e.g. when creating nodes or boxes. In this case, the event **VcInteractionObjectChanged** comes as soon as the object was created internally, being the real chart node where nodes are concerned.
- The end of the action is indicated by the **VcInteractionEnded** event. Every additional element having been used during the interaction can be removed here.
- When new objects are created with Interaction events, the process is as follows:
 - VcInteractionStarted
 - VcInteractionObjectChanged

156 Important Concepts: Interaction Events

- Modifying/Modified Events, showing modifications when creating an element
- Creating und Created Events
- VcInteractionEnded.

> InInteraction Events activated during the interaction



When the Interaction events are also enabled during the interaction (**vcGantt.InInteractionEventsEnabled = true**), there will be an additional event indicating the end of these events upon releasing the mouse key: **VcInInteractionEventsEnded**.

This makes it easy to differentiate the object events being thrown during the interaction from those that are thrown at the end of the interaction. If this event is thrown, the next object event will be the concluding event.

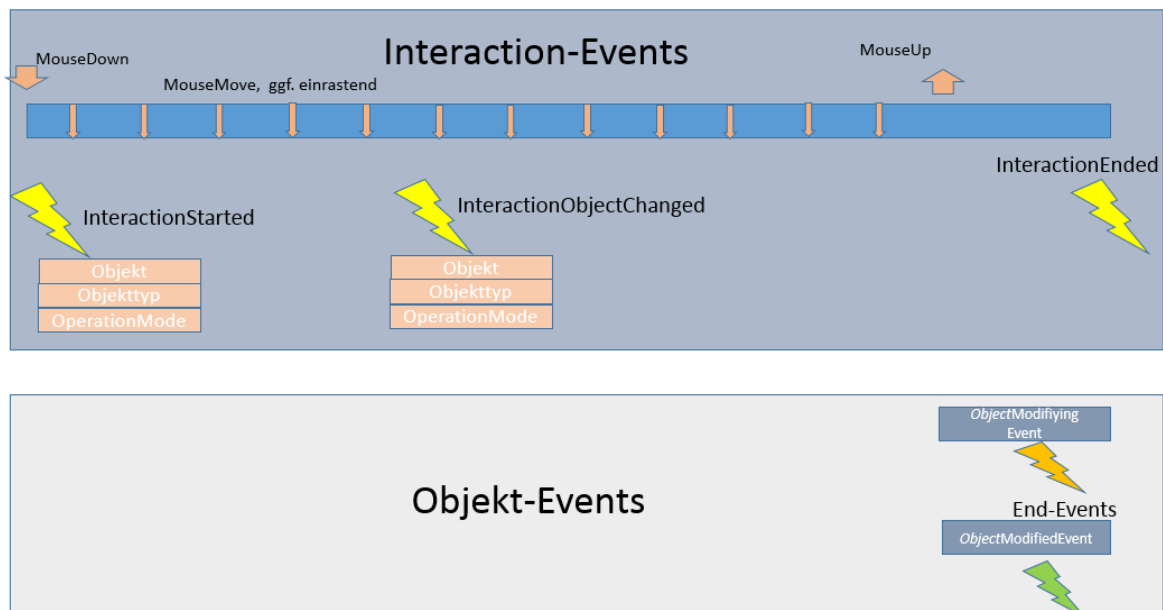
> Possible Scenarios

In other words, there are two possible conditions when using Interaction events.

Controlling an interaction with:

- InInteraction Events being switched off
- InInteraction Events being switched on

> **Cooperation with the events of the involved objects while the InInteraction events are deactivated**



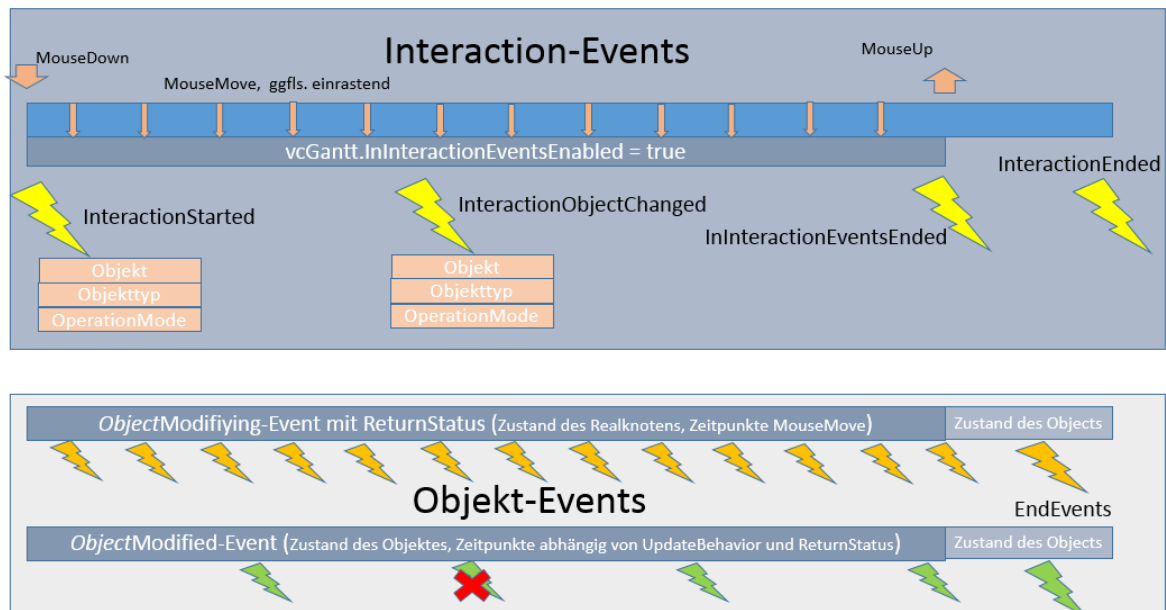
The screenshot shows how the Interaction (yellow lightnings) and the object events (ochre and green lightning) cooperate when InInteraction events are switched off (**vcGantt.InInteractionEventsEnabled = false**):

The interaction is started which is indicated by the **InteractionStarted** event.

When releasing the mouse key, the object events appear first, e.g. **VcNodeModifying** and **VcNodeModified** with a node. In other words this is the old behavior regarding object events so that existing code in the object events doesn't have to be modified if the InInteraction events are not used.

The end of the interaction is indicated by the **VcInteractionEnded** event.

> **Cooperation with the events of the involved objects while the InInteraction events are activated**



If the InInteraction events are used, the following events appear:

- **VcInteractionStarted** upon pressing the left mouse key
- Modifying and Modified events while the mouse is moved
- **VcInInteractionEventsEnded** and afterwards the finishing object events when the left mouse key is released
- **VcInteractionEnded** to indicate the end of the interaction.

Example: Moving a node:

The interaction starts when the left mouse key is pressed while the mouse cursor is at a node. The event **VcInteractionStarted** appears.

The events appearing upon moving the mouse indicate the status of the real node (**VcNodeModifying**) and while updating (**VcNodeModified1>**) **the chart node.**

When the mouse key is released, the VcInInteractionEventsEnded event appears

The object events **VcNodeModifying** and **VcNodeModified** indicate the status of the chart node at the end of the interaction.

The last to appear is the **VcInteractionEnded** event.

> **Example: Behavior of the object events when the node update behavior "On mouse move" is set**



Since the **VcNodeModifying** event allows for the **EventReturnstatus** (**e.ReturnStatus**) to be modified, this can now also be done during the interaction.

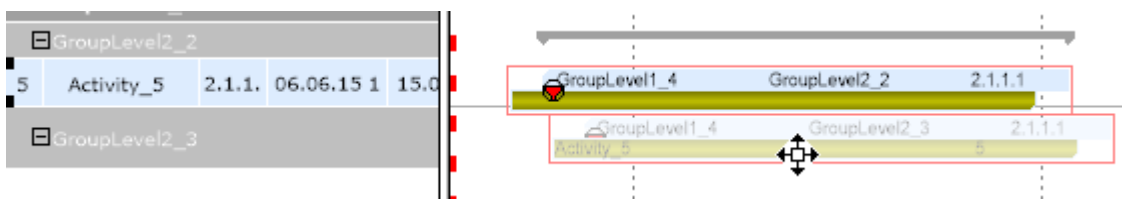
So, if **e.ReturnStatus = ReturnStatusFalse** indicates that the provided data are not "valid", the object in the chart will not be refreshed with the next possible update and the according **VcNodeModified** event will not be thrown.

This is visualized by the object remaining at its old place and the current position being still indicated by the phantom.

The status of objects visualized by reals (currently only nodes and node boxes) is indicated as follows:

The current position is visualized by a brightened real, the values of which also still being provided in the events.

The last valid status, i.e. the last one not returning **ReturnStatusFalse** as **e.ReturnStatus**, is indicated by another real, that quasi "gets stuck" there; this way both pieces of information are being visualized.



At the node, the values of the last valid status, i.e. that of the stuck real, correspond to the **e.OldNode** in the **VcNodeModifying-Event**

If the last **VcNodeModifying** event before the **VcInInteractionEventsEnded** was finished with **ReturnStatusFalse**, the last valid state will be provided in the End events.

There it can be decided whether to accept this state or not. If in the End event **ReturnStatusFalse** is set, the original start status will be restored.

Practical Tip: We recommend to create an "accompanying **InteractionInfo**" object that provide the needed information on the interaction in the events and can be evaluated accordingly.

3.15 Layers

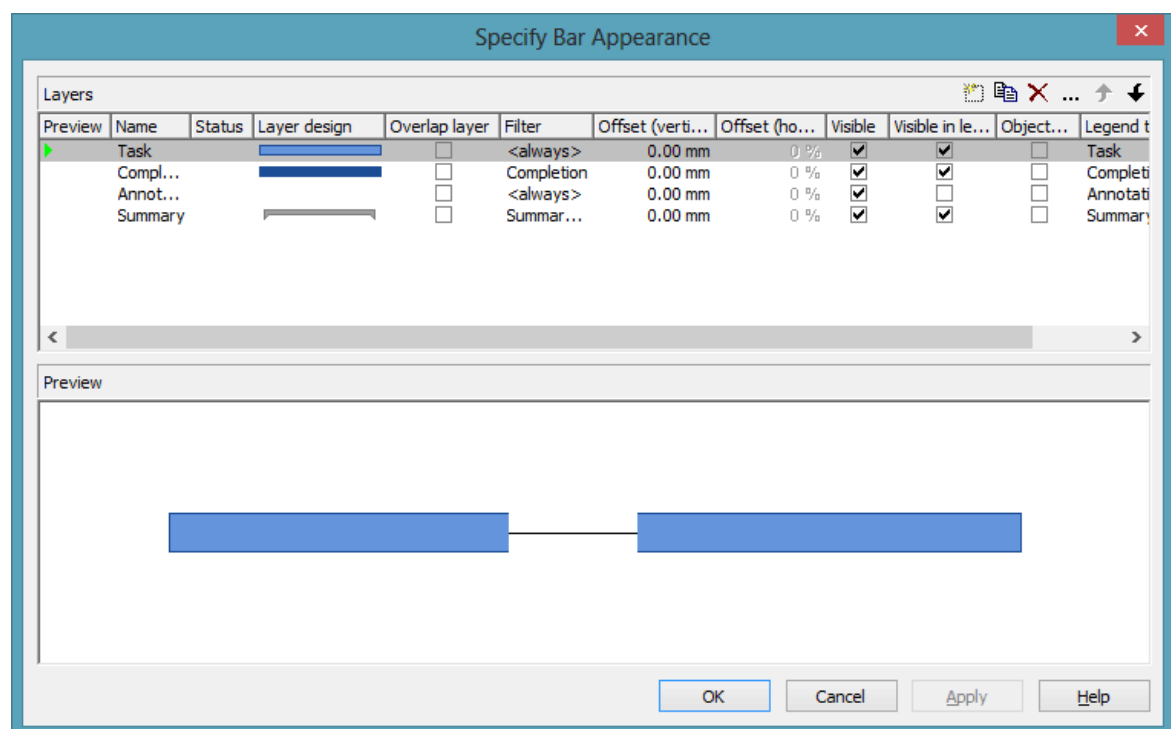
A layer is the graphical representation of a single date (symbol or bitmap layers) or of a pair of dates (rectangle, wedge-shaped or line layers).





Activities (or nodes) are graphically displayed by one or more layers. If an activity comprises several layers, the layers are graphically "stratified", starting with the layer of lowest priority and finishing with the layer of highest priority.


For each layer a filter is used. By the filter, only those layers are collected that comply with the criteria defined by the filter.

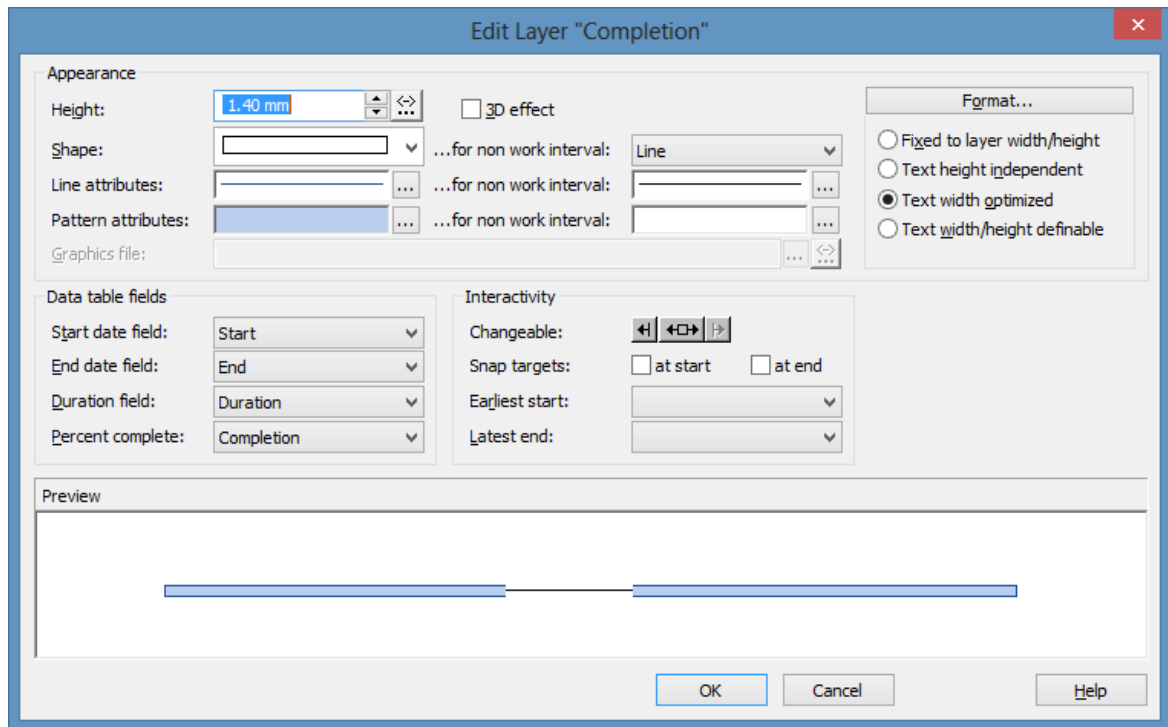
Layers can have different patterns, background colors, pattern colors and annotations. In addition, they can be of varying heights and may have a vertical and horizontal offsets. These options enable the layers of a node to differ from each other and to remain visible when displayed.

In the **Specify Bar Appearance** dialog box, you can define the layers of a node and specify their drawing priority.



By clicking on the buttons right-hand at the top of the layer you can add () , copy () , delete () or edit layers () .

To edit a layer, please select it from the list and click on the **Edit layer** button () or double-click on the **Layer graphics**. The **Edit Layer** dialog box will open and lets you edit the graphical attributes of the layer.



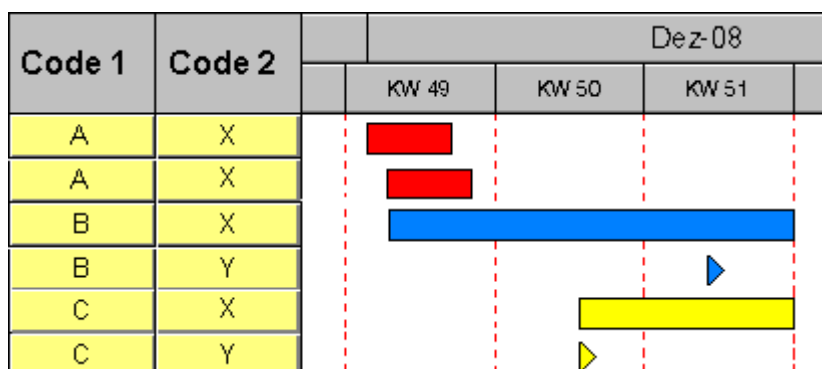
> Filter for a layer

By using filters, you can assign a layer to selected nodes, the selection of which depends on their data.

To define the conditions of a filter, in the **Specify Bar Appearance** dialog please click in the **Filter** field. Of the two buttons appearing, please click on the **...** button to open the **Administrate Filters** dialog box. You can reach the **Edit Filter** dialog by pressing **...** in the top right corner of the window.

(Also see "Important Concepts: Filters".)

In the example below, for a value of "X" in the field "Code2" a rectangle layer is defined and for a value of "Y" a symbol layer is defined. The colors are mapped using the field "Code1".



> Layer shapes

You can choose between rectangle layers, wedge-shaped layers, line layers, symbol layers, bitmap layers and invisible symbol layers.

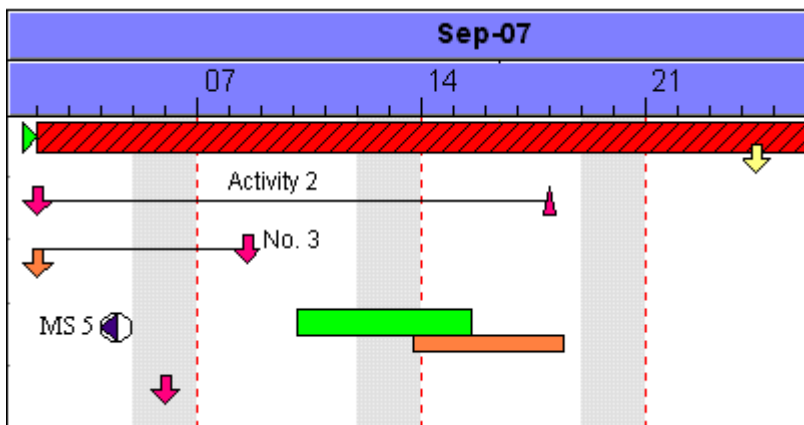
Select the layer shape from the **Shape** select box in the **Edit Layer** dialog box.

Symbol layers represent specific dates in time. There are some symbol layers that were predefined, but you can also define your own symbol layers. For example, you can use your company logo as a bitmap layer. You can specify the desired bitmap files in the **Graphics file** field.

Pairs of dates are visualized by rectangle, wedge-shaped or line layers. Wedge-shaped layers are useful for visualising increasing and decreasing activities, e. g. during the project start or end.

Of an invisible symbol layer, only its annotation is visible, and these layers will not be displayed in the legend.

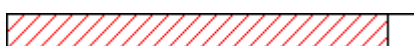
Combining layer forms, patterns, colors and filters, a large variety of different layers can be defined. The below picture shows some examples:



> Degree of completion

VARCHARTXGantt allows to display the degree of completion of an activity. For this, please

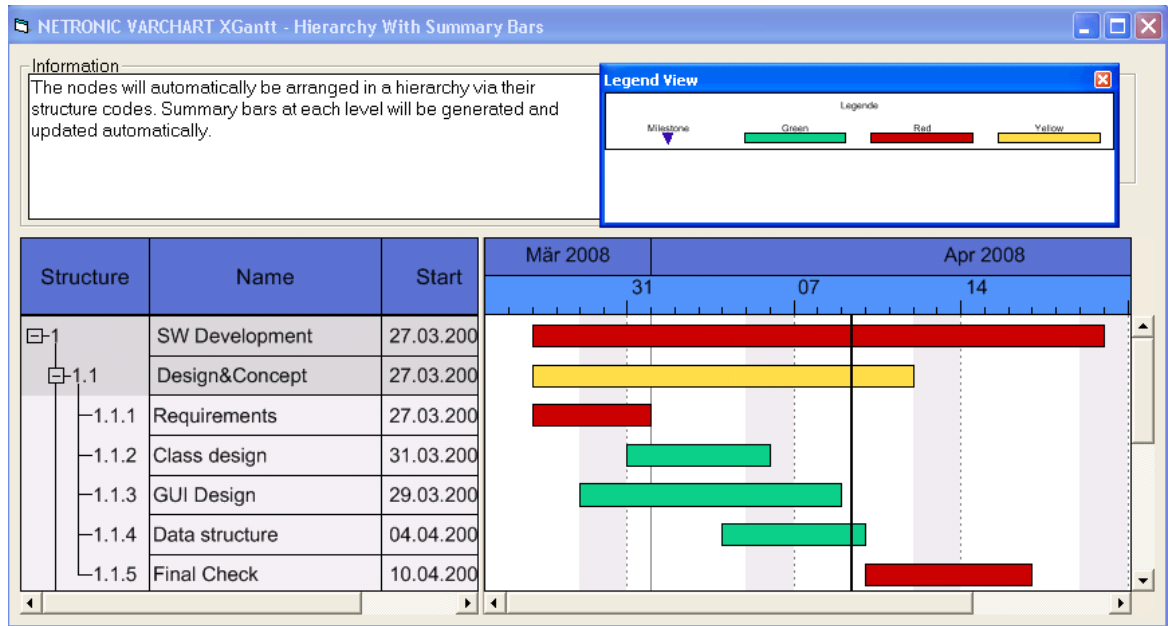
create a layer named "Completed" and edit it by using the **Edit Layer** dialog box. For wedge-shaped and rectangle layers you can select the data field that contains the percentage degree of completion of the selected layer. So, for the layer "Completed" please select the data field "% completed". Now specify the graphical attributes (color, pattern etc.) so that the "Completed" layer can be easily recognized.



Degree of completion: 90 %

3.16 Legend View

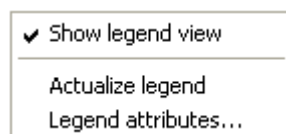
The legend view is an additional window that lets you display a legend on the screen. The layout of the legend can be specified with the legend attributes of **VcBorderBox** or in the dialog **Legend attributes** which can be reached from the **Border area** property page



At runtime, you can switch on and off the legend view in the default context menu by the menu item **Show legend view**.



Moreover, you can switch on or off the legend view in the legend's context menu.



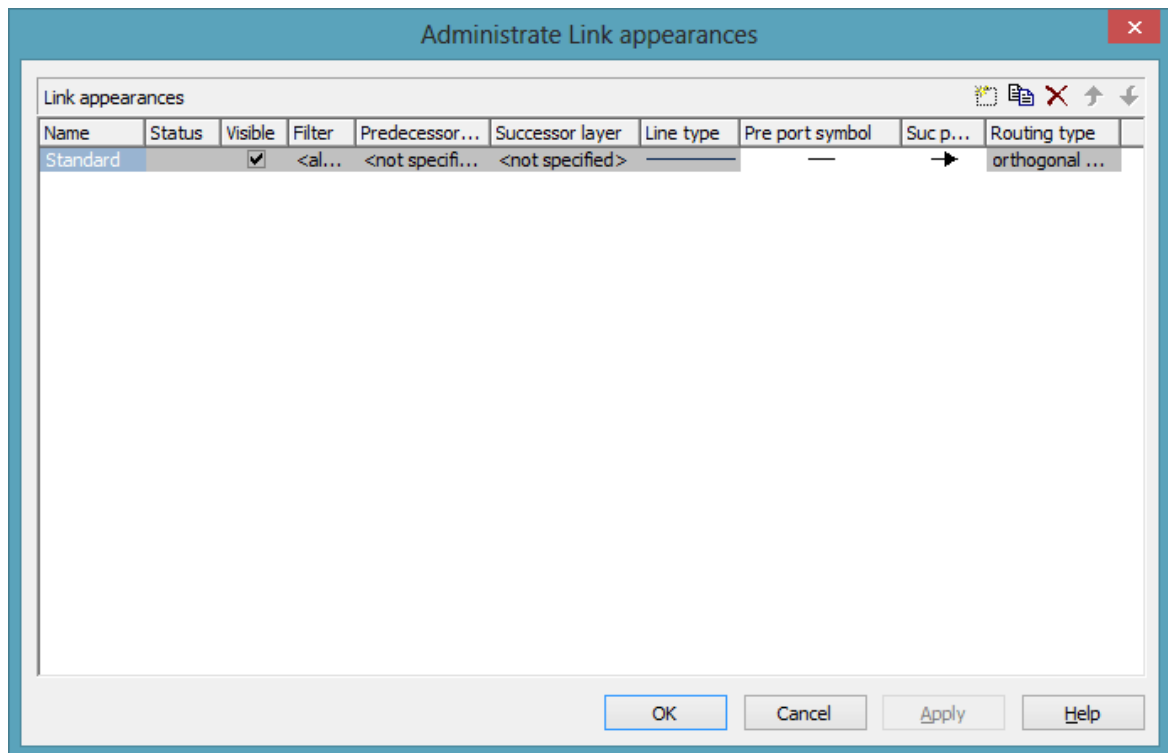
The context menu offers two more items: **Actualize legend** and **Legend attributes**. By selecting the latter you call the corresponding dialog.

The refreshing of the legend is needed after modifications in the chart, such as adding or deleting nodes, because they are not displayed automatically. The refreshing can also be carried out by switching off and on the legend view. This concerns the loading of nodes as well. If on the property page **Additional views** the attribute **Initially visible** was selected for the legend view and no nodes have been loaded when running the program, the legend stays empty until it was refreshed.

On the **Additional Views** property page you can set the properties of the Legend View. For details please see **The Additional Views Property Page** in the chapter **Property Pages and Dialog Boxes** .

The properties of the Legend View can also be set by the API property **VcGantt.VcLegendView**.

3.17 Link Appearance



You can define different link appearances in the **Administrate Link appearances** dialog. The link appearances will be assigned to the links dynamically by filters.

> Further Specifications for the Link Appearances

Fur further information about link appearances please see chapter 4.28 "The Administrate Link Appearances Dialog Box".

3.18 Links

A link is defined by a record of the data table which contains the link data. Link data are automatically and simultaneously generated on the generation of nodes. Link data can be loaded from a file via the API or they can be generated interactively by the user.

> Generating Links

During run time, you can use the mouse to draw links between two activities if the **Mode: Create Link** was activated before.



The link is drawn between the first layer of the activity where the link starts and the first layer of the activity where the link ends.

You can as well generate links using the API by the method **InsertLinkRecord**.

Interactive generating of links is reported to the application by the **VcLinkCreating** event.

> Deleting Links

You can delete a link by clicking on it with the right mouse button to pop up the context menu and by selecting the menu item **Delete**. Beside, you can delete links by the method **VcGantt.DeleteLinkRecord** or by the method **VcLink.DeleteLink**.

> Events

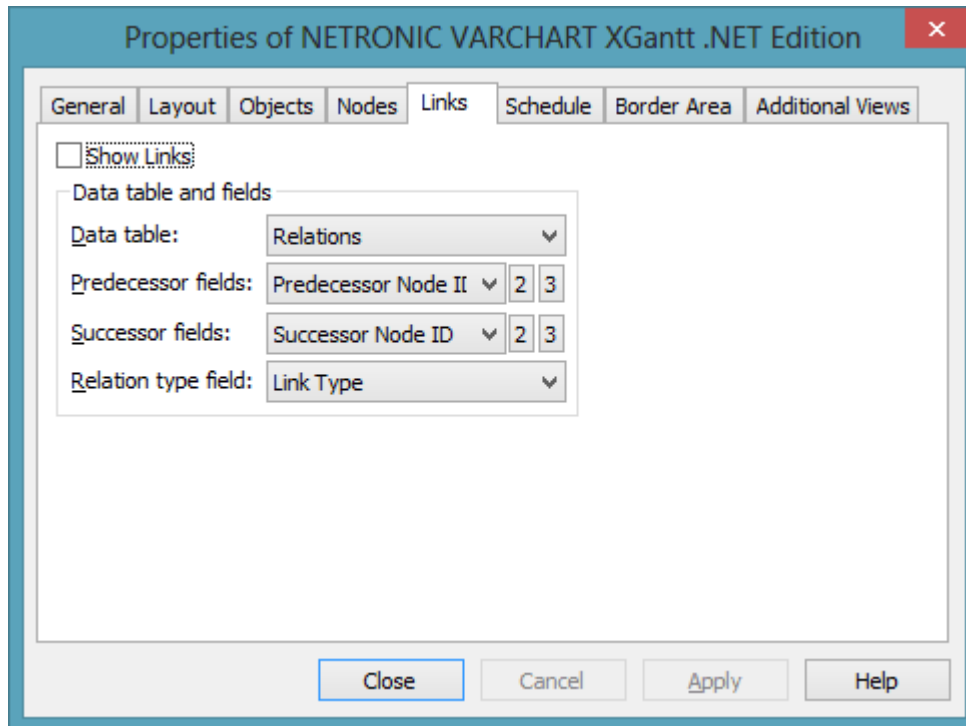
You can react to the below events:

- **VcLinkCreating**
- **VcLinkCreated**
- **VcLinkDeleting**
- **VcLinkDeleted**
- **VcLinksLeftClicking**
- **VcLinksLeftDoubleClicking**
- **VcLinksRightClicking**

> Specifying Links

On the **Links** property page you can choose whether the links are to be displayed, and, if desired, set more options.

Furthermore you can define link appearances in the dialog **Administrate Link Appearances**. For each one you can select a filter, set the predecessor / successor layer, choose a line type, the predecessor / successor port symbols and the routing type.



> Specifying Links

You can specify data fields in which the identifications of the predecessor/successor nodes and the relation types are to be stored. If the identification of a predecessor or successor node consists of more than one field, the corresponding link has to match this identification. That means that according to the ID of the respective node, a second or third field has to be selected if necessary. The first field is displayed by default. For setting a second or third field, click on the corresponding button and select the desired field from the drop-down list

> Types of Links

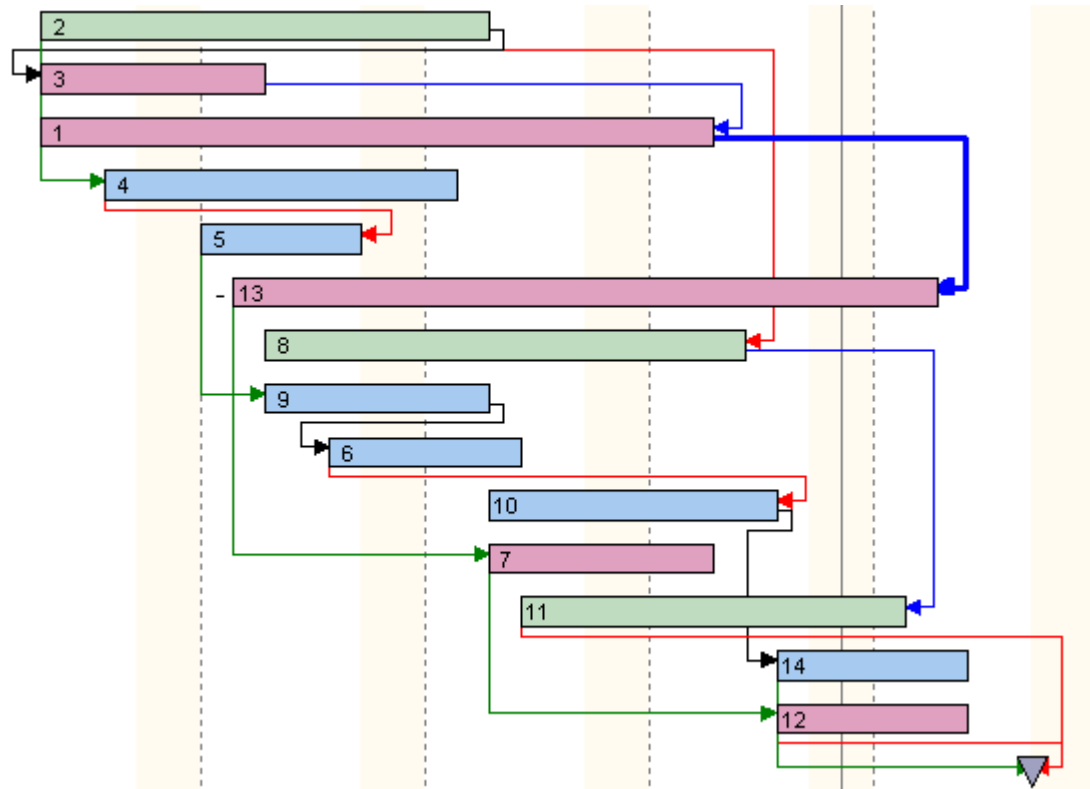
In the combo box **Relation type field** you can select a data field that the link type is to be loaded from.

Link types:

- FF: Finish-Finish

- FS: Finish-Start
- SF: Start-Finish
- SS: Start-Start

This data field enables the link type to be visualized by the appropriate line routing.



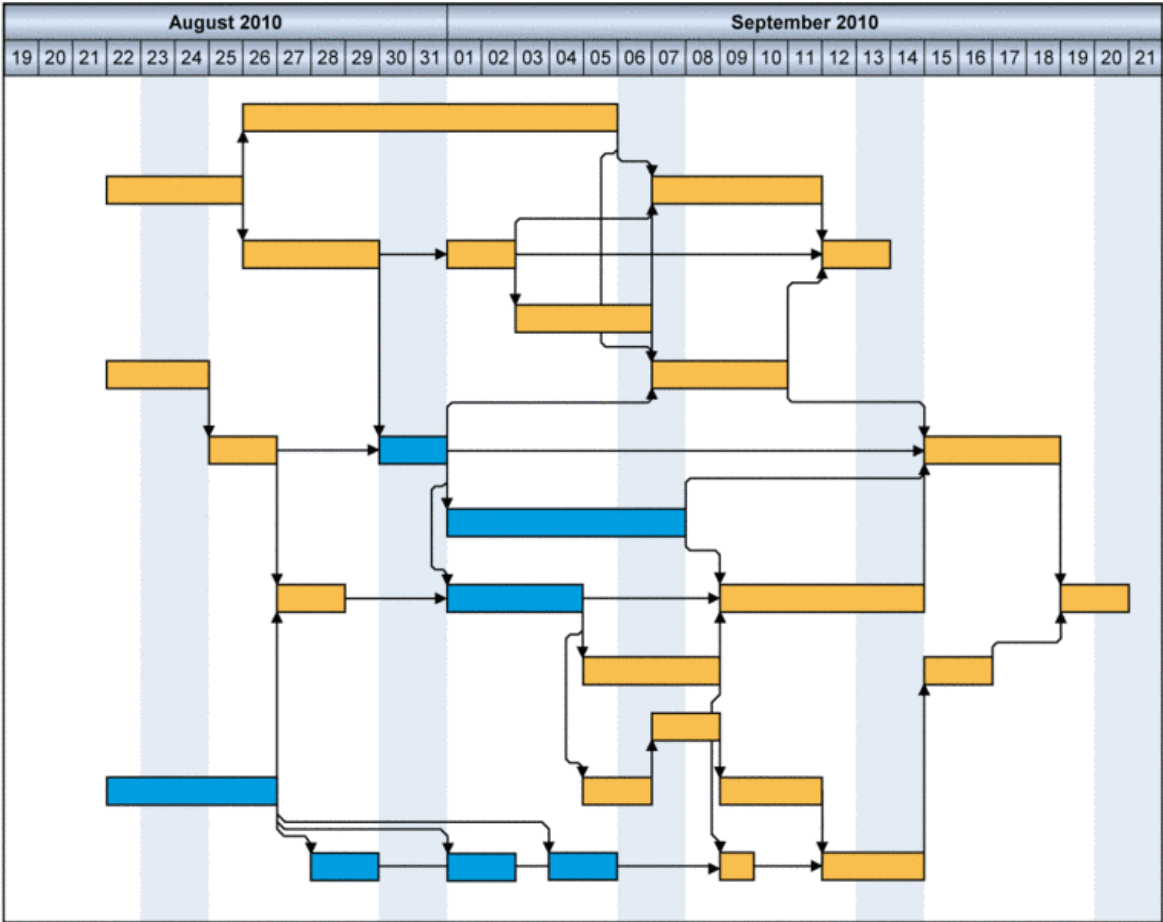
Example visualizing different link types

> Automated Layout

For the link routing a layouter is available to automatically display links in their optimum position. It can nest elbows so that line cross-overs are reduced to a minimum. The link routing is always unambiguous and allows the user to clearly distinguish where a link comes from and where it leads to.

The row heights in Gantt charts automatically adapt in order to create the required space to display all parallel horizontal link sections in a row.

Little slants are drawn in each elbow to indicate the direction into which the link is going.

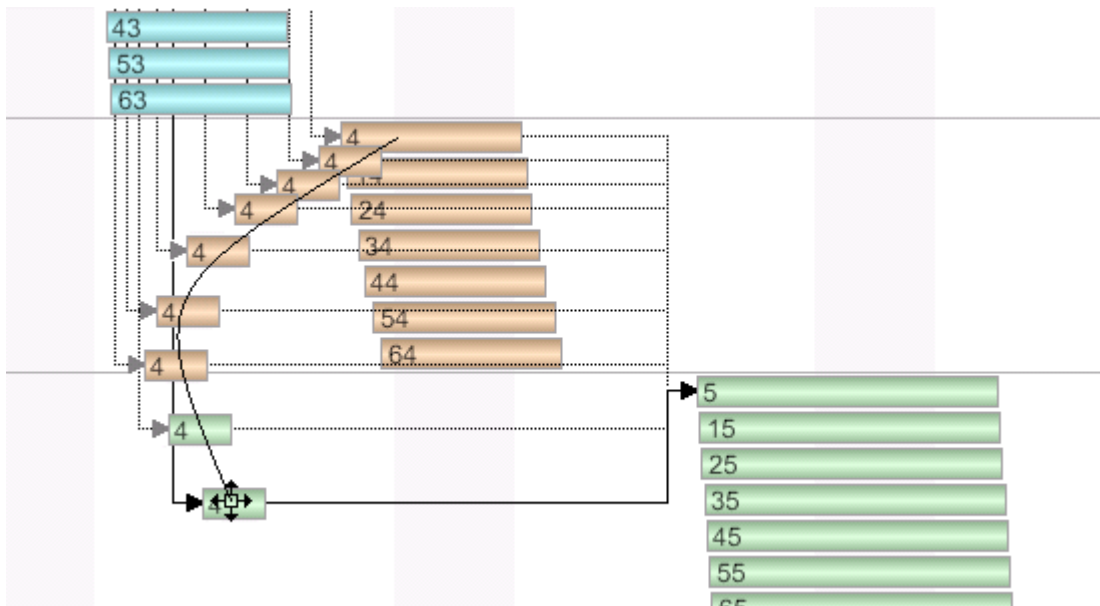


3.19 Live Update

What is Live Update?

With the Live Update, being available from XGantt Version 5 onward, the consequences of a mouse interaction are visualized immediately during the action and not only after ending it.

Up to version 5, VARCHART XGantt used phantoms and the consequences for the overall planning were indicated by the Gantt graph as soon as the dragging action was finished by releasing the mouse key. The live update function, however, lets the planner recognize the results of the mouse action while interacting, since every mouse movement results in updating the node, meaning that the modifications are repeated constantly on the object thus resulting in a live update of the object and the chart. At any point during dragging a visualization of the node matching the respective cursor position with the appending links is shown.



Two Ways of Modifying Data

There are two ways of changing and evaluating data:

- Modifications only relating to the particular object such as simple data changes, called **individual** changes in the following. Individual changes occur during each interaction.
- Modifications that do not only affect the particular object but also result in changing complete structures, such as grouping or optimizing, called **structural** changes in the following.

Structural changes can currently only occur while shifting nodes or groups, since only these can be summed up and arranged in structures.

Structural changes are carried out timer-driven (see also below: **Timer-driven Live Update**). **OldNode** and **PreviewNode** are not planned.

After a structural change, the cursor is automatically scrolled under the cursor again (node tracking).

Interactions affected by Live Update

The interactions affected by live update are: shifting of nodes and groups and interactively creating nodes and links.

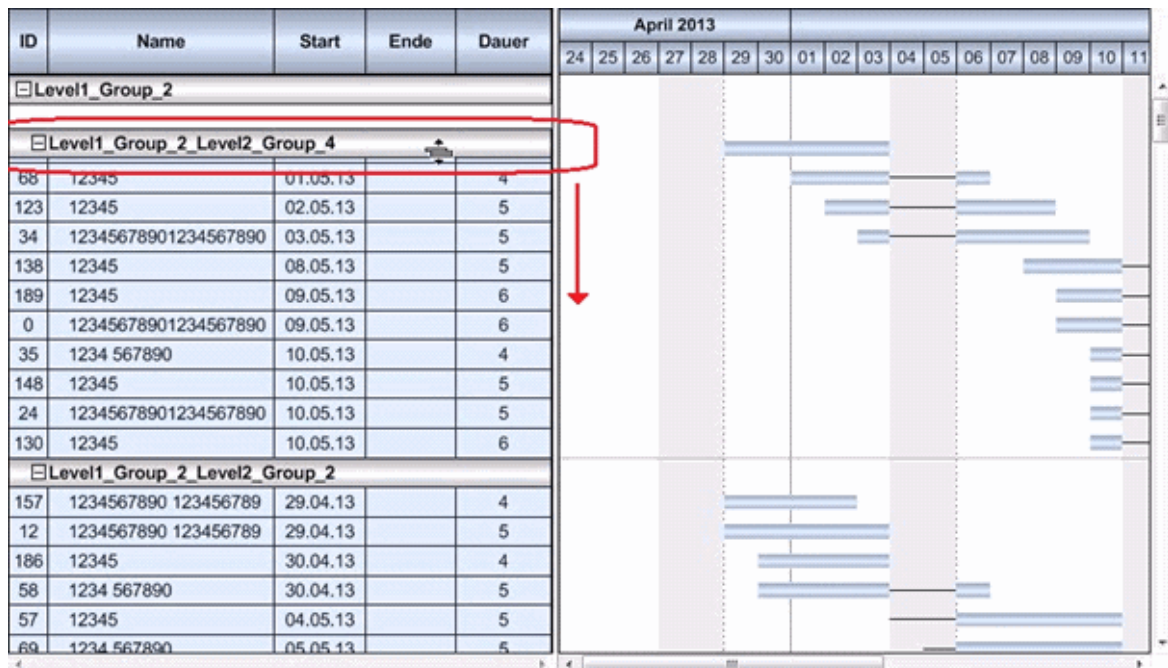
> Shifting of nodes and links in the diagram

Nodes and links can be freely moved in terms of optic, the horizontal and vertical position of the node being always adjusted to the cursor position, thus being always under the mouse cursor. Appending links, being drawn with linkrouting <orthogonal> or <straight> are dragged along accordingly. The linkrouting <distinguish> doesn't work in this case, so <orthogonal> is used. While changing the position, the visualization of the nodes and links is also constantly updated, meaning that filters and mapping are applied to the complete construct. An empty area will remain at the former node position, reinforcing the dragging effect. The node is dragged away from his former position. For this, the node with its links is set to **VC_VISIBILITY=VC_NO** and copies of nodes and links are made and updated while dragging.

> Shifting of Groups

In VARCHART XGantt groups can be moved interactively within their levels. This is done by either shifting the summary bar or the group node vertically in the diagram or by vertically moving the respective table format in the table. This structure modification equals a manual sorting, having no equivalent in terms of data, hence no data are modified. After the modification will be done, the shifted summary bar/group node or the shifted table format respectively will be scrolled back under the cursor again automatically, this scroll behavior being called group tracking here.

In the diagram area, a VARCHART node phantom with real presentation of the summary bar/group node is used and in the table area a VARCHART node phantom with real representation of the table box. The real representation will remain unchanged since there will be no data modification during the dragging interaction.



Timer-Driven Live Update

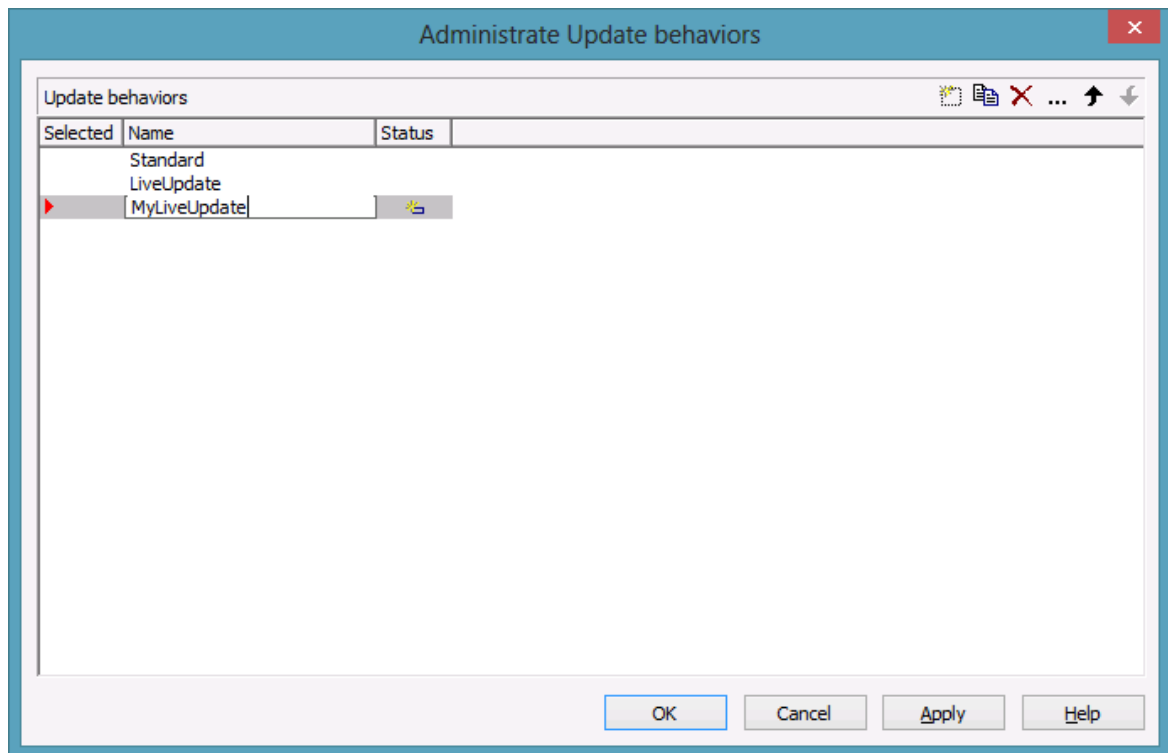
The whole chart gets quite unsteady by the constant (sometimes comprehensive) visual changes and the immediate changing of status without animation options could be confusing if not disturbing so that an alternative for the immediate change of status is called for. Updating caused by structural changes should not be constant but timer-driven. If the user shortly pauses during the mouse interaction, the structural modification will be only carried out after a short, but significant waiting time and the chart be updated. The graphic shown always matches the respective cursor position. Now the user can continue interacting since he is still moving the mouse while holding the key pressed. The structural changes are again impended until the user pauses again and again they will be only carried out and the chart be updated after a short, but significant waiting time. This is repeated until the interaction ends (releasing the mouse key). This technique ensures that the chart will remain rather steady.

Setting up Live Update in VARCHART XGantt

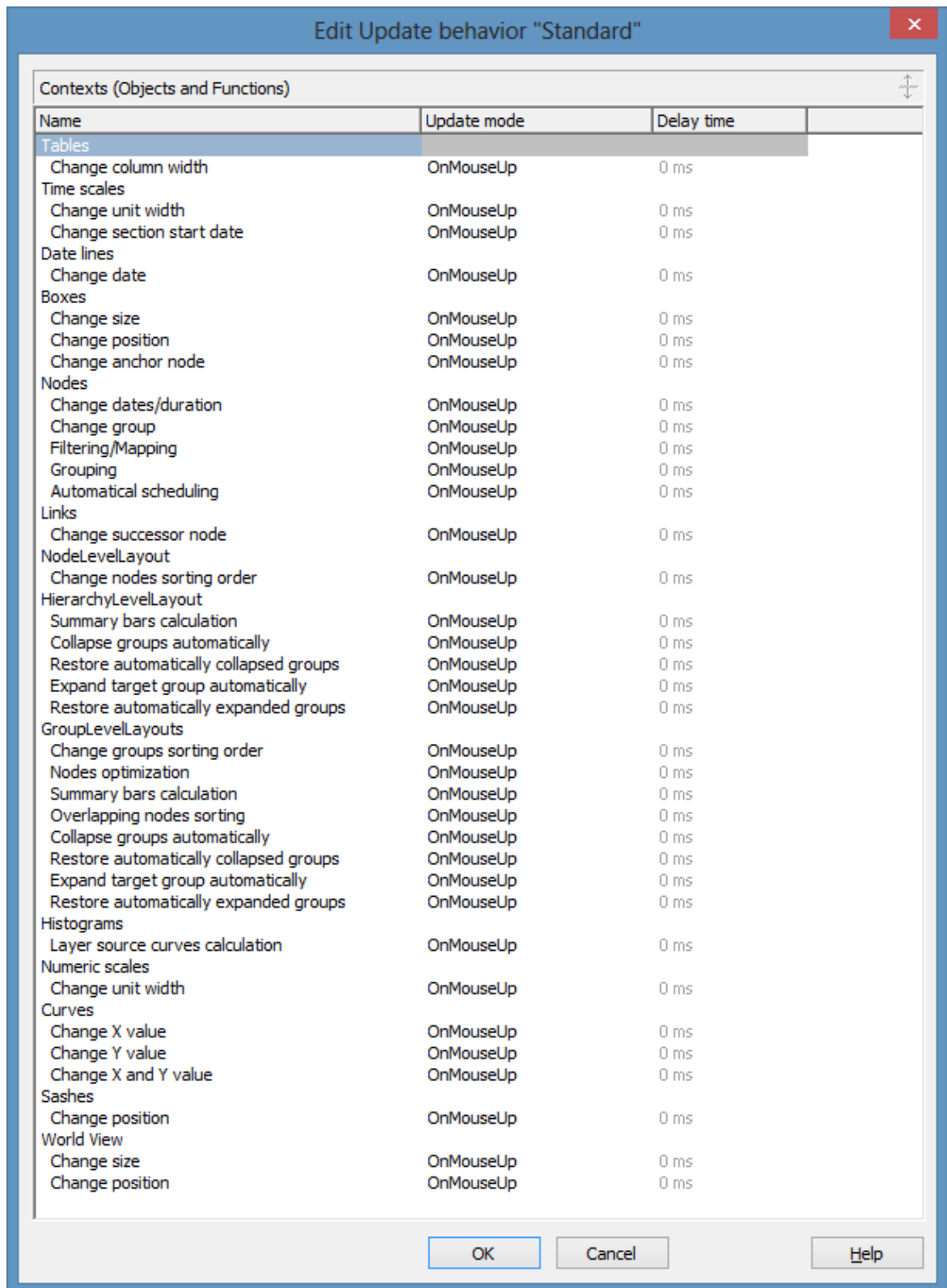
> At Design time

The live update settings can be made in the **Administrate Update Behavior** and the **Edit Update Behavior** dialogs at design time. VARCHART XGantt comes ahead with the update behaviors **Standard** and **Live Update** the settings of which can **not** be customized by the user.

The user can, however, create individual update behaviors that can be customized at will in the two dialogs shown below.



174 Important Concepts: Live Update



Note: Please note that individual update behaviors for data driven objects (nodes, links and groups) can **only** be assigned by API.

> **At runtime**

The settings are made in the following objects:

- VcBox
- VcCurve
- VcDateLine
- VcGantt
- VcGroup
- VcLinks
- VcNode
- VcNumericScale
- VcTable
- VcTimeScale
- VcUpdateBehavior
- VcUpdateBehaviorCollection
- VcUpdateBehaviorContext
- VcWorldView

For further information see the API reference of this manual.

3.20 Localization of Text Output

The **VcTextEntrySupplying** event allows to replace all items in context menus, dialog boxes, info boxes, error messages, the names of the months and days that appear during runtime in order to, for example, translate them into a different language.

To do so, activate the check box **VcTextEntrySupplying events** on the **General** property page. Or set the property **TextEntrySupplyingEvent-Enabled** to **True** to activate the event.

Example Code VB.NET

```
VcGantt1.TextEntrySupplyingEventEnabled = True
```

Example Code C#

```
vcGantt1.TextEntrySupplyingEventEnabled = true;
```

Then capture the **VcTextEntrySupplying** event and specify the text you want to have appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcTextEntrySupplying(ByVal sender As Object, ByVal
e As NETRONIC.XGantt.VcTextEntrySupplyingEventArgs) Handles
VcGantt1.VcTextEntrySupplying

    Select Case e.ControlIndex
        Case VcTextEntryIndex.vcTXERibCW
            e.Text = "CW"
        Case VcTextEntryIndex.vcTXERibDay0
            e.Text = "Mo"
        Case VcTextEntryIndex.vcTXERibMon8
            e.Text = "September"
        Case VcTextEntryIndex.vcTXERibQuar3
            e.Text = "Quarter 3"
    End Select
End Sub
```

Example Code C#

```
private void VcGantt1_VcTextEntrySupplying(object sender,
NETRONIC.XGantt.VcTextEntrySupplyingEventArgs e)
{
    switch(e.ControlIndex)
    {
        case VcTextEntryIndex.vcTXERibCW:
            e.Text = "CW";
            break;
        case VcTextEntryIndex.vcTXERibDay0:
            e.Text = "Mo";
            break;
        case VcTextEntryIndex.vcTXERibMon8:
            e.Text = "September";
    }
}
```

```
        break;
    case VcTextEntryIndex.vcTXERibQuar3:
        e.Text = "Quarter 3";
        break;
    }
}
```

3.21 Maps

Maps are used to set certain properties in dependence on data, thus avoiding to define large numbers of filters.

By using maps you can for example assign background colors, patterns, pattern colors and more properties to layers in dependence on their data.

Maps consist of a list of mappings. Each mapping consists of a key and a value. Depending of the map type, the value can be a graphics file name, a pattern etc. The key is a possible entry in a data field. At runtime, the keys are compared to the actual contents of the addressed data field and if they match, the value for the addressed graphic property is applied.

If there are more than two columns, more than one value is assigned to one key.


Example: In the map, the key "A" is assigned to the value "green". If the map is applied and some node field contains the value "A", the color green is assigned to this node (as background color of its layer, for instance). As a second value, a legend text could be assigned saying "finishes in time".

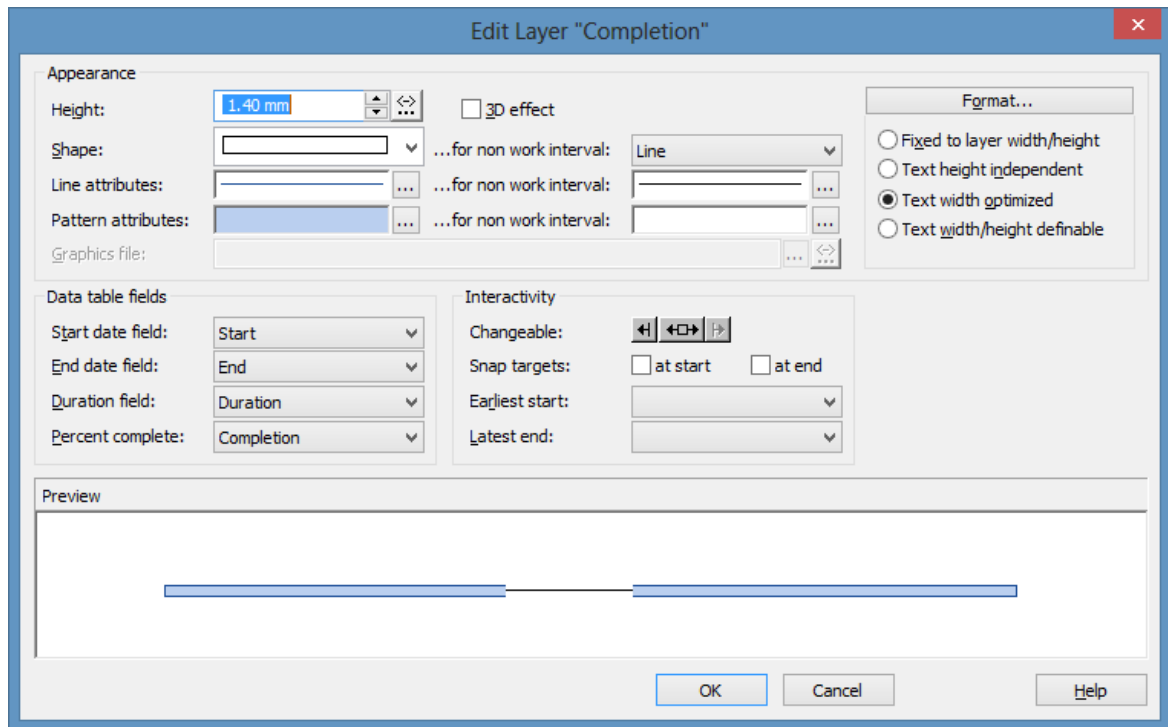
So, as a basic principle, the field values are compared to the keys of the map. If they match, the map value(s) are used.

By using filters instead of keys you can specify more complex mappings. Basically, the concrete keys are interpreted first and only if they do not apply, the filters are interpreted.

> Example: Background color of layers

In the below example, the background color of a layer will be assigned in dependence on the node data by using a map.

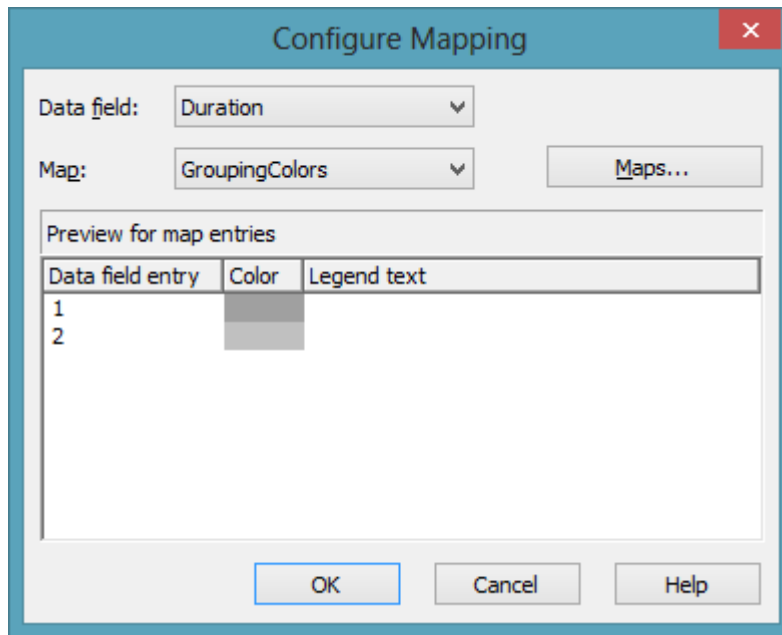
In the **Edit Layer** dialog box, please click on the () button near the **Background color** field.



You will get to the **Configure Mapping** dialog.

> **Configuring Mapping**

The **Configure Mapping** dialog lets you assign a data field of a node to a map, so that the value in the data field can be compared to the keys of the map. Thus the desired property, in our example the background color of the layer, is specified data- dependent. If the attribute shall not be dependent on only one single value but on a range of values, you can create a filter for this range of values which you select in the **Edit Map** dialog instead of a single value. This filter will then be displayed in the **Configure Mapping** dialog in the list of the data field entries.

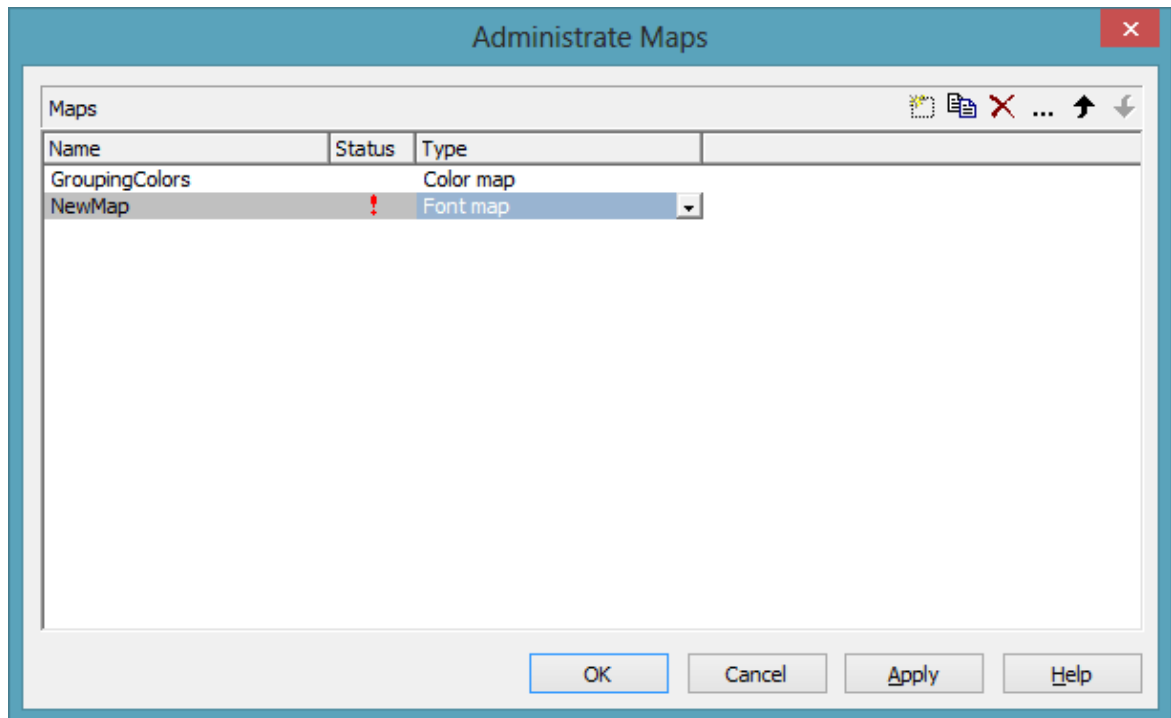


To configure a mapping, please select a node **Data field** at the top of the dialog, the values of which shall be compared to the key values of the map. From the field below, select an appropriate map **Map**. (Only those maps are selectable which match the attribute selected in the **Edit Layer** dialog. Because in our example you have selected the background color, only maps of the type "Color map" are displayed). After having selected the map, its contents become visible in the preview of the dialog. If there isn't a map to select, please create one as described in the chapter below.


> Administration of Maps

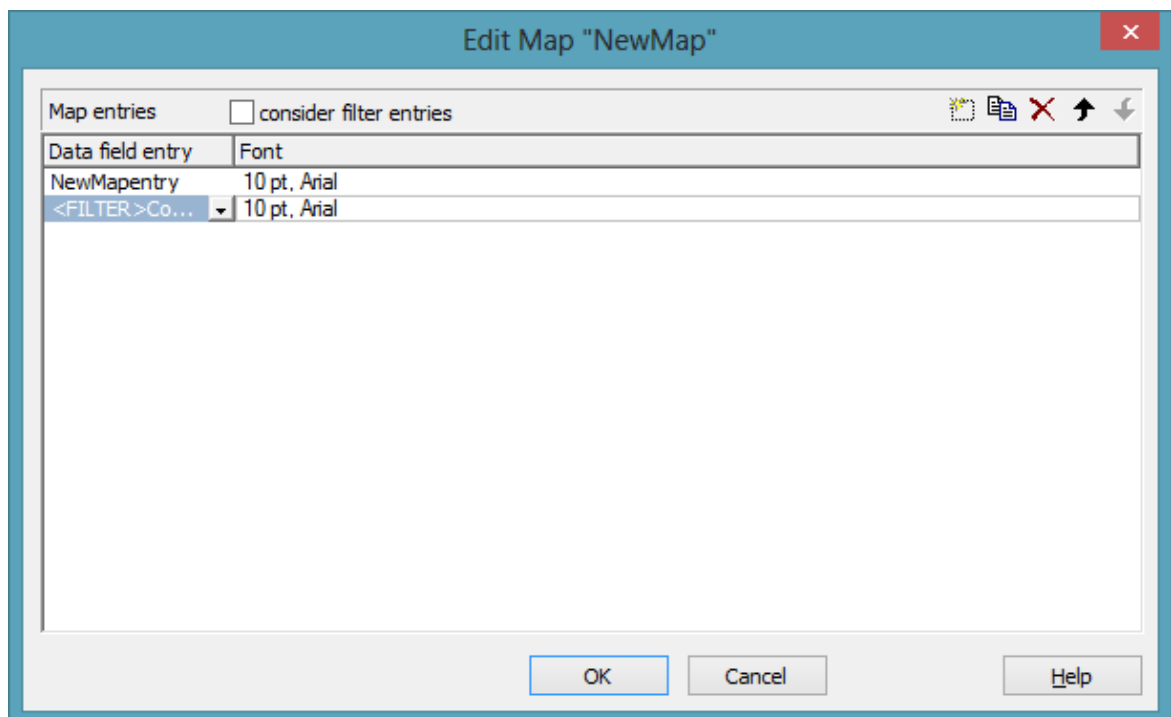
In the **Administrate Maps** dialog which can be invoked by clicking the **Maps** button or by clicking the **Maps** button of the **Objects** property page, you can modify the name and the type of a map by directly entering the corresponding data fields. By clicking the corresponding buttons on the right at the top of the window, you can also create, copy, edit or delete maps.

You can choose between different types of maps, according to whether colors, patterns, graphic files, fonts, lengths or numbers are to be allocated to data field contents.



> Editing Maps

To edit a map, mark it in the table and click on the  button above the table. The **Edit Map** dialog box will open.



Of each key (=data field entry), the table shows its corresponding values, which, depending on the map type, in our example are the color and the legend text assigned.

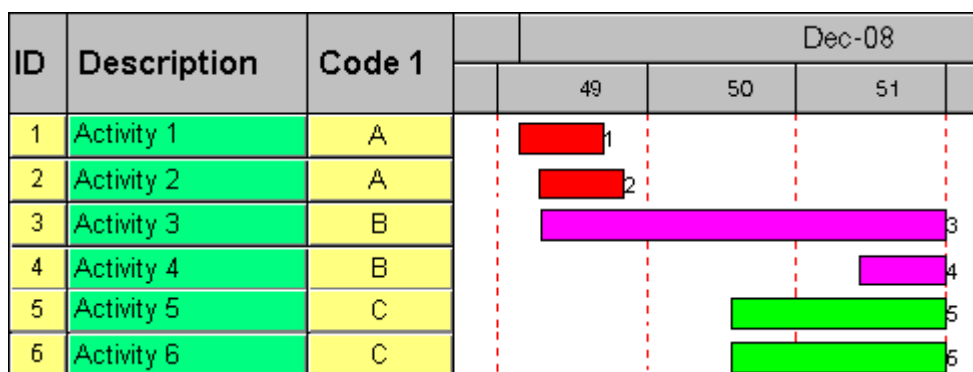
By the buttons right-hand at the top you can create, copy or delete keys (map entries) or modify their position in the table.

If you have ticked the check box **consider filter entries** not only the single values from the list of data field entries are considered as keys but also the filters which can be selected from the drop down list. Thus you can not only specify a single value as key but also more complex criteria.

In a map you can create 150 map entries at maximum. If you need more map entries, please create a new map, e. g. as a copy of the one being edited.

> Example

The below example shows a layer where the activities of the field value = "A" are displayed in red, the activities of the field value = "B" are displayed in pink, etc. The default background color is gray. The latter is used for activities that have no data field value or that have filed value which is not defined in the map.



For further details please read the chapters "Property Pages and Dialog Boxes".

> Adjusting the Map during Runtime

You can adjust the map during runtime using VcMap methods, which lets the user modify your default settings via a dialog designed by yourself.

3.22 MultiState Fields

What are MultiState Fields?

It is possible in the table section to display different contents of data fields as different graphics by using maps and graphical fields. MultiState fields are an enhancement of this principle, where a click on a picture results in a change of state of the associated data field. MultiState fields are a comfortable way to edit data fields that can adopt a final number of different states. This is why multiState fields can only work if the module **Data Editing** was licensed.

> The Way they Work

A click on the field triggers the search for the next picture in the map that differs from the present one. The corresponding value (i.e. the key in the map) will be assigned to the data field. If, apart from the map, another graphics file was set as a default, it will also be considered when the map is searched through. If the default picture appears, an empty string will be set to the data field. In other respects the default picture will appear, if in the data field a value occurs that does not equal a key in map.

A most simple application of multiState fields are boolean data fields, which, for example, display the values **true** and **false** by check boxes that show or do not show a check. When clicking on the present state, the picture will change to the opposite state and the value of the corresponding data field will turn from **true** to **false** (or vice versa).

> Instructions for Programming

- Keys in the map that point to the same graphics file should be placed consecutively. This is the only way to have the same graphics file displayed just once when the map is searched through. This is because on a click, the next picture file will be selected which is different to the picture presently displayed. For example, you can link the keys **true**, **t** and **True** to the same graphics file. If the file is displayed, a different file will be displayed on the subsequent click. So displaying the same graphics file for three times is avoided.
- For the same reason, you should put all keys at the beginning of the map, that point to a graphics file equal to the default graphics file.
- If the same graphics file consecutively appears in the map, the value written to the data field will always be the first key. If **true**, **t** and **True**

were put consecutively in the map (pointing to the same graphics file), always **true** will be stored to the data field, but never **t** or **True**.

- MultiState fields only change their state if editing is allowed (see the corresponding VcGantt properties **InPlaceEditingOnGroupsInDiagramEnabled**, **InPlaceEditingOnGroupsInTableEnabled**, **InPlaceEditingOnNodesInDiagramEnabled**, **InPlaceEditingOnNodesInTableEnabled**).
- To avoid the pictures to be displayed in different sizes, the height of a graphics field should be set to a value unequal to 0 mm (see dialog **Edit table format** in the VARCHART XGantt property pages).

For more information on graphics files and maps please read the chapters **The "Edit Table Format" Dialog Box** and **Maps** in the User's Guide and the documentation of the VcGantt property **FilePath** in the Reference Manual.

3.23 Node (Activity)

A node (activity) represents a record of the Maindata table. Nodes can either be loaded by calls of the programming interface (API) or interactively created by the user.

> Creating Nodes

On the **Nodes** property page you can specify whether the user

- can create new nodes by dragging the mouse (in the **Mode: Create Node**) (**Node creation allowed**)
- can create new nodes by double-clicking (**Node creation via double-click**),
- can directly edit new nodes via the **Edit Data** dialog box (**Node creation with dialog**).

At runtime, when a new node is created by dragging the mouse, the **Create Activity** box appears that indicates the start/end date and the duration of the new node.

Create Activity	
Start:	07.09.2007
End:	09.09.2007
Duration:	2 days

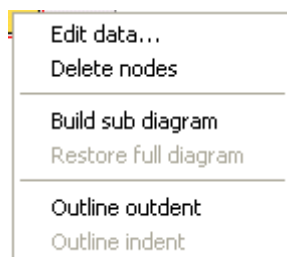
As soon as you release the mouse button, the **Edit Data** dialog box will appear, if the checkbox **Node creation with dialog** on the **Nodes** property page was ticked before. The box shows the data of the interactively created node that you can edit now.

You also can create nodes via the API by the method **InsertNodeRecord**.

When a node is created interactively, this is reported to the application by the event **VcNodeCreating** or **VcNodeCreated**.

> Deleting Nodes

To delete a node at runtime, position the cursor on the node to be deleted and press the right mouse button. The below context menu will appear:



Select the **Delete Nodes** option.

When a node is deleted interactively, this is reported to the application by the event **VcNodeDeleting**.

You also can delete nodes via API by the VcGantt method **DeleteNodeRecord**.

> **Further Settings to Nodes**

Beside, you can set on the **Nodes** property page:

- The data fields that the data of start, finish, and duration of interactively created nodes are to be stored to.
- Whether workfree periods are to be highlighted. In rectangle layers this will be indicated by a solid line.
- Whether calendars are to be assigned to the nodes. The influence of calendars becomes visible when nodes are moved and when durations are calculated. When moving activities, their start and finish dates will not be placed on workfree days. When calculating durations, workfree periods will be taken into account. By default, a five-days calendar ("WeekCalendar") is defined.
- If a calendar is required for an individual node, you can set a data field to store the name of the calendar.
- Whether a user is enabled to move several marked nodes at a time.
- Whether a marked node is enabled to be moved as a whole, that is, with all its layers.

> **Events**

You can react to the below events:

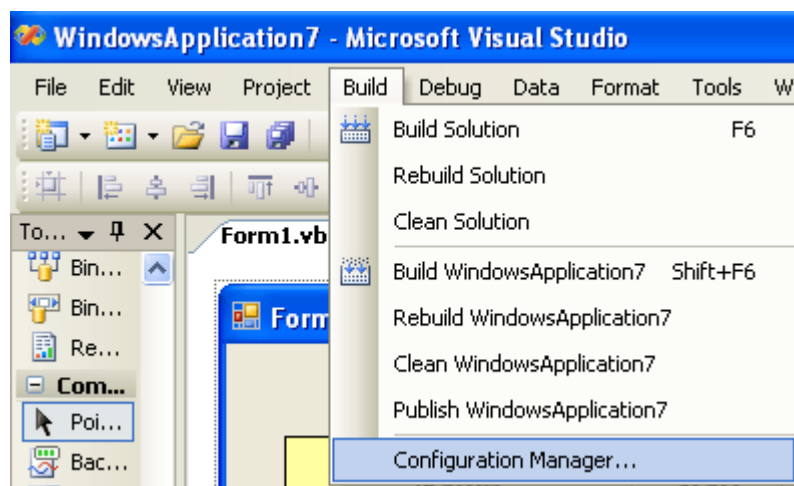
- VcNodeCreating
- VcNodeCreated
- VcNodeDeleting
- VcNodeLeftClicking
- VcNodeLeftDoubleClicking
- VcNodeModified
- VcNodeRightClicking
- VcNodesMarked
- VcNodesMarking

3.24 Platforms x86 and x64

Applications written with the .NET framework are usually compiled into MSIL, a processor-independent bytecode. On starting the application, MSIL is translated into a machine code understood by the respective computer's processor and run in its full speed. Applications in MSIL can hence be run on any processor under windows as long as no components (assemblies or dlls) in pure machine code are used. They can even be run on other operating systems such as Mono with Linux as long as no operating system-dependent components are used. If an application does not fulfill the conditions for the processor-independence it should be marked accordingly. Otherwise it might be started by mistake on an unsupported processor, thus causing more or less understandable error messages when a processor- or operating system-independent component is used for the first time.

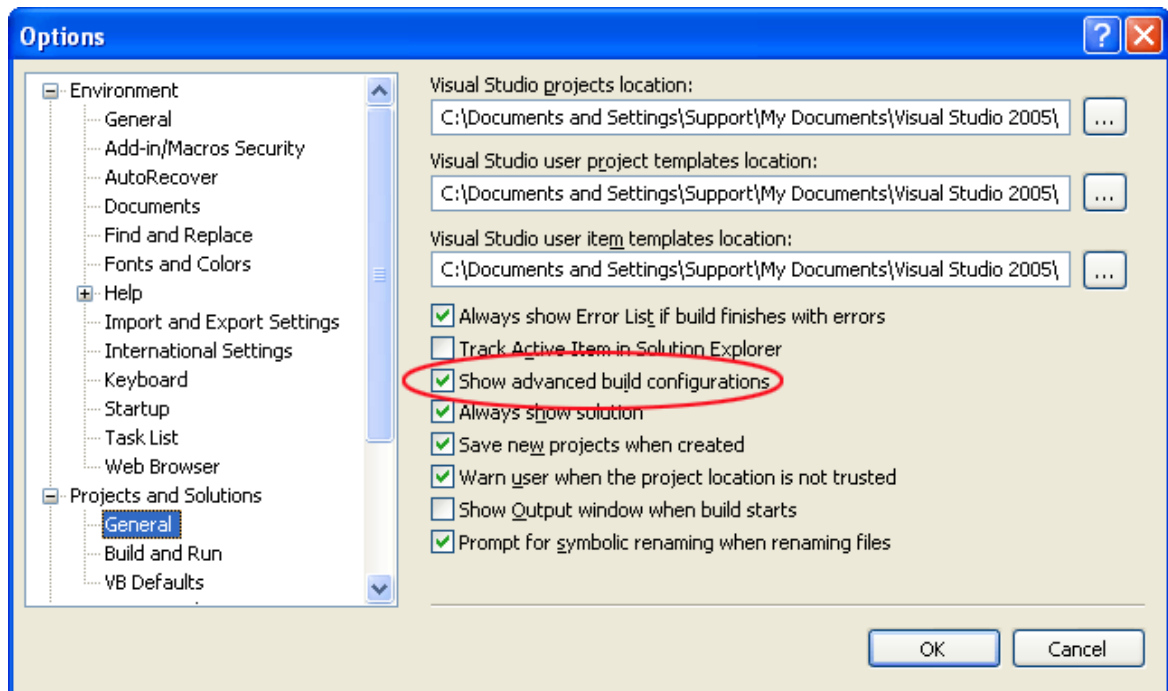
Internally VARCHART XGantt is in part written in pure machine code, called **Mixed Mode** under .NET so that XGantt has to be translated anew for each processor it is used with. Versions are available for x86 processors and from version 4.3 on also for x64 processors.

Applications that use VARCHART XGantt are hence not processor-independent. As this is not recognized automatically by Visual Studio in the versions 2005 to 2010, the processor has to be set manually in a project or a solution. This is done in the **Configuration manager** dialog which you can open by clicking **Build/Configuration Manager**.

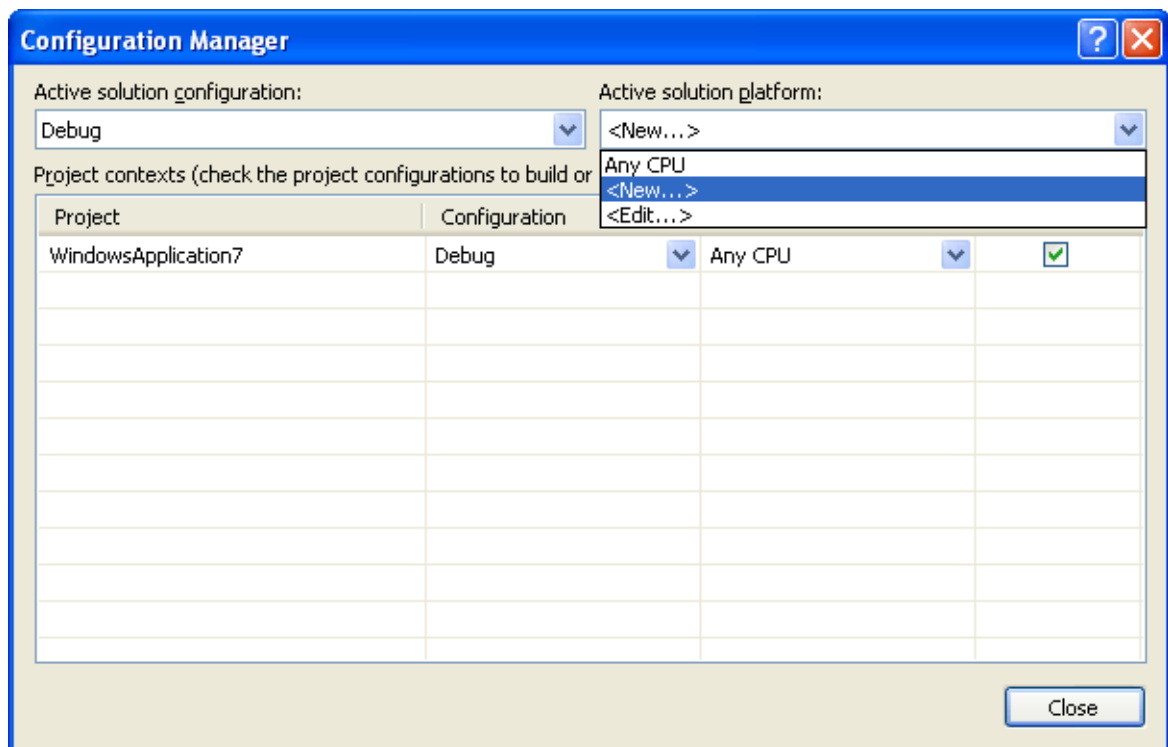


If this menu item is not visible you have to tick the option **Show advanced build configurations** in the dialog **Tools/Options.../Projects and Solutions/General** first.

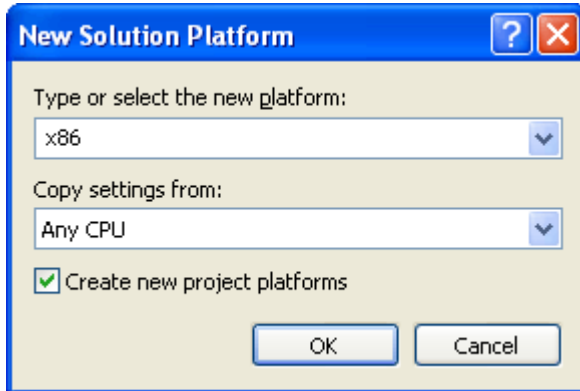
188 Important Concepts: Platforms x86 and x64



In the configuration manager you can create or delete platforms. To create a new one, select **<New...>** in the **Active solution platform** dropdown list.



In the corresponding dialog you can create the desired platforms **x86** or **x64**:



If you want to delete a platform click **<Edit...>** in the **platform** drop-down list and in the following dialog select the desired platform and delete it by clicking **Remove**.

To make sure that Visual Studio will always use the correct version of XGantt, the following procedures, that can be found in the BuildSteps directory within the XGantt installation directory (for target framework :NET 2.0 please adjust the line "set DOTNET=..." in both build events) have to be integrated into the pre-build and the post-build event. After having compiled your project once you will receive a not unexpected error message by Visual Studio. Then you have to insert a reference to the XGantt.dll in the new directory C:\XGanttReference (you might have to delete an existing reference to the XGantt installation directory before). Finally, please compile your project once more.

3.25 Resource Scheduler

The ResourceScheduler2 is a substantial enhancement of ResourceScheduler1 (version 3.1). The different object types required for resource scheduling are now anticipated in data tables of their own, which was facilitated by version 4.0 of VARCHART XGantt. In contrast, ResourceScheduler1 merely allowed the different objects like tasks, operations, assignments and resources to be implicitly defined in the maindata table.

The below object types exist in ResourceScheduler2 and need to be defined in data tables of their own; resources may even be defined in up to 25 different tables:

- **Tasks:** These objects are composed by operations (see below) and hold basic properties such as the release date, the due date, priority and quantity.
- **Operations:** These objects can be assigned to resources (see below) by assignments (see below) and will receive the start and end dates of the processing time as a result of scheduling. Operations have a defined position within a sequence of their task and can be marked as "started". Beside, several different sequences of operations can be defined that represent mutually exclusive "routes" of processing. All operations of a route selected by the scheduling procedure will be scheduled.
- **Resources:** As their main features, these objects are part of a capacity curve and after scheduling, they also are part of a workload curve. Beside, they time the operations that they have received (timing resource). Therefore, in order to be scheduled, an operation needs to be assigned to a resource. Beside a timing resource, also work and material resources can be assigned to an operation. Another essential feature of a timing resource is its ability to be grouped on multiple levels. A timing resource may belong to different groups at one time.
- **Assignments:** These objects are the links between operations and resources, that allow to specify a factor for the quantity to be multiplied or divided. When groups of timing resources are scheduled, the assignments are marked correspondingly and additional assignments are generated for each single resource, so that they can be scheduled and displayed in VARCHART XGantt.
- **Links:** These objects describe the sequence of tasks, i.e., preceding tasks have to be finished before the succeeding ones can start.

Survey of the Objects and Their Properties

Task Table	
TaskDataTableName	Name of the task table
TaskDueDateFieldIndex	Date, up to which a task has to be finished
TaskPlanningStrategyFieldIndex	Planning strategy: ASAP or JIT for single tasks
TaskPriorityFieldIndex	By assessing the importance of a job, the priority will bring forward a job or put it on hold.
TaskQuantityFieldIndex	Quantity to be produced by the task.
TaskReleaseDateFieldIndex	Date from which onward a task is allowed to be scheduled.
TaskResultEndDateFieldIndex	Scheduled date of finish
TaskResultPostEndDateFieldIndex	Scheduled date of post time finish
TaskResultPreparationStartDateFieldIndex	Scheduled date of preparation time start
TaskResultProcessingStepFieldIndex	Scheduled sequence number of the task
TaskResultProcessingTimeFieldIndex	Scheduled planning time of the task
TaskResultRouteFieldIndex	Scheduled route consisting of the resources available that work off the task
TaskResultStartDateFieldIndex	Scheduled start date of the task

Operations Table	
OperationDataTableName	Name of the operation table
OperationMaximumInterruptionTimeFieldIndex	Maximum time for which the operation is allowed to be interrupted while occupying a resource
OperationLoadPerItemFieldIndex	Load of resource per item
OperationOverlapQuantityFieldIndex	Overlapping time with other resources
OperationPostLoadFieldIndex	Post load of the operation
OperationPreparationLoadFieldIndex	Preparation load of the operation
OperationResultPostEndDateFieldIndex	Scheduled finish of the post time
OperationResultProcessingTimeFieldIndex	Scheduled processing time of the operation
OperationResultPreparationStartDateFieldIndex	Scheduled start date of the preparation time
OperationResultSelectedTimingResourceIDFieldIndex	Determined ID of the timing resource

192 Important Concepts: Resource Scheduler

Operations Table	
OperationResultStatusFieldIndex	Error or warning state
OperationRouteFieldIndex	Route to which the operation belongs
OperationSequenceNumberFieldIndex	Sequence of the operation within the route
OperationStartLockDateFieldIndex	Fixed start date
OperationTaskIDFieldIndex	Task, to which the operation belongs
OperationWorkInProgressFieldIndex	Degree of completion of the operation

Resourcen Table	
ResourceCalendarNameFieldIndex	Name of the resource calendar
ResourceCapacityType	Finite or infinite capacities for all resources
ResourceCapacityTypeFieldIndex	Finite or infinite capacities for single resources
ResourceConstraintTypeFieldIndex	Condition for work and material resources
ResourceDataTableName	Name of the resource table
ResourceEfficiencyFieldIndex	Efficiency in %
ResourceGroupDataTableName	Name of the table of group resources
ResourceGroupIDFieldIndex	Group identity of the resource
ResourceNameFieldIndex	Name of the resource
ResourceResultLoadCurveNamePrefix	Curve to which the scheduled work load of work and timing resources is to be stored
ResourceResultStockCurveNamePrefix	Curve to which the scheduled stock of material resources is to be stored
ResourceSelectionStrategy	Selection strategy of resources
ResourceSoftConstraintStartDateFieldIndex	Date of status change of a resource from "hard" to "soft"
ResourceType	Type of resource
ResultProcessingStepCount	Number of scheduled tasks

Assignment Table	
AssignmentDataTableName	Name of the assignment table
AssignmentIsResultFieldIndex	Was the data record generated by the scheduling procedure?
AssignmentIsVisibleFieldIndex	Should the assignment be visible in

Assignment Table	
	the chart?
AssignmentLoadOrConsumptionFieldIndex	Value per item
AssignmentMaximumLoadFieldIndex	Maximum work load limit
AssignmentMinimumLoadFieldIndex	Minimum work load limit
AssignmentOperationIDFieldIndex	Operation assigned
AssignmentResourceSelectionStrategyrFieldIndex	ASAP or JIT for a single resource
AssignmentResourceIDFieldIndex	Resource assigned

Link Table	
LinkDataTableName	Name of the link table
LinkDurationFieldIndex	Minimum time offset
LinkPredecessorTaskIDFieldIndex	Predecessor task of the link
LinkSuccessorTaskIDFieldIndex	Successor task of the link

General Properties and Methods	
BaseTimeUnit	Separate time unit for resource scheduling
BaseTimeUnitsPerStep	Coarse or small steps for scheduling?
DataRecordEventsEnabled	Should DataRecord events be enabled?
DefaultOperationMaximumInterruptionTime	Maximum duration of a unique interruption for operations
DefaultResourceCalendarName	Default calendar for scheduling
FullUsageOfPlanningUnitsEnabled	Using up remaining capacities of resources
PlanningEndDate	End of the scheduling time span
PlanningStartDate	Beginning of the scheduling time span
PlanningStrategy	Planning strategy: ASAP or JIT for all tasks
Process	starting the scheduling procedure
ToleranceTimeOnASAPDueDates	Allowance to the due date
ToleranceTimeOnJITReleaseDates	Allowance to the release date
ToleranceTimeOnStartLockDates	Allowance to a locked start date
WorkInProcessType	Unit of the degree of completion
WritingDebugFilesEnabled	Should debug files be written?

After having set the properties of the table, the scheduling procedure can be started by invoking the method **Process**.

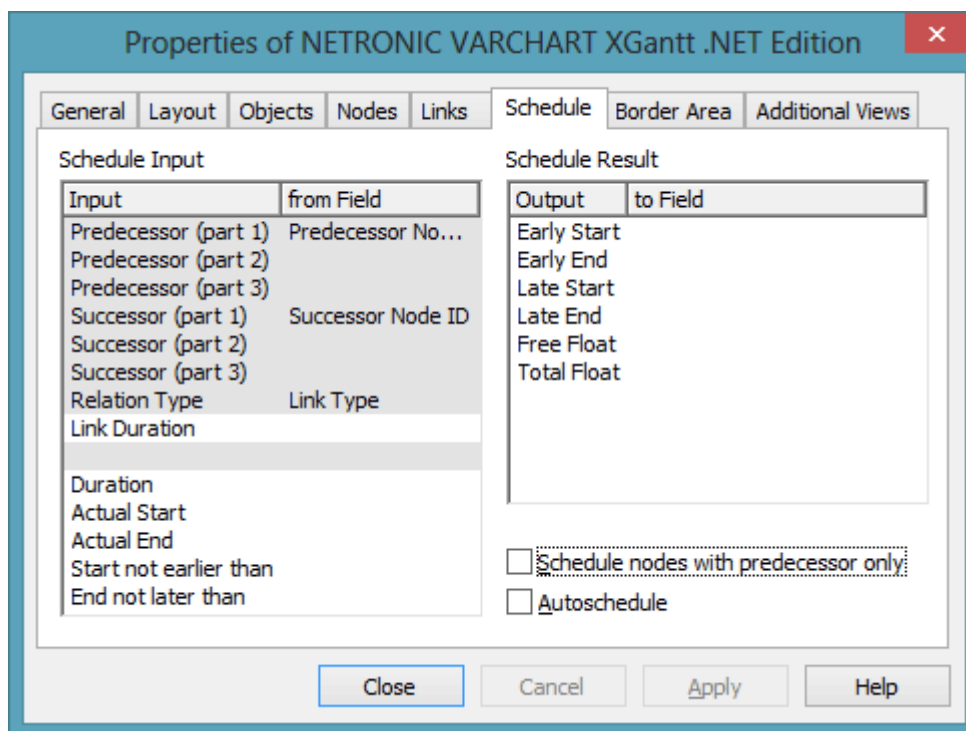
3.26 Schedule

You can perform simple date calculations by using the VARCHART XGantt scheduler. The start and end dates of the project are to be passed as parameters.

By the **Schedule** property page you can adapt the date calculation settings of VARCHART XGantt to your interface by specifying the data fields that you want to use for the input (**Schedule Input**) and output (**Schedule Result**) of the scheduler.

The scheduler uses data fields of the respective nodes and links tables .

The key data for calculating the dates is the durations of the activities, their logical dependencies and the project start. Those informations are used to calculate the early/late start and end dates plus the total float and the free float. The **Predecessor** and **Successor** fields cannot be edited in the **Schedule Input** table. They merely display the settings that were made on the **Links** property page.

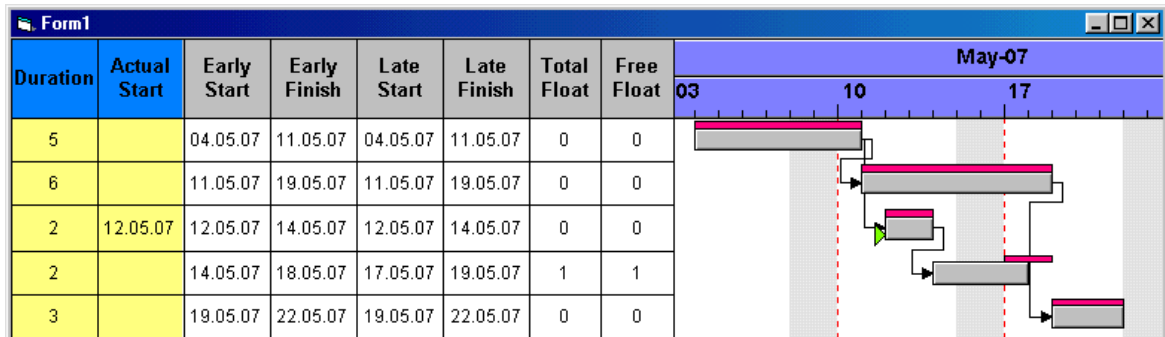


The results are stored to data fields of the interface. Available results are: **Early Start, Early Finish, Late Start, Late Finish, Total Float** and **Free Float**. To each of the results you can assign a field from the list of fields specified in the data definition. The below examples were calculated for the project start on May 4th, 2013, which you can set in the API by typing the below code:

Example Code VB.NET

```
VcGantt1.ScheduleProject ("04.05.2013", "11.05.2013")
```

The settings illustrated above would give the following graphical display:

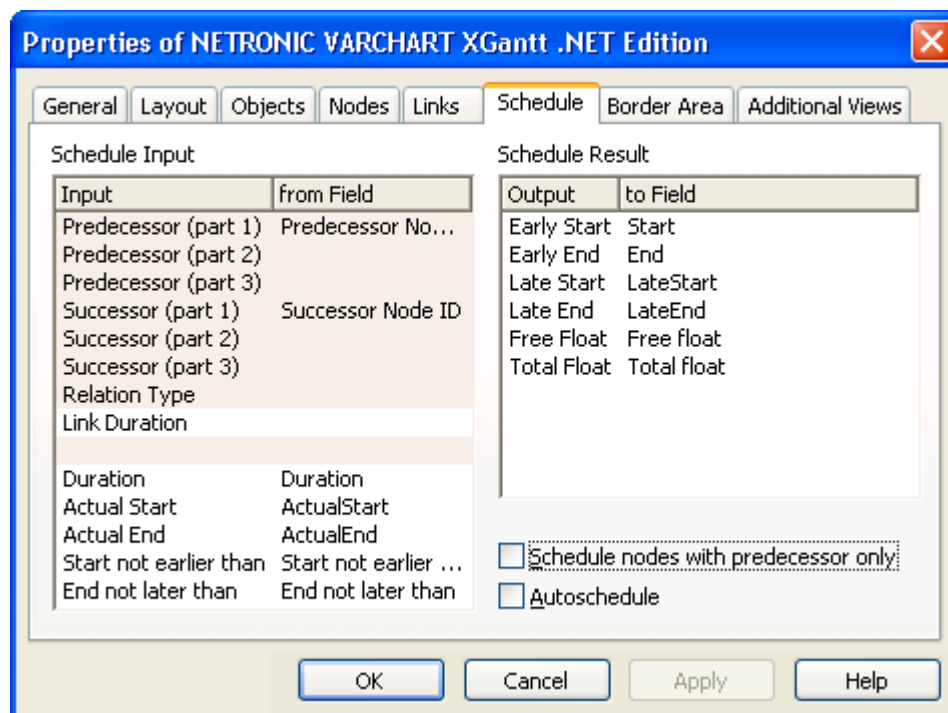


In this example, the early and late dates are both shown as layers.

There are further possibilities to manipulate VARCHART XGantt scheduler's date calculations.

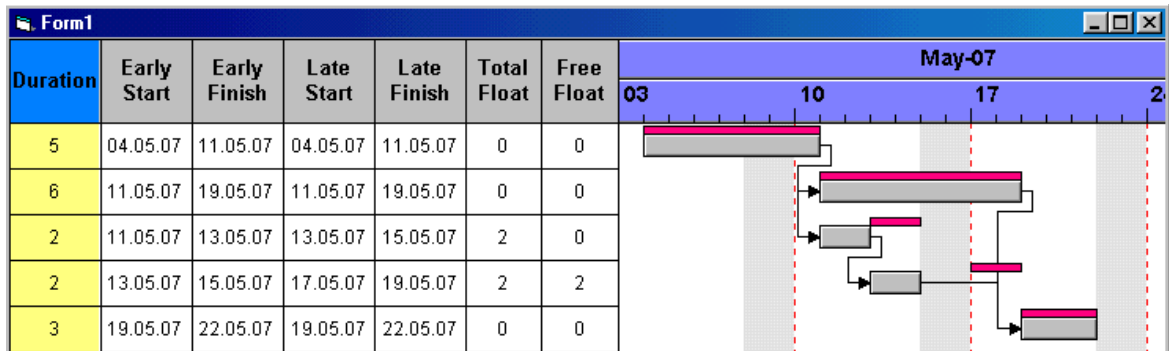
1. You can specify actual start/end dates. This way, the activities cannot be moved.
2. You can specify reference dates for the **Start not before** and **End not later than** expressions by defining a field from the data definition for each respective value in the left-hand table on the **Schedule** property page.

The below diagram shows the settings that were made for the example following:

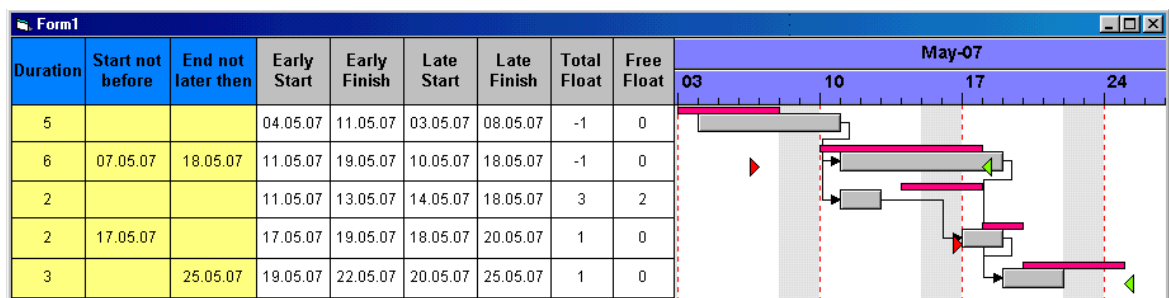


196 Important Concepts: Schedule

By setting the actual start of an activity, the early and late dates are also fixed. In the following example, the actual start date set is marked by a green triangle.



Using the expressions **Start not before** and **End not later than** may or may not have an effect. In the following example, the date limits are marked by red and green triangles. Some do not have any effect on the date calculation, although the end date restriction of the second activity means that a negative float has been calculated for the first two.



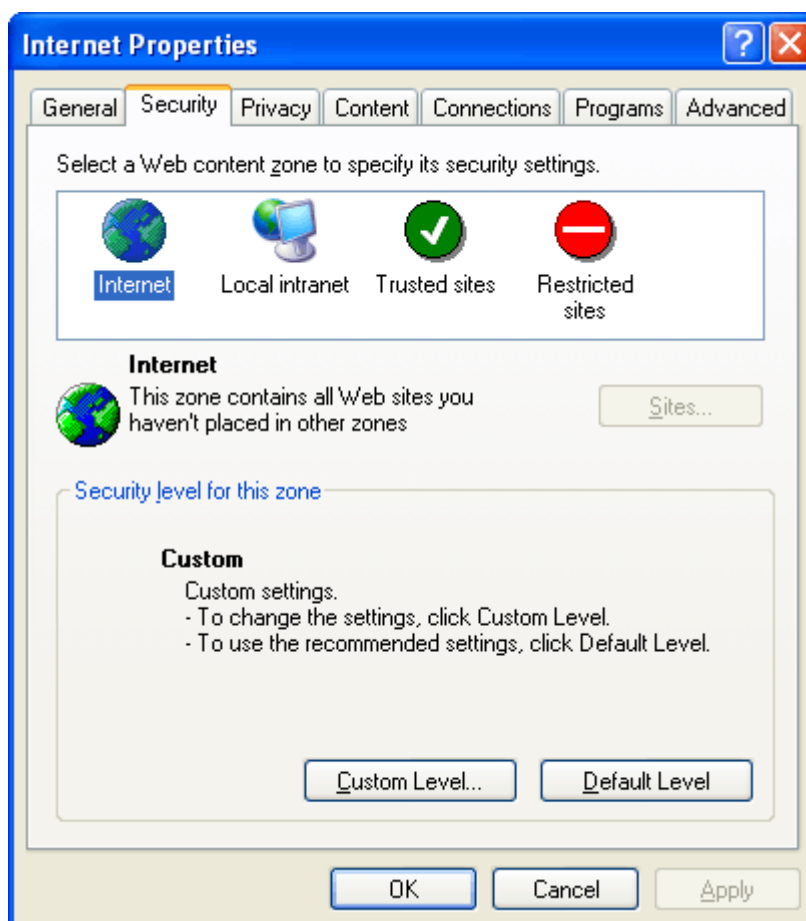
3.27 Security Guidelines for the Deployment in the Internet Explorer

In order to use the VARCHART XGantt control in a HTML page in the Internet Explorer the security guidelines have to be modified.

As soon as the browser loads the control from a web server on the Internet the **Security guidelines** of the Internet_Zone become active. The default settings prevent the control from being executed. The Internet Explorer has to permit the execution of .NET components so that they become visible at all.

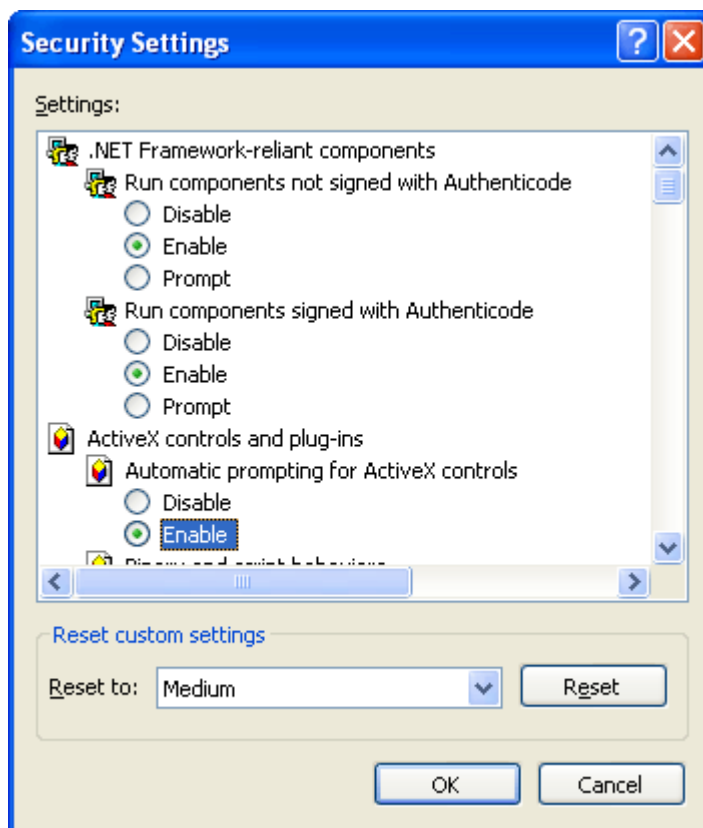
The guidelines can be modified in the Internet Explorer dialog **Security Settings** which you can reach by **Control Panel > Internet Properties > Security > Internet**.

Please select the **Zone Internet** or **Trusted Sites**.



For the zone selected, please click on **Custom Level...** and enable both, **Run Components not signed with Authenticode** and **Run Components signed with Authenticode**.

198 Important Concepts: Security Guidelines for the Deployment in the Internet Explorer



In addition, the runtime guidelines on the local computer need to be changed.

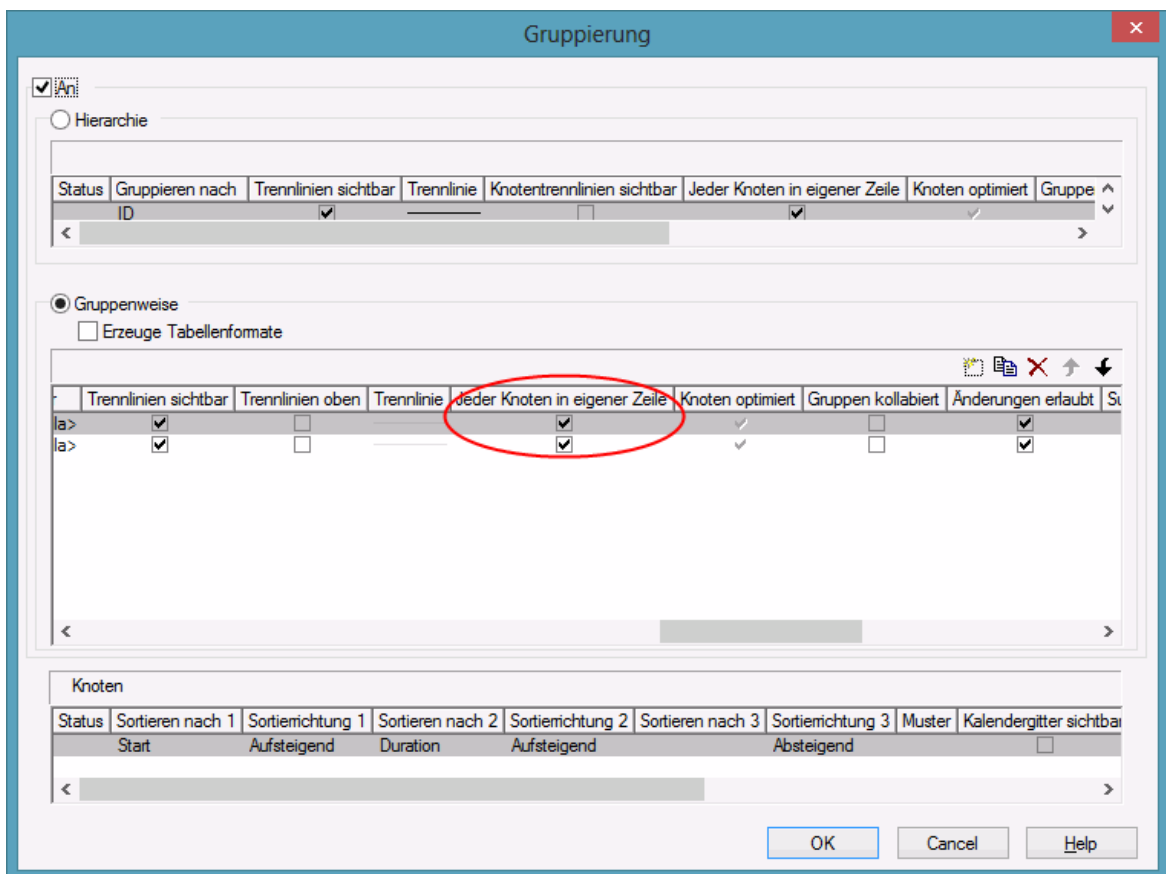
In the **CAS** directory of the VARCHART XGantt installation you can find two complementing batch files. The first one is **AddRights.bat**. It lets you create a permission set and a code group for NETRONIC controls. If later on you wish to deliver your application to a customer, the batch file needs to be executed on each client system before running your application. The second one is named **RemoveRights.bat** and lets you cancel permissions. Thus the VARCHART XGantt control can be executed on a HTML page in the internet Explorer using a minimum set of permissions.

3.28 Sorting

Usually, applications require activities to be sorted according to certain criteria. Only those nodes can be sorted, that do not form part of a hierarchy, i.e. that are base nodes or belong to a group. So you will find setting options in places where you can set properties of group nodes and base nodes. When sorting nodes, it makes a difference whether nodes are arranged in separate rows or whether several nodes are displayed in a single row.

Arrangement: Nodes in Separate Rows

If you wish the nodes to be arranged in separate rows, please invoke the **Grouping** dialog that you can get to by selecting the **Objects** property page and then **Grouping**:



In the center window, please tick the box **Nodes in separate rows**. Alternatively, you can set this feature by the API property **VcGroupLevel-Layout.AllNodesInOneRow**.

In the window below which is called **Nodes** you can specify three data fields by which the activities are to be sorted when the diagram pops up. In

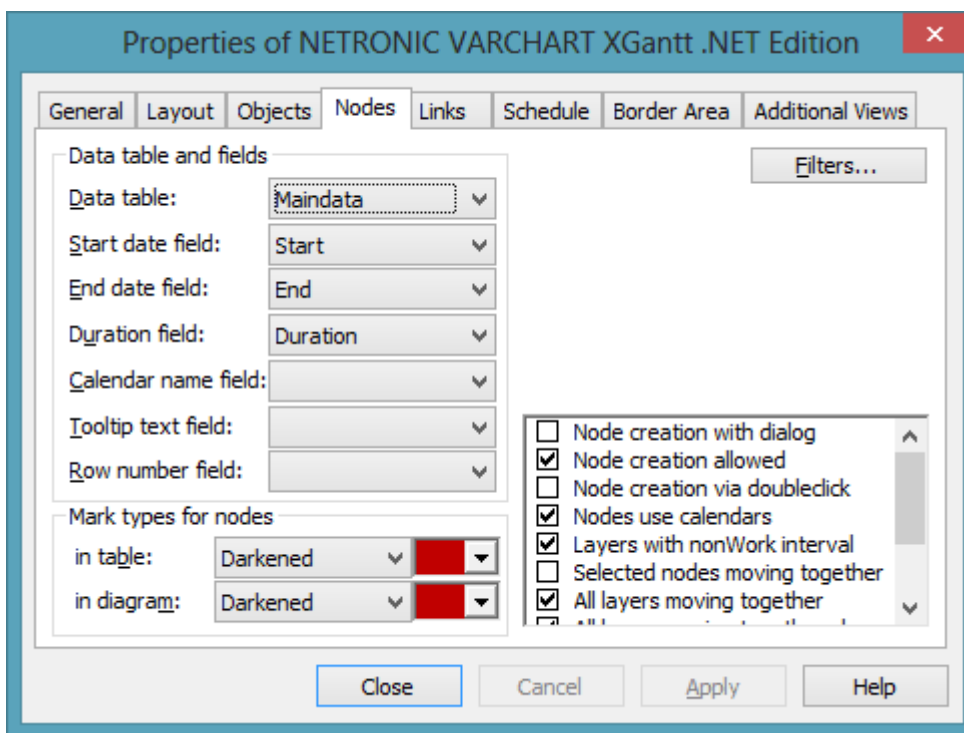
addition, you can select an ascending or a descending sorting order for each of the data fields.

If the activities are grouped, sorting will apply to the nodes of each group.

Beside, the below options for defining the appearance of the node line are available :

- Selection of a **Pattern**
- display, position and style of the **Separation Line**
- specify after how many activities a separating line should be drawn by entering a value in the field **Separation line step size**. If the activities are grouped, the counting will be done separately for each group.

Further sorting options can be set on the **Nodes** property page:



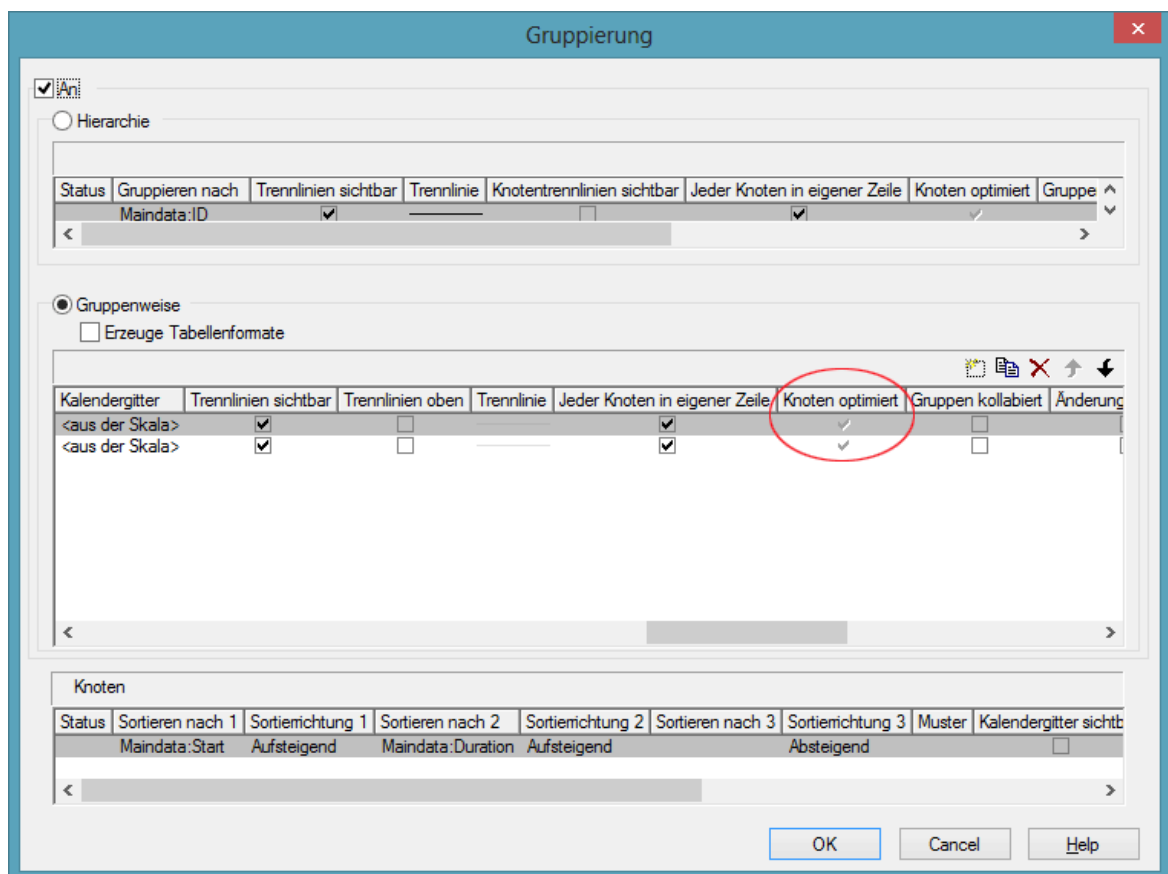
- You can select a data field to which the row numbers of the activities are stored. The **row number field** will not be updated until saving the data by the **Save As** method.
- By ticking **Moving a node vertically via diagram allowed** and/or **Moving a node vertically via table allowed** you can enable the user to modify the order of the activities by dragging them to a different row. If an activity is moved to a different group, its grouping code and color will adjust to the new group. If an activity is comprises more than one layer, the **Shift** key has to be pressed in addition.

Note: Please note that the settings in the **Grouping** dialog and on the **Nodes** property page are only used to sort the data when the application is started. If you want to sort the activities later again, please use the method **SortNodes**. So an update of the sorting has to be invoked separately by this call.

Arrangement: Nodes of a Group in One Row

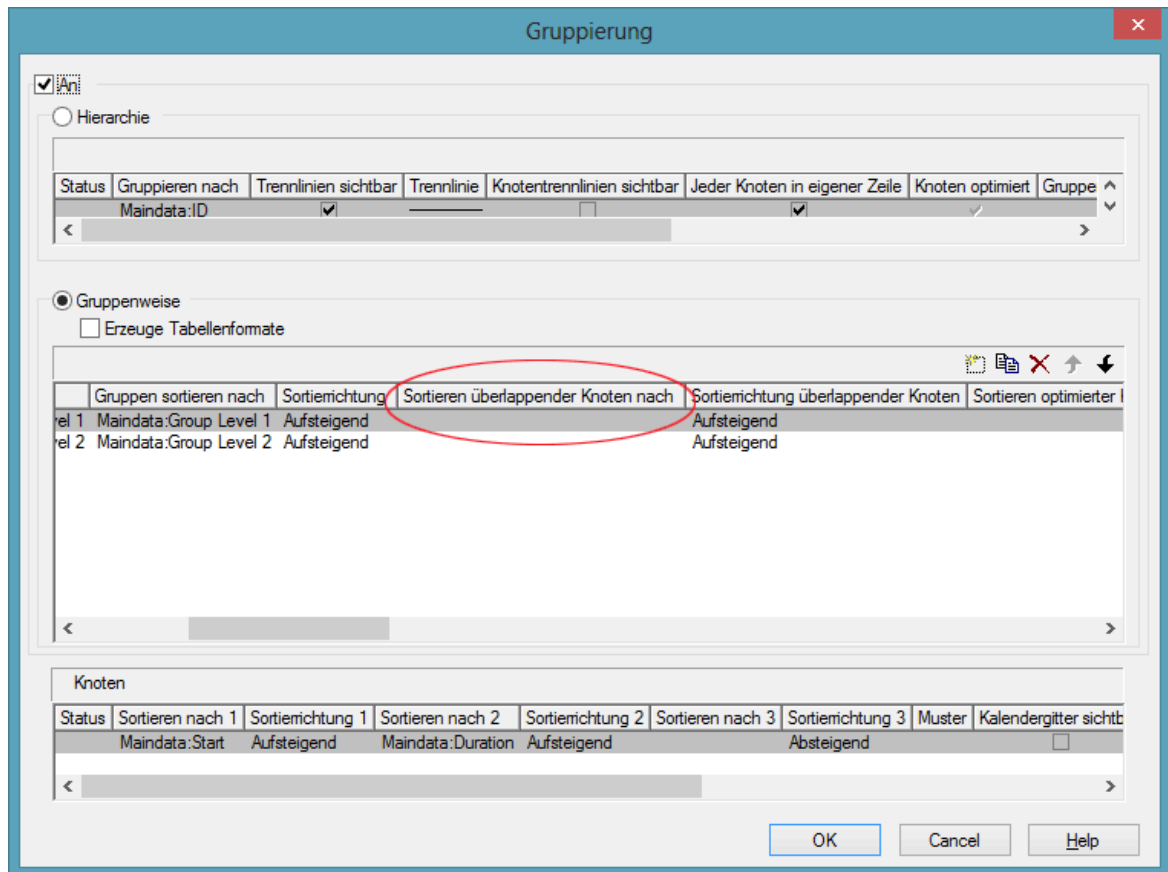
If several nodes (i.e. the nodes of a group) are put in a single row, you can assign a drawing priority (which is also a kind of sorting) to the nodes. Two different types of arrangement exist, the **overlapping** one and the **optimized** one, where the activities of one row either overlap each other or avoid overlapping by widening the row.

You can put several nodes in one row by unticking the box **Nodes in separate rows** in the **Grouping** dialog. By default, the adjacent field **Nodes optimized** will appear activated:



You can deactivate this check box which will entail the nodes of a row being displayed as overlapping. You can alternatively set this feature by the API property **VcGroup.NodesArrangedOptimized**.

The drawing priority of the nodes you can set by the field **Sort overlapping nodes by**:

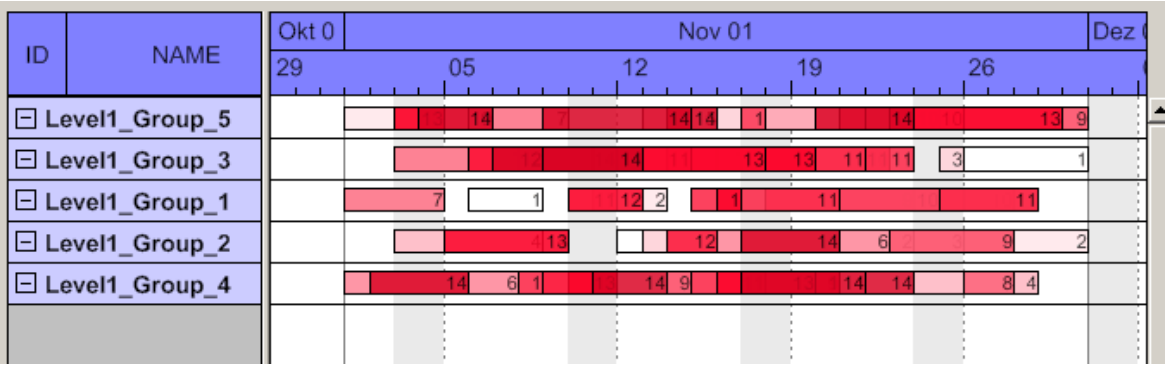


In analogy to overlapping nodes, you can sort optimized nodes by the field **Sort optimized nodes by**.

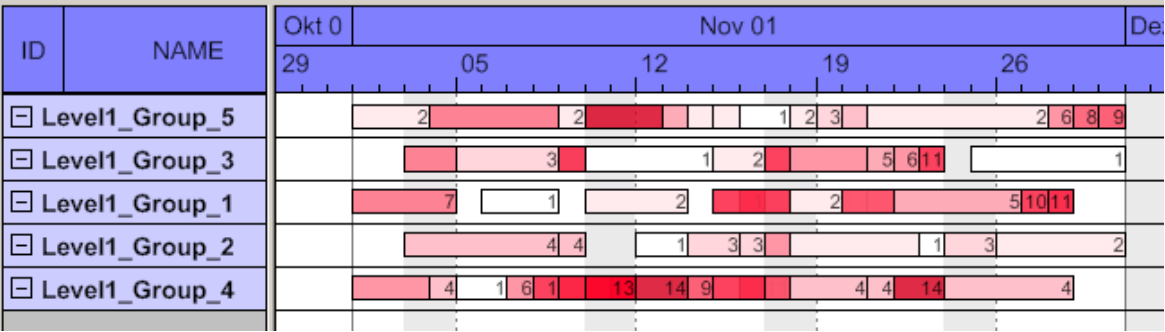
If you do not set a sorting priority, the nodes by default will be displayed in the order of their date and duration, the latest and shortest ones being drawn on top of the earlier and longer ones. The drawing priority can also be set by the API properties **VcLevelLayout.OverlaidNodesSortDataFieldIndex** and **VcLevelLayout.OptimizedNodesSortDataFieldIndex**.

You do not need to update the sorted nodes by a separate call, they will update automatically. Besides, by the adjacent field **Overlapping nodes sort order** you can assign an ascending or descending sort order. The sorting direction can alternatively be set by the API properties **OverlaidNodesSort-Order** and **OptimizedNodesSortOrder**, respectively.

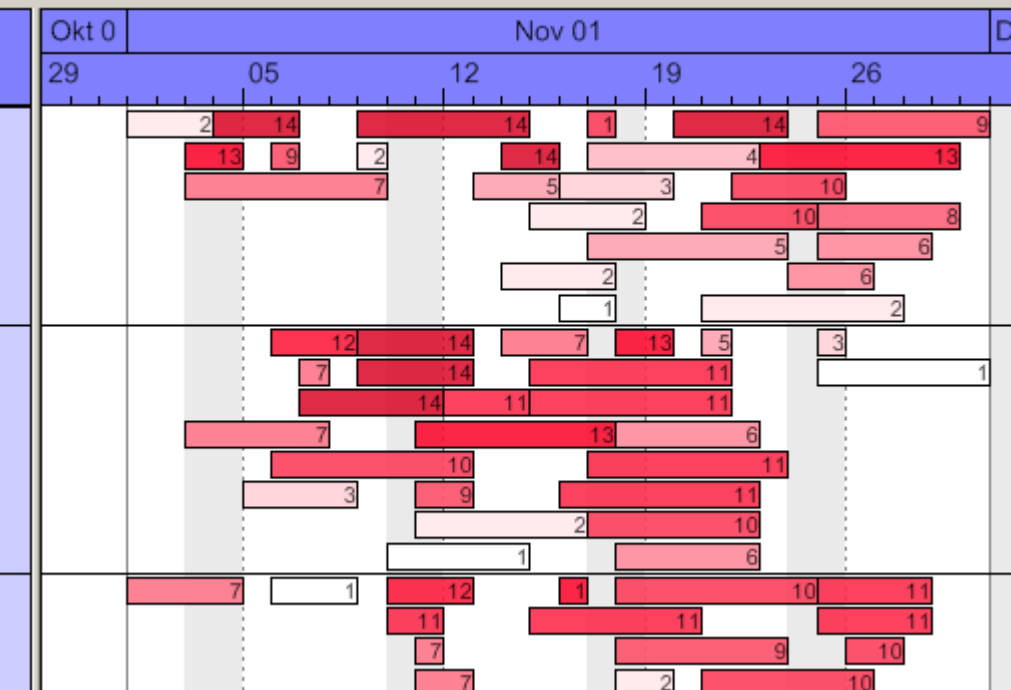
Below, some results of the settings are shown:



Overlay node arrangement showing an ascending drawing priority of dark nodes (dark nodes drawn on top of light nodes)

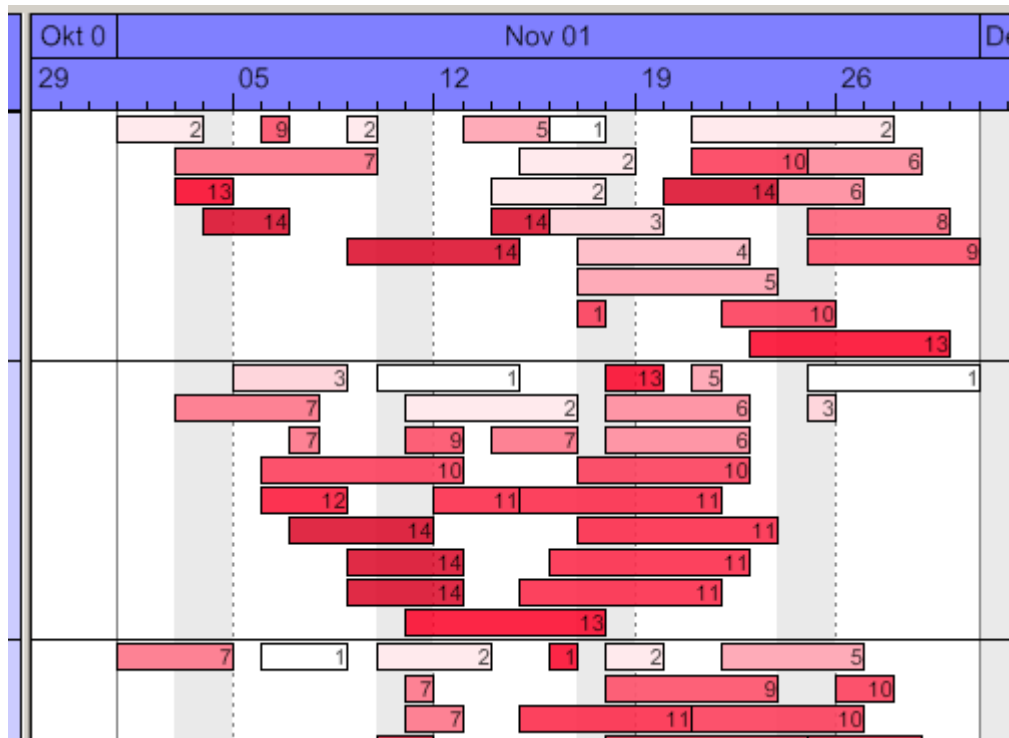


Overlay node arrangement showing an descending drawing priority of dark nodes (light nodes drawn on top of dark nodes)



Optimized node arrangement showing an ascending drawing priority of dark nodes (dark nodes drawn in the upper section of the row)

204 Important Concepts: Sorting



Optimized node arrangement showing an descending drawing priority of dark nodes (light nodes drawn in the upper section of the row)

3.29 Table

The properties of the table can be set by three different dialogs, that can be reached by the property page **Objects** and the button **Table**. The dialogs of the actual table features are named **Specify Table**, **Edit Table** and **Edit Table Format**. You can create several tables in the **Specify Table** dialog.

The table consists of six columns (default) that are only visible if they are assigned a width greater than 0. The rows in the table are defined by table formats. For each table format you can specify the font style, font color, background color, alignment and margins. Each format is applied in certain conditions:

- **StandardListCaption** for the table header
- **StandardList** for activities/rows.

In addition to the default table formats you can create table formats for that you can specify names and filters individually.

Table formats for a hierarchical arrangement:

The hierarchical arrangement can be set on the property page **Objects** by clicking on the button **Grouping**.

- **Hierarchy:** Format for hierarchical levels when expanded; the second field (usually the activity name) will be indented to display a lower level. A "-" indicates that the level can be collapsed.
- **HierarchyCollapsed:** Format for collapsed hierarchy levels. A "+" indicates that the level can be expanded.

ID	NAME	START
1	<input type="checkbox"/> SW Development	02.09.98
1.2	<input checked="" type="checkbox"/> Design&Concept	02.09.98
1.3	<input type="checkbox"/> Coding	09.09.98
1.3.1	Phase A (DB)	09.09.98
1.3.2	Phase B (GUI)	15.09.98
1.4	<input checked="" type="checkbox"/> Testing	17.09.98
1.5	Sales & Marketing	05.09.98
1.6	Delivery	24.09.98
1.7	Final Party	

Picture above: The format **HierarchyCollapsed** is displayed in the row **Design&Concept** indicating a collapsed hierarchy level; the format **Hierarchy** is displayed in the row **Coding**, indicating an expanded hierarchy level.

Table formats for a grouped arrangement:

A grouped arrangement can be set on the property page **Objects** by clicking on the button **Grouping**.

- **Subtitle:** for the headers of non-collapsed groups. The header consists of a single field that fills the width of the table completely. A "-" indicates that a group can be collapsed.
- **Collapsed:** Format for the headers of collapsed groups. A "+" indicates that a group can be expanded.

ID	NAME	START
[-] A		
1	SW Development	02.09.08
3	Requirements	02.09.08
7	Final Check	16.09.08
12	QA Requirement Check	23.09.08
[+] Group C		
[+] Group B		
[-] E		
15	Final Party	30.09.08

Picture above: The format **Subtitle** is displayed in the rows **GroupC** and **GroupB** indicating a collapsed group level; the format **Subtitle Collapsed** is displayed in the rows **A** and **E**, indicating an expanded group level

3.30 Time Scale

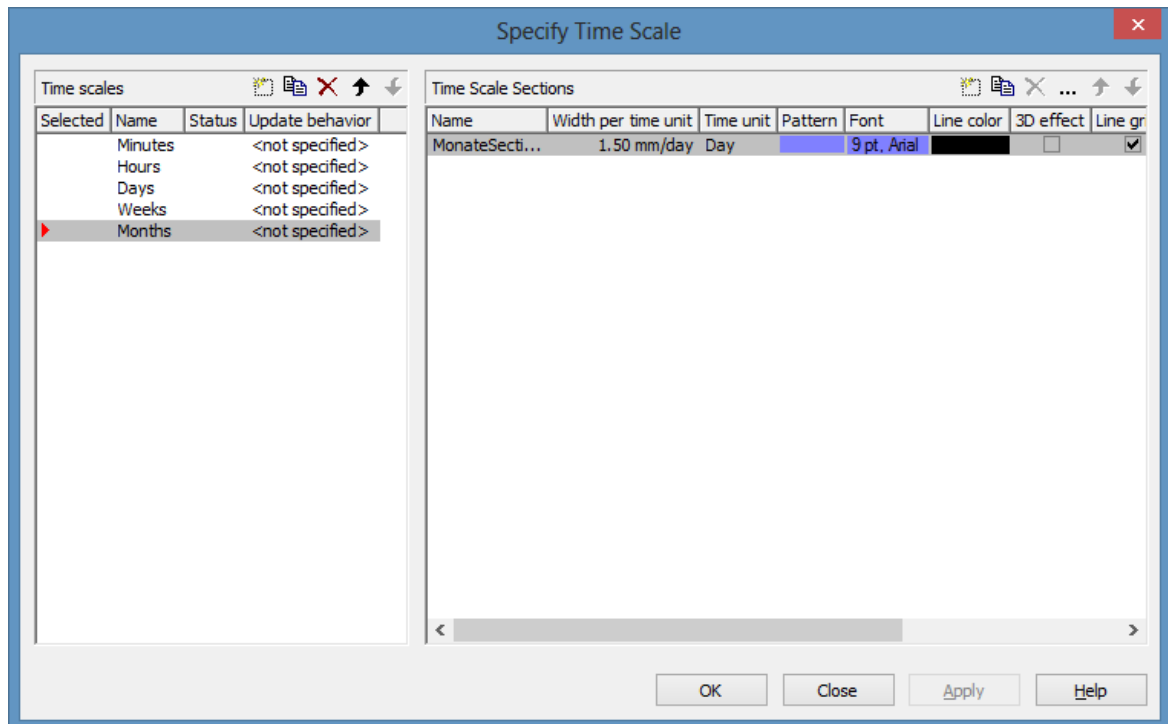
Above the diagram area, the time scale is displayed. You can display one or more annotated ribbons of the time scale below the diagram area, too (see **Edit Time Scale Section** dialog box, **Ribbons, Position**). The appropriate timescale for the time period displayed can be selected.

You can divide the time scale into sections, specifying the number of sections to be displayed, their ranges and scales. Project phases that you want to plan in particular detail can be displayed in a more "magnified" form than the other phases: Perhaps you wish to present your project plans for the immediate future in more detail than your plans for the distant future or past, enabling you to concentrate on the project phases that are currently of most interest to you and shift the focus as your project progresses. Or you can start with a general project overview and continue your planning in increasing detail.

Nov-07				Dec-07			
CW 45	CW 46	CW 47	CW 48		CW 49	CW 50	CW 51

There is a whole range of options for designing the timescale, sections and grids. For each individual object, you can specify the scales, notations, font attributes, text alignment, colors, line thicknesses, line types, and so on. To keep your planning transparent, you can define grids, e.g. a day or week grid, for each section.

You can select the timescale you want to use for your diagram (**Selected**) from the range of preset timescales offered in the **Specify Time Scale** dialog. The time scales differ from one another in the width of the time unit and the ribbons.



It is possible to change the selection during runtime.

> Specifying start and end dates of the time scale

The default start and end dates of the time scale are specified on the **General** property page (**Project Start** and **Project End**). At runtime, fit this value to the current data via the **TimeScaleStart** property or the **OptimizeTimeScaleStartEnd** method. The date format is "DD.MM.YYYY;hh:mm:ss".

Note: The end date is not included. If you specify **TimeScaleEnd** = "31.12.02" for example, the last day displayed will be the 30.12.02.

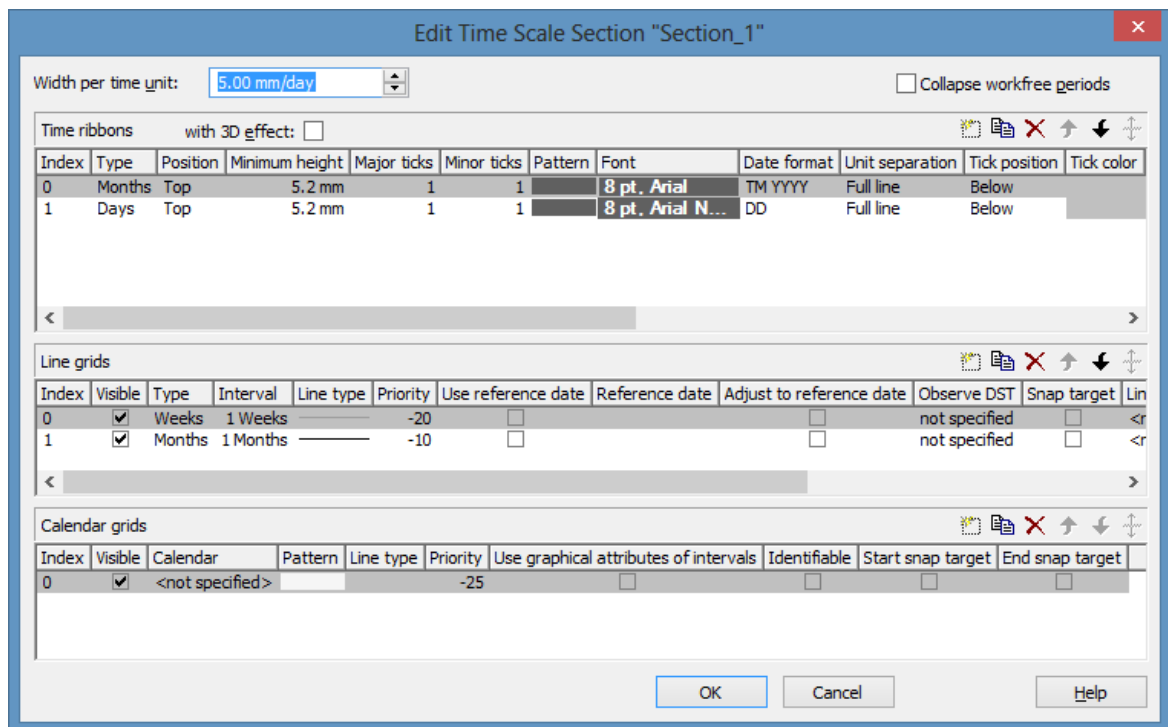
> Sections

You can split the time scale into sections to highlight certain planning phases and specify different ribbons for each section. In the **Specify Time Scale** dialog you can set the **Time unit** and the **Width per time unit** individually for each section. Also, for each section you can define a separate color, font, pattern 3D effect, line grid and calendar grid, and specify whether workfree periods are suppressed.

When you select a line grid, variable vertical grid lines are displayed in the appropriate section.

When you use a calendar, a predefined calendar grid can be displayed in the appropriate section where workfree periods are marked by colored vertical areas.

From the **Specify Time Scale** dialog you can reach the **Edit Time Scale Section** dialog box where you can edit each of the ribbons and grids of each section.



> Width per time unit

The unit is the smallest unit the time scale is divided to. Possible unit widths are: second, minute, hour and day. You can specify the unit in the **Specify Time Scale**.

You can specify the **Width per time unit** in millimetres per unit width in steps of 100th of a millimetre per unit width. The minimum width you can assign to the time unit is 0.01 mm.

> Ribbons

Ribbons serve the purpose of annotating the timescale. Each section may be assigned several ribbons (e.g. one with a monthly and a second with a daily scale). For each ribbon you can specify the **Position**, i. e. whether it is to be displayed or not and whether it is to be displayed at the top or at the bottom of the diagram. Furthermore, you can specify for each ribbon the following: the type, minimal height, major and minor ticks, color, font, date format, unit separation, alignment, serial annotation, reference date, calendar.

Which date formats are available for a particular ribbon depends on the type of ribbon selected.

To compose the date you can use the following tokens:

210 Important Concepts: Time Scale

D:	first letter of the day of the week (not adjustable)
TD:	Day of the Week (adjustable by using the event VcTextEntrySupplying)
DD:	two-digit figure for the day of the month: 01-31
DDD:	first three letters of the day of the week (not adjustable)
M:	first letter of the name of the month (not adjustable)
TM:	name of the month (adjustable by using the event VcTextEntrySupplying)
MM:	two-digit figure for the month: 01-12
MMM:	first three letters of the name of the month (not adjustable)
YY:	two-digit figure for the year
YYYY:	four-digit figure for the year
WW:	two-digit figure for the number of the calendar week: 01-53
TW:	text for "calendar week" (adjustable by using the event VcTextEntrySupplying)
Q:	one-digit figure for the quarter: 1-4
TQ:	name of quarter (adjustable by using the event VcTextEntrySupplying)
hh:	two-digit figure for the hour in 24 hours format: 00-23
HH:	two-digit figure for the hour in 12 hours format: 01-12
Th:	Text of "o' clock" (adjustable by using the event VcTextEntrySupplying)
TH:	"am" or "pm" (adjustable by using the event VcTextEntrySupplying)
mm	two-digit figure for the minute: 00-59
ss:	two-digit figure for the second: 00-59
TS:	short date format, as defined in the regional settings of the windows control panel
TL:	long date format, as defined in the regional settings of the windows control panel
TT:	time format, as defined in the regional settings of the windows control panel
xC/XC:	<i>The usage of this date format requires a special setting in the .ini</i>

file. Please contact NETRONIC if necessary. You can set a maximum ten-place, simple upward counting from a reference date onward, for example "15:05:07:16:00", which equals 15 months, 5 days, 7 hours, 16 minutes, 0 seconds. The notation is: **xC44:C33:C22:C11:C00**. In written language: Show at least 2 digits for the counters 4...0 and a preceding "-" symbol if the value is negative. The separators are variable and can be replaced by other separators symbols. "x" means: Display a preceding "-" symbol if the value is negative, but no "+" symbol if it is positive. "X" means: Display a preceding "-" symbol if the value is negative and a "+" symbol for positive values. In the dialog **Edit Time Scale Section...** the check boxes **Use reference date** and **Adjust major ticks to reference date** need to be ticked, also, the parameter **Serial annotation** has to be set to **No**. In the application the reference date is set at run time by the call **VcRibbon.set ReferenceDate**, overriding any settings in the dialog.

Note: Characters which are not to be interpreted as part of the date should be preceded by a backslash '\'. '\\' for instance results in '\'. The special characters: ':, /, -' and **blank** don't need '\' as prefix.

> Example for the ribbon annotation

1. ribbon: **TWWW - TM - TQ - YYYY**, 2. ribbon: **TD**

CW37 - September - Quarter 3 - 2007							
Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday	Monday

You can replace the predefined texts by our own texts by setting the property **TextEntrySupplyingEventEnabled** to "True". Then you can react to the following values of the ControlIndex:

- vcTXERibDay0 to vcTXERibDay6 (2212 to 2218)
- vcTXERibCW (2223)
- vcTXERibMon0 to vcTXERibMon11 (2200 to 2211)
- vcTXERibQuar0 to vcTXERibQuar2 (2219 to 2222)

Example Code VB.NET

```
Private Sub VcGantt1_VcTextEntrySupplying(ByVal sender As Object, ByVal e As NETRONIC.XGantt.VcTextEntrySupplyingEventArgs) Handles VcGantt1.VcTextEntrySupplying
```

```
    Select Case e.ControlIndex
        Case VcTextEntryIndex.vcTXERibDay0
```


212 Important Concepts: Time Scale

```
        e.Text = "Lundi"
    ...
    Case VcTextEntryIndex.vcTXERibCW
        e.Text = "Semaine"
    Case VcTextEntryIndex.vcTXERibMon8
        e.Text = "Septembre"
    Case VcTextEntryIndex.vcTXERibQuar3
        e.Text = "3. Trimestre"
End Select
End Sub
```

Example Code C#

```
private void VcGantt1_VcTextEntrySupplying(object sender,
NETRONIC.XGantt.VcTextEntrySupplyingEventArgs e)
{
    switch(e.ControlIndex)
    {
        case VcTextEntryIndex.vcTXERibDay0:
            e.Text = "Lundi";
            break;
        ...
        case VcTextEntryIndex.vcTXERibCW:
            e.Text = "Semaine";
            break;
        case VcTextEntryIndex.vcTXERibMon8:
            e.Text = "Septembre";
            break;
        case VcTextEntryIndex.vcTXERibQuar3:
            e.Text = "3. Trimestre";
            break;
    }
}
```

Semaine 37 - Septembre - 3. Trimestre - 2007							
Lundi	Mardi	Mercredi	Jeudi	Vendredi	Samedi	Dimanche	Lundi

3.31 Tooltips during Runtime

You can use tooltips to provide information on the object that was touched by the mouse. By the event **VcToolTipTextSupplying** you can edit the texts of all the tooltips that appear at run time, for example in order to translate them into different languages or suppress them.

To activate the event, activate the check box **VcToolTipTextSupplying events** on the **General** property page.

Or set the property **ToolTipTextSupplyingEventEnabled** to **True**.

Example Code VB.NET

```
VcGantt1.ToolTipTextSupplyingEventEnabled = True
```

Example Code C#

```
VcGantt1.ToolTipTextSupplyingEventEnabled = true;
```

Example Code VB.NET

```
Private Sub VcGantt1_VcToolTipTextSupplying(ByVal sender As Object,
ByVal e As NETRONIC.XGantt.VcToolTipTextSupplyingEventArgs) Handles
VcGantt1.VcToolTipTextSupplying

    Select Case e.HitObjectType
        Case VcObjectType.vcObjTypeDateLine
            e.Text = "Date line"
        Case VcObjectType.vcObjTypeBox
            e.Text = "Box"
    End Select

End Sub
```

Example Code C#

```
private void VcGantt1_VcToolTipTextSupplying(object sender,
NETRONIC.XGantt.VcToolTipTextSupplyingEventArgs e)
{
    switch (e.HitObjectType)
    {
        case VcObjectType.vcObjTypeDateLine:
            e.Text = "Date line";
            break;
        case VcObjectType.vcObjTypeBox:
            e.Text = "Box";
            break;
    }
}
```

Then capture the **VcToolTipTextSupplying** event and define the text you want to have appear or whether no tooltip should be displayed at that location.

3.32 Unicode

To display Unicode characters on the property pages at design time, an appropriate font has to be set by following the menu of the operating system through **Start / Settings / Control Panel / Display / Appearance** to the **Window** field.

Besides, only those characters can be displayed that belong to the language set by the menu items **Start / Settings / Control Panel / Regional and Language options**.

All objects in a VARCHART component which contain texts can display Unicode characters if an appropriate font was set in the corresponding property **Font**.

A Unicode font can be assigned to context menus, tooltips and run time dialogs by the property **DialogFont** of the **VcGantt** object.

You will find an overview of all available fonts, which contain at least part of all unicode characters in "Wazu Japa's Gallery of Unicode Fonts" (<http://www.wazu.jp/index.html>). Detailed information on the Unicode standard is also offered on the homepage of the Unicode Consortium (<http://www.unicode.org>) and on Microsoft's GlobalDev Homepage (http://www.microsoft.com/globaldev/getwr/steps/wrg_unicode.mspx). In Windows 2000 and XP you can find out about the characters contained in the built-in fonts under **Start / Programs / Accessories / System Tools / Character Map**.

When importing CSV files, the method **VcGantt.Load** automatically recognizes whether there is a Unicode or an ANSI file.

Note: The development environments of Visual Studio 6 are not able to use Unicode characters in source code files. Internally however, the strings of VB6 are displayed in Unicode. If you use Visual C++ combined with MFC you have to set the Defines_UNICODE and UNICODE to use strings in Unicode. The version Visual Studio .NET 2002 and later versions allow to edit source code files in Unicode coding. When saving a file, you need to select the coding type "Unicode".

3.33 Using the Control in a Browser Environment

Windows forms controls which are embedded in an HTML page can be displayed by the Microsoft Internet Explorer. Please proceed as follows to use VARCHART XGantt in an HTML page:

At first, develop your application in form of a Windows control library. The major difference to a Windows application lies in the fact that in Microsoft Visual Studio you use another project template. Thus, the new class doesn't derive from **System.Windows.Forms.Form** but from **System.Windows.Forms.UserControl**. Drag the control VARCHART XGantt to the UserControl1. The code which you will have to write corresponds to a Windows application. The result will be an assembly in form of a DLL instead of an EXE-file. The starter sample from the last chapter is to be found as a complete windows control library in the directory `UserGuideSamples\VB.NET\XGantt_Tutorial01_Web` resp. `UserGuideSamples\CSharp\XGantt_Tutorial01_Web`.

You should assemble all files which are needed for the publication over a web server in one folder. We did this for you already in the folder `UserGuideSamples\Web`.

All in all, five files are needed:

File name	Content
NETRONIC.XGantt.dll	VARCHART XGantt control
XGantt_Tutorial01_Web.dll	control library with UserControl1
Configuration.xml	configuration file for the path search
Tutorial01.html	HTML/Start page
xdependent.cab	container for DLLs needed by XGantt

With the file **configuration.xml** you make sure that the assembly DLLs are searched for in the same directory from which the HTML files are retrieved.

The contents of **configuration.xml** looks as follows:

Example Code

```
<configuration>
  <runtime>
    <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
      <probing privatePath=""/>
    </assemblyBinding>
  </runtime>
</configuration>
```

The HTML file **_Tutorial01.htm** has to contain references to the control and the **xdependent.cab**. This purpose is met by object tags. They contain the attributes **id**, **classid**, **height** and **width**.

Example Code

```
<html>
  <head>
    <title>VARCHART XGantt .NET WinForm</title>
    <link rel="Configuration" href="Configuration.xml">
  </head>
  <body>
    <object id="XDependentDummy" width=0 height=0
      standby="Please wait while loading the diagram
prerequisites..."
      classid="CLSID:544C9013-D784-472f-8EA6-BDF86ECF0427"
      codebase="xdependent.cab#version=8,4,2,0"/>
    <object id="XGantt_Tutorial01_WebLibrary"

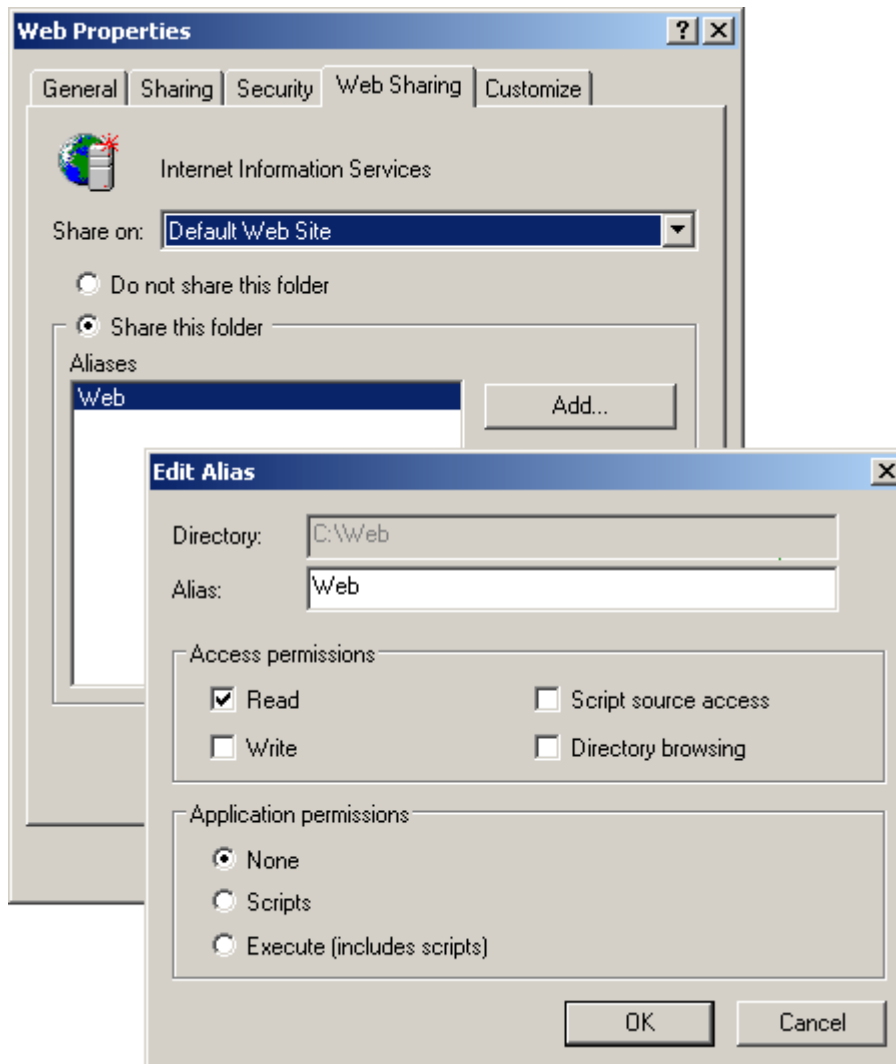
classid="http:XGantt_Tutorial01_Web.dll#XGantt_Tutorial01_Web.UserContro
ll"

      height="500" width="910"/>
  </body>
</html>
```

With **id** you assign an arbitrary identifier. The **classid** specifies the origin. With the HTTP protocol the assembly **XGantt_Tutorial01_Web.dll** is downloaded and the control **XGantt_Tutorial01_Web.UserControl1** addressed within the control library. It is necessary to specify the precise name of the control because in principle a control library may contain several controls. The extent of the area which the control will cover on the HTML page is specified by the attributes **height** and **width**.

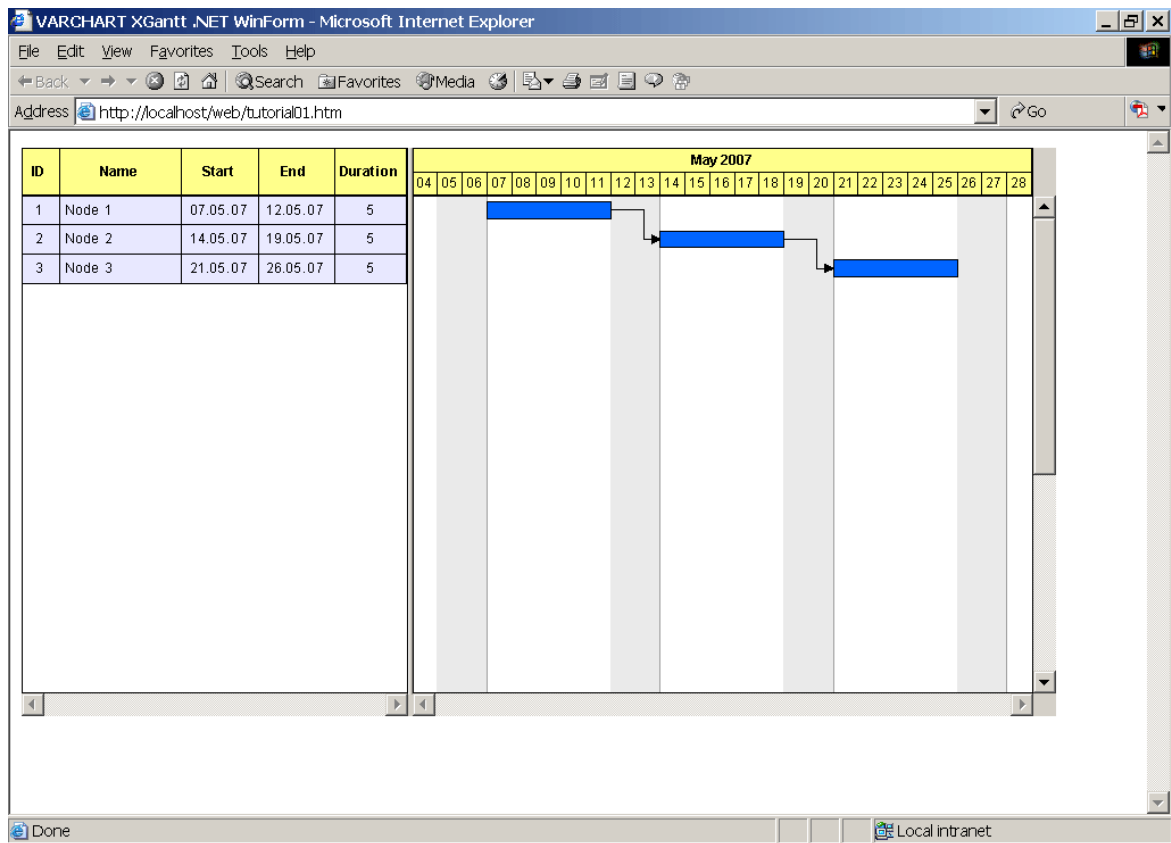
The object tag for **xdependent.cab** is necessary, because the Internet Explorer respectively the .NET Framework 2.0 do not support the automatic download of dependent DLLs for an assembly like XGantt. XGantt needs the DLLs **MFC80U.DLL** and **MFVC80P.DLL** to run. Therefore these DLLs will be downloaded by a CAB file, which installs a small dummy ActiveX control only the first time it is used. This CAB file is signed by NETRONIC. You will have to change the Internet settings for the zone the server lies in, in order to download and install signed ActiveX controls. For further information about the Internet settings see chapter 3.16 **Security Guidelines for the Deployment in the Internet Explorer**.

The web application may be tested simplest with the help of the local internet information service. A web release for the folder **Web** is created. You will reach the corresponding dialog via the context menu of the folder and the entry **Properties**.



After that you can call up the page under the address <http://localhost/web/tutorial01.htm>.

218 Important Concepts: Using the Control in a Browser Environment



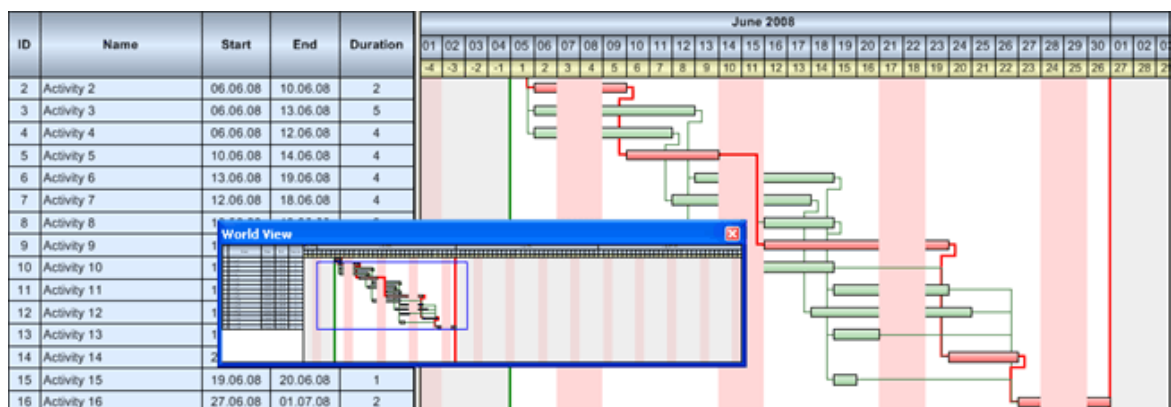
3.34 Viewer Metafile (*.vmf)

VMF is a graphics format that was especially developed for the WebViewer (a Java applet independent of platforms and browsers) by NETRONIC Software GmbH. The VMF format allows you to view, zoom or move your diagrams in a browser on the intranet/internet.

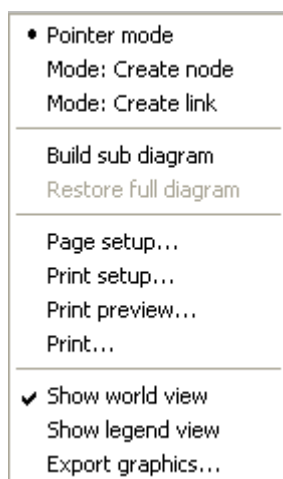
The method **ExportGraphicsToFile** of object VcGantt or the default context menu for the diagram lets you store the diagram to a file.

3.35 World View

The world view is an additional window that shows the diagram completely and, if switched on, also the histogram completely. A frame surrounds the diagram section currently displayed in the main window. If you move one of these frames, the corresponding section in the main window will move proportionally as soon as you release the mouse button. In a similar way, you can enlarge or reduce the display in the main window by zooming the frame in the world view. Vice versa, the position or the size of the frame will be modified when you scroll or zoom the section in the main window.



At run time, you can switch on/off the world view via the item **Show world view** of the default context menu.



On the **Additional Views** property page you can specify the properties of the World View. For details please read the chapter "Property Pages and Dialog Boxes", the "Additional Views" Property Page.

Beside, you can specify the properties of the World View by the API (**VcWorldView**).

3.36 Writing PDF files

Writing PDF files is only possible if an appropriate PDF printing driver is available. The drivers that are free of charge and those that are commercially available differ in their functionality and in the quality of the created PDF files.

Due to the lack of a consistent standard for the controlling of drivers, each printing driver has to be configured individually. The target path for the output file of many PDF printing drivers for instance is preset and can only be modified by altering the Windows registry, by editing INI files or by using driver-specific function APIs or COM objects.

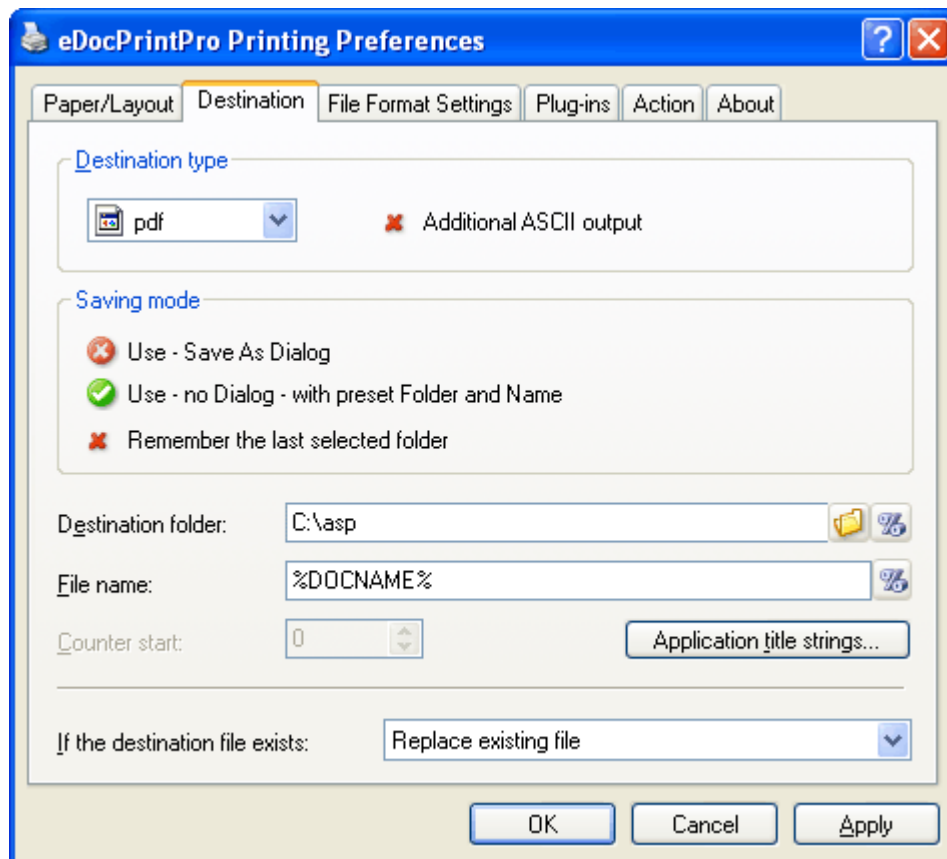
To be suitable a PDF printing driver has to fulfill the below requirements concerning controlling and print quality:

- Depending on the design of the application, it may be necessary that the driver offers the option of switching off all runtime dialogs and message boxes, in particular dialogs for setting file names and paths.
- If file names and paths shall not be set until runtime and if this is only possible by modifying entries of the Windows registry, the permissions of the user account have to be set accordingly.
- For the correct output of texts, Unicode support is needed.
- Fill patterns have to be displayed in sufficient quality. Please note that apart from bitmaps, transparencies cannot be displayed. In bitmaps however, unwanted artifacts may occur.
- The driver has to support vertical text output, otherwise the vertical annotation of date lines in VARCHART XGantt cannot be used.

The aforementioned requirements are fulfilled for instance by the printing driver included in the **Adobe Acrobat Suite** from version 6 onward [www.adobe.com] and the free driver **eDocPrintPro** [www.pdfprinter.at].

Below, please find an outline of the required steps to control the printing driver, using the example of **eDocPrintPro**:

- The dialog **Printing Preferences** can be accessed by the driver's settings in the control panel or by the driver's entry in Start/Programs or by the usual print dialog of an application. If necessary you can in that dialog select that the PDF file should be created without a dialog popping up and that the name of the target file is to be derived from the name of the document for instance. The required settings in **eDocPrintPro** then look as follows:



- In the program, the VcPrinter object of VARCHART XGantt should contain the below settings:

Example Code VB.NET

```
VcGantt1.Printer.PrinterName = "eDocPrintPro"
VcGantt1.Printer.DocumentName = "abc.pdf"
VcGantt1.PrintEx
```

Example Code C#

```
vcGantt1.Printer.PrinterName = "eDocPrintPro";
vcGantt1.Printer.DocumentName = "abc.pdf";
vcGantt1.PrintEx;
```

Very few printing drivers require a different program code:

Example Code VB.NET

```
VcGantt1.Printer.PrinterName = "Win2PDF"
VcGantt1.PrintToFile "abc.pdf"
```

Example Code C#

```
vcGantt1.Printer.PrinterName = "Win2PDF";
vcGantt1.PrintToFile "abc.pdf";
```

For further information concerning configuration and usage of **eDocPrintPro** please contact the producer.


4 Property Pages and Dialog Boxes

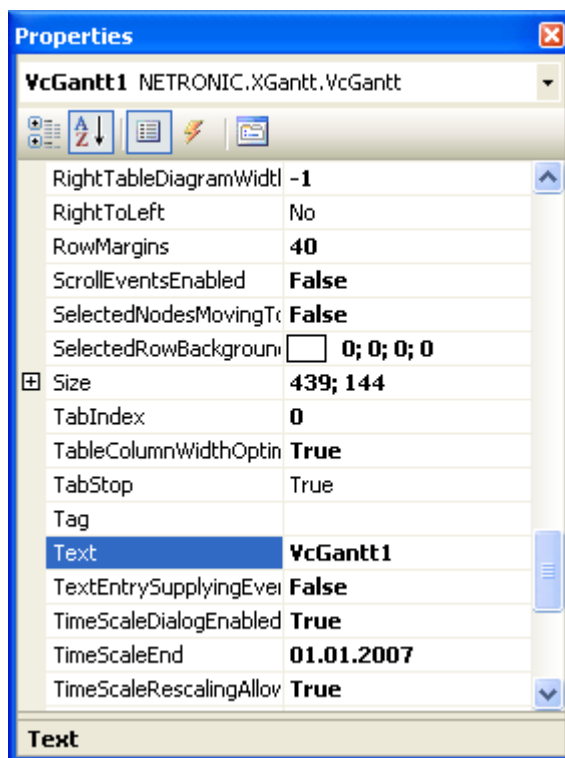
4.1 General Information

Property pages allow to configure VARCHART XGantt already at design time. There are two ways to get to the property pages:

- Press the right mouse button while the mouse pointer is on the control and select **Properties** from the context menu.

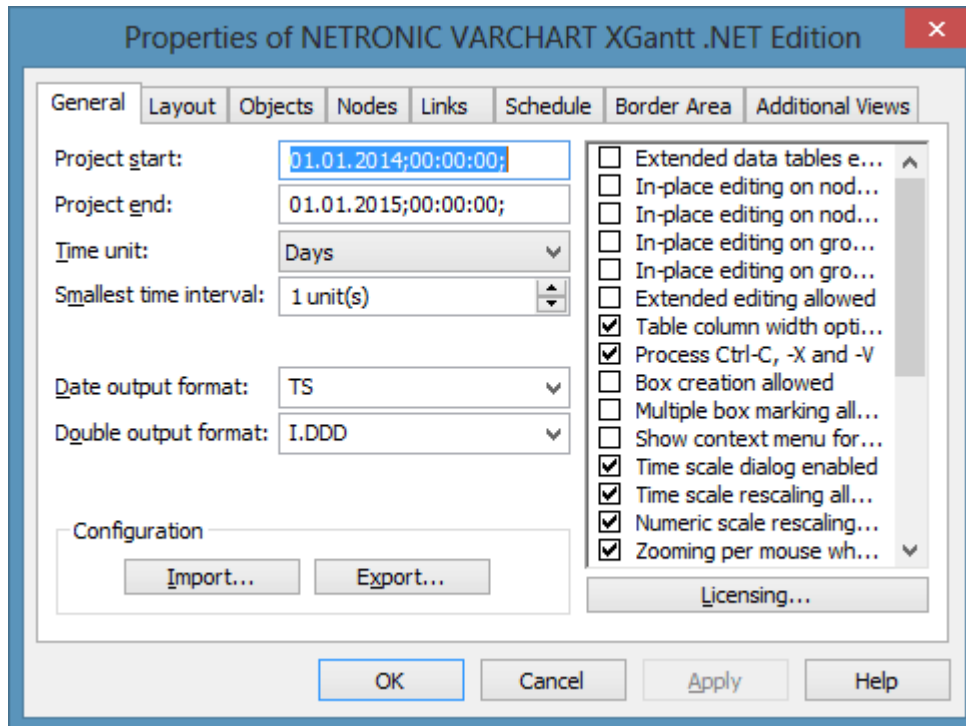
or

- In the **Properties** box of the control (to be invoked by the F4 key) click on the right icon in the icon bar .



More information about the functions of property pages and dialog boxes you can obtain by either clicking on the **Help** button or by pressing the **F1** key of your keyboard. This will open the corresponding online help file.

4.2 The "General" Property Page



On this property page you can enter the general settings of VARCHART XGantt.

Project start

Specify the default start date of the time scale. At run time, this value can be adapted to the current data by the property **TimeScaleStart** or by the method **OptimizeTimeScaleStartEnd**. The date format is "DD.MM.YYYY;hh:mm:ss;".

Project end

Specify the default end date of the time scale. At run time, this value can be adapted to the current data by using the property **TimeScaleEnd** or the method **OptimizeTimeScaleStartEnd**. The date format is "DD.MM.YYYY;hh:mm:ss;".

Note: The actual end date is not included. If you set **TimeScaleEnd** = "31.12.09" for example, the last day displayed will be December, 30st 2009.

Time unit

Select the time unit for your diagram. The value entered here will be used to calculate the duration (see Chapter "Important Concepts: Layer") and for the interactive modification and moving of the nodes in the diagram.

Example: If you select the time unit "Days" here, the nodes can only be moved in as many day steps as specified in the field **Smallest time interval**.

This feature can also be set by the property **VcGantt.TimeUnit**.

Smallest time interval

Specify how many time units are equivalent to one step.

Example: If you set the **Time Unit** to "Minutes" and the **Smallest time interval** to "30", the nodes can be moved in half-hour steps. This way a bar or layer will "snap" at a full hour and at half an hour.

This feature can also be set by the property **VcGantt.TimeUnitsPerStep**.

Date output format

From the combo box, select a format for your date output, or define a format. The format will also apply to the dialogs at runtime.

This feature can also be set by the property **VcGantt.DateOutputFormat**.

To compose the date you can use the following tokens:

- D: first letter of the day of the week (not adjustable)
- TD: Day of the Week (adjustable by using the event **VcTextEntrySupplying**)
- DD: two-digit figure for the day of the month: 01-31
- DDD: first three letters of the day of the week (not adjustable)
- M: first letter of the name of the month (not adjustable)
- TM: name of the month (adjustable by using the event **VcTextEntrySupplying**)
- MM: two-digit figure for the month: 01-12
- MMM: first three letters of the name of the month (not adjustable)
- YY: two-digit figure for the year
- YYYY: four-digit figure for the year
- WW: two-digit figure for the number of the calendar week: 01-53

TW:	text for "calendar week" (adjustable by using the event VcTextEntrySupplying)
Q:	one-digit figure for the quarter: 1-4
TQ:	name of quarter (adjustable by using the event VcTextEntrySupplying)
hh	two-digit figure for the hour in 24 hours format: 00-23
HH:	two-digit figure for the hour in 12 hours format: 01-12
Th:	Text of "o' clock" (adjustable by using the event VcTextEntrySupplying)
TH:	"am" or "pm" (adjustable by using the event VcTextEntrySupplying)
mm	two-digit figure for the minute: 00-59
ss:	two-digit figure for the second: 00-59
TS:	short date format, as defined in the regional settings of the windows control panel
TL:	long date format, as defined in the regional settings of the windows control panel
TT:	time format, as defined in the regional settings of the windows control panel

Note: Characters which are not to be interpreted as part of the date should be preceded by a backslash '\'. '\\' for instance results in '\'. The special characters: ':, /, -' and **blank** do not need '\' as a prefix.

Double output format

From the select box, please choose a format for the data type **Double**. You can choose between **I** (whole number), **I.DDD**, **I.DDDDDD** or **I,DDD**, **I,DDDDDD** (3 or 6 decimal digits) and **\$ I,III.DD** or **I.III,DD €** (two-digit currency).

This feature can also be set by the property **VcGantt.DoubleOutputFormat**.

Configuration

You can store the settings of the property pages to a configuration outside your project at any time, and load them when required. This is very useful if

you want to use previous settings again or you need the settings for different projects.

A configuration consists of two files of the same name that have different extensions, an ini- and an IFD file, which both are indispensable.

You can specify either a local file including the path or a URL.

An URL should be used as configuration file only if the configuration is specified during runtime by the API because only then the INI and IFD files will be loaded from the URL specified. If you specify a URL for configuration already at design time, the INI and IFD files will be downloaded, but they will be added to the project as a resource and be used at run time rather than loading the files directly.

How to save your current configuration:

Click on the **Export** button and enter a name for the INI file. An IFD file of the same name will be created automatically.

How to load a saved configuration:

Click on the **Import** button and select the file needed.

Extended data tables enabled

If you tick this box you can create and use up to 99 data tables, instead of merely the two default tables **Main data** and **Relations**. This option can also be set by the property **VcGantt.ExtendedDataTablesEnabled**.

In-place editing on nodes in table

Tick this option if in-place editing of node data (if grouping is switched on: of leaf node data) is to be allowed in the table. This feature can also be set by the property **VcGantt.InPlaceEditingOnNodesInTableEnabled**.

If to certain data fields in-place editing shall not be permitted, please don't select the option **editable** in the data definition.

In-place editing on nodes in diagram

Tick this option if in-place editing of node layers (if grouping is switched on: of leaf node layers) is to be allowed in the diagram. This feature can also be set by the property **VcGantt.InPlaceEditingOnNodesInDiagramEnabled**.

If to certain data fields in-place editing shall not be permitted, please don't select the option **editable** in the data definition.

In-place editing on groups in table

Tick this option if in-place editing of group node data is to be allowed in the table. This feature can also be set by the property **VcGantt.InPlaceEditing-OnGroupsInTableEnabled**.

If to certain data fields in-place editing shall not be permitted, please don't select the option **editable** in the data definition.

In-place editing on groups in diagram

Tick this option if in-place editing of group node layers is to be allowed in the diagram. This feature can also be set by the property **VcGantt.InPlace-EditingOnGroups InDiagramEnabled**.

If to certain data fields in-place editing shall not be permitted, please don't select the option **editable** in the data definition.

Extended editing allowed

Tick this box to use extended features to edit the table contents and to navigate. This feature can also be set by the property **VcGantt.Extended-EditingBehavior**.

- **Mark activities and enter new contents:**

When clicking on an activity not only the table line and the corresponding node in the diagram will be marked but you can also directly enter data into a data field.

Please take notice of the following:

When clicking in the **diagram**, the **first** field of the corresponding table line will be marked and will be ready for editing, no matter which field was marked before. By clicking on a different node, the marking will move accordingly and the first field of the corresponding line will be marked.

When clicking in the table area, the field hit will be edited.

For both procedures the following is valid:

You can move the marking by the arrow keys up/down or by the ENTER key and thus mark the previous/next line. If in the table area a field different to the first one should have been marked before, a corresponding selection will appear in the newly marked line. In an already marked table line, the arrow keys right/left will move the marking to the next/previous field, respectively.

Note: By pressing the ESC key, all markings will be undone.

- **Modify field contents**

To modify the contents of a table field you can either click on the field once more or press the F2 key.

There are some data types however which do not require this any more. You can modify date and time fields by clicking on the arrow button. For more information about the usage of the date dialog box please see chapter 4.40 The "Specify Date Lines" Dialog.

The value of numeric data fields may be increased or decreased by clicking on the corresponding arrow buttons.

Note: By pressing the ESC key you can leave the edited fields without saving the modifications.

- **Insert new table lines**

By the INS key you can insert a new line above the currently marked line. If no line was marked, a new line will be inserted at the end of the table.

Table column width optimization allowed

If you tick this box at run time, a double-click on a parting line between two columns will cause that the width of the column on the left will be adapted automatically to the length of the texts which it contains.

This feature can also be set by the property **VcGantt.TableColumnWidthOptimizationAllowed**

Process Ctrl-X, -C and -V

If you activate this check box, the key combinations Ctrl+C, Ctrl+X and Ctrl+V will be translated automatically into the clipboard commands **CopyNodesToClipboard**, **CutNodesToClipboard** and **PasteNodesFromClipboard**, respectively. You can revoke this feature by leaving the check box blank, in order to avoid interfering with menu commands in Visual Basic. This feature can also be set by the property **VcGantt.CtrlCXVProcessingEnabled**.

Box creation allowed

If you tick this box, the user can create new boxes at run time. To do so, select the **Mode: Create box** or set **InteractionMode** to **VcCreateBox**.

This feature can also be set by the property **VcGantt.BoxCreationAllowed**.

Multiple box marking allowed

By ticking this box, the user can select several boxes at the same time by clicking on them without having to keep the CTRL-key pressed. This option is disabled by default.

This feature can also be set by the property **VcGantt.MultipleBoxMarking-Allowed**.

Show context menu for boxes

Tick this option to enable the context menu for boxes at runtime.

This feature can also be set by the property **VcGantt.ContextMenuFor-BoxesEnabled** .

Time scale dialog enabled

Activate this option if the **Edit TimeScale** dialog box is to appear when the user double-clicks on the time scale.

This feature can also be set by the **VcGantt.TimeScaleDialogEnabled** property.

Time scale rescaling allowed

You must activate this option if you want to allow the user to interactively modify the resolution of the timescale.

This feature can also be set by the property **VcGantt.TimeScaleRescaling-Allowed**.

Numeric scale rescaling allowed

If you tick this box, the user can rescale the numerical scale of the histogram.

This feature can also be set by the property **VcGantt.NumericScale-RescalingAllowed**.

Zooming by mouse wheel allowed

Tick this option if zooming by mouse wheel is to be allowed. For zooming the user has to press the Ctrl key and roll the mouse wheel.

This feature can also be set by the property **VcGantt.ZoomingPerMouse-WheelAllowed**.

VcToolTipTextSupplying events

Tick this option if the event **VcToolTipTextSupplying** is to be activated. It also can be set by the **ToolTipTextSupplyingEventEnabled** property. The event **VcToolTipTextSupplying** lets you set the text strings to be displayed as tooltip texts with the objects.

Scroll events enabled

By ticking this box, you may enable or disable the scroll events. This feature can also be set by the **VcGantt.ScrollEventsEnabled** property.

Note: The scroll events are **disabled** by default.

VcTextEntrySupplying events

By ticking this box you can trigger the **VcTextEntrySupplying** event. This event lets you modify the texts of context menus, dialog boxes and error messages that occur during run time, for example for translation into different languages.

This feature can also be set by the property **VcGantt.TextEntrySupplying-EventEnabled**.

Events security check

Tick this option if a security check for the event and **VcNodeModifying** is to be performed. Then in these events the set calls to the corresponding object types will be suppressed.

This feature can also be set by the property **VcGantt.EventsSecurityCheck**.

Automatic reduction of row heights

This option controls the way of calculating the row height in the diagram. If the check box is not ticked, the vertical offsets of the layers are applied by using an imaginary zero line in the vertical center of a node line. To keep the zero line always in the center of the row, it thus may happen that either the top or the bottom row margin will seem rather broad. The layers with a vertical offset of 0, however, stay always vertically centered.

If the check box is ticked, the imaginary zero line is still used but its position is no longer necessarily in the center of the row but so that the row height is as low as possible. Thus it may happen that layers with a vertical offset of 0

are not on the same level as the vertical centered text of the corresponding table row.

This feature can also be set by the property **VcGantt.RowHeightReduction-Enabled**.

Font anti-aliasing

This option allows to set anti-aliasing to font characters. If the legibility of certain fonts - in particular non- latin ones - changes for the worse, the option should be switched off.

The anti-aliasing with GDI+ has yet another effect: regardless of the selected zoom factor, texts keep their relative dimension so that the number of characters that fits in a table field will always be the same. If the option is switched off the settings of the operating system are applied instead (the settings can be found in the **Control Panel**, dialog box **Display**, Tab **Appearance: Effects**). Thus, if the option **Smooth edges** is switched on in the **Control Panel**, the texts might still be anti-aliased, notwithstanding the settings of the **General** property page. In this case, at some zoom levels more text could be visible than at others, since the native edge smoothing does not guarantee that the same relative dimension is always kept.

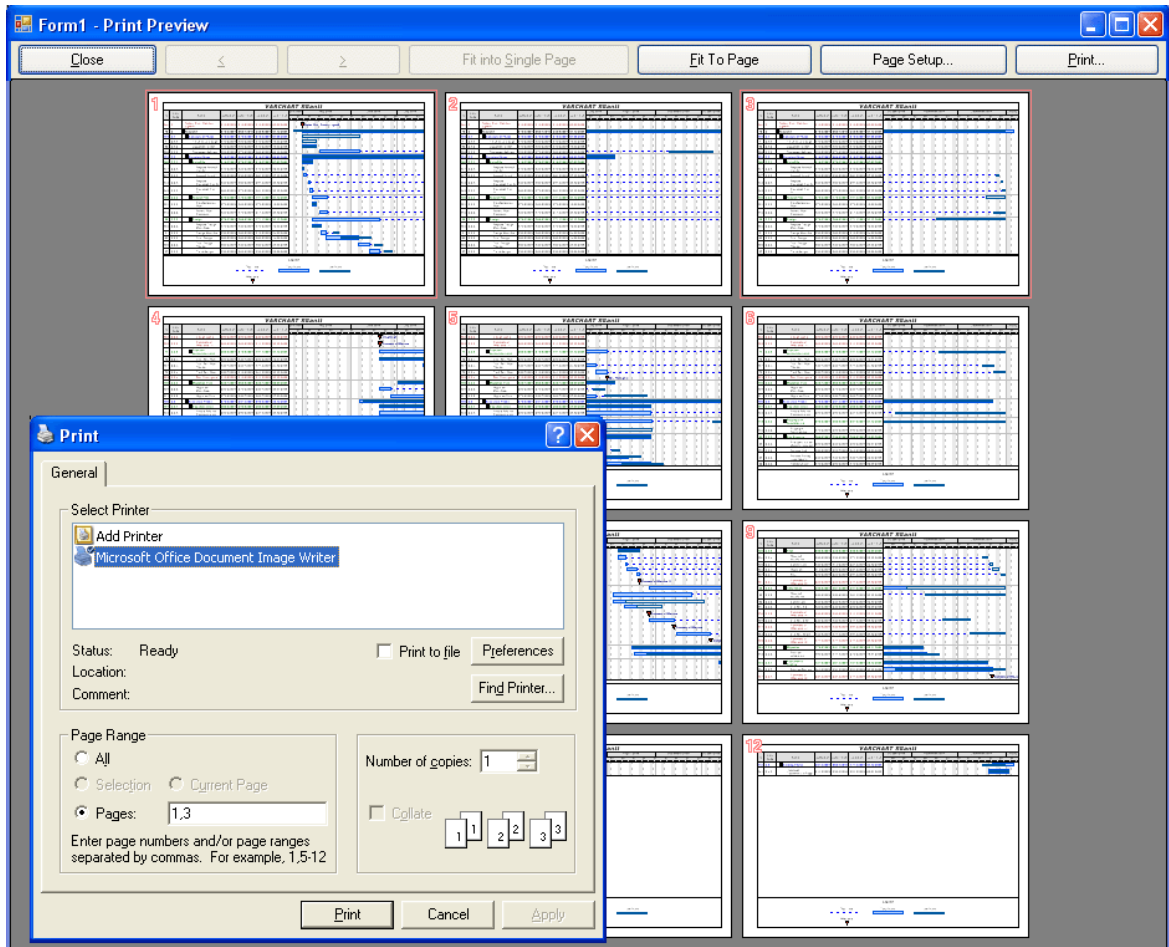
This feature can also be set by the property **VcGantt.FontAntiAliasing-Enabled**.

Use PrintDlgEx dialog

If you tick this check box, the item **Printer setup** will be missing at runtime both in the print preview and in the context menu because the corresponding dialog is now to be found in the (extended) **Print** dialog. If a new project is created, this option is ticked by default whereas in already existing projects it is ticked off for compatibility reasons.

In the print preview you can now select pages by a left click (one page) or by using CTRL + left click (more pages). The selected pages are then preset already as pages to be printed in the **Print** dialog.

If you invoke the **Print** dialog from the print preview, all pages have a page number to make the selection of pages easier.



This feature can **not** be set by an API property.

Rounded link slants

If you activate this check box, the slants of links of the routing type **vcLRTOrthogonalDistinguishable** are displayed as quarter circles instead of straight lines. This feature can also be set by the VcGantt property **RoundedLinkSlantsEnabled**.

Consider relation type on node dragging

Tick this box if you want the phantom lines that represent the links to be displayed indicating their type if dragged, and if links are switched on at all. The phantom lines will not start off from the center of the node, but from the left and right side of the node.

This feature can also be set by the **VcGantt.ConsiderLinkRelationTypes-OnNodeDragging** property.

Optimization of groups on interactions

If this property is set to true, the nodes of the target group automatically are optimized on interactions such as creating nodes, moving nodes or modifying their start or end date, if they had been in the optimized state of display before. If this property is set to false, on the interactions mentioned the node will be placed at the cursor, if this doesn't cause nodes to overlap. If it does, the node will be placed with other nodes in the next line, if this doesn't cause overlaps. If it does, a new line will be created below the one where the cursor is and the node will be put there.

This feature can also be set by the **VcGantt.GroupOptimizationOn-InteractionsEnabled** property

Wait cursor enabled on time-critical operations

Tick this box if you want to set us an internal wait cursor on time-critical operations.

This feature can also be set by the **VcGantt.WaitCursorEnabled** property

Panning mode allowed

Tick this box to be able to move certain screen sections at runtime. The contextmenu will then show the additional item **Panning mode**.

Activating the panning mode will apply to **all** view components by default. The **VcGantt.VcViewComponent** property allows to set the panning mode for certain selected components only.

This feature can also be set by the **VcGantt.PanningModeAllowed** property.

Selection via rubber rect allowed

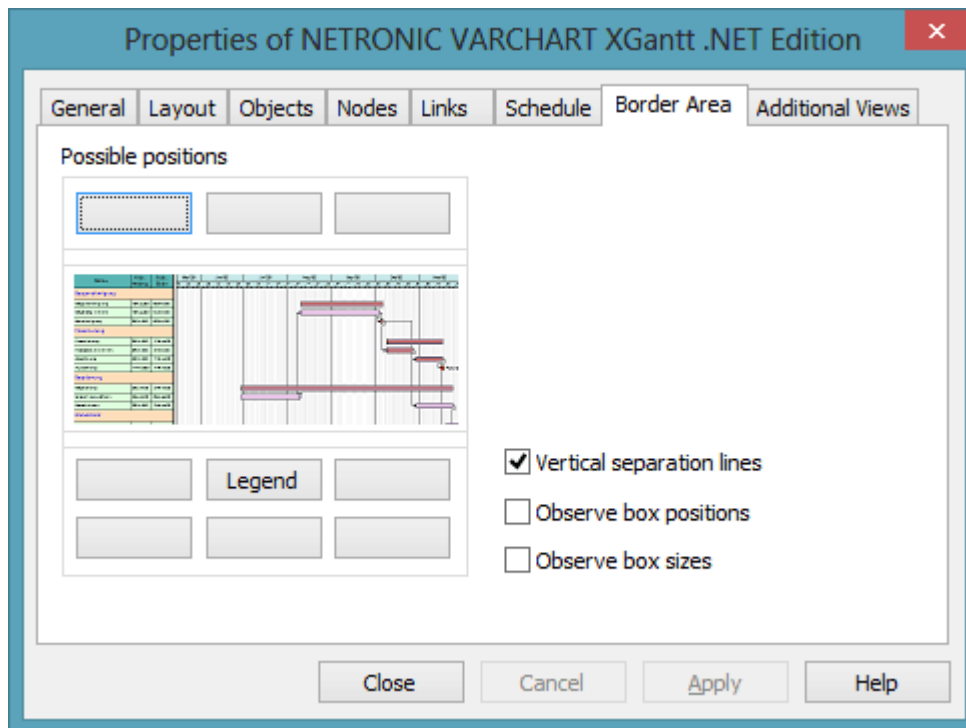
This option allows to enable/disable the selection of nodes by rubber rectangle.

This feature can also be set by the **VcGantt.AllowSelectionViaRubberRect** property.

Licensing

Press this button to get to the **Licensing** dialog box. For further information see chapter **Licensing**.

4.3 The "Border Area" Property Page



Possible positions

There are three areas above and six areas below the diagram which you can use for texts, graphics or a legend. These areas are displayed only in the print preview and in the print output. Click on one of the buttons above or below the diagram to get to the **Specification of texts, graphics and legend** dialog box.

Vertical separation lines

Activate this check box, if the areas for texts, graphics or the legend are to be separated by vertical lines.

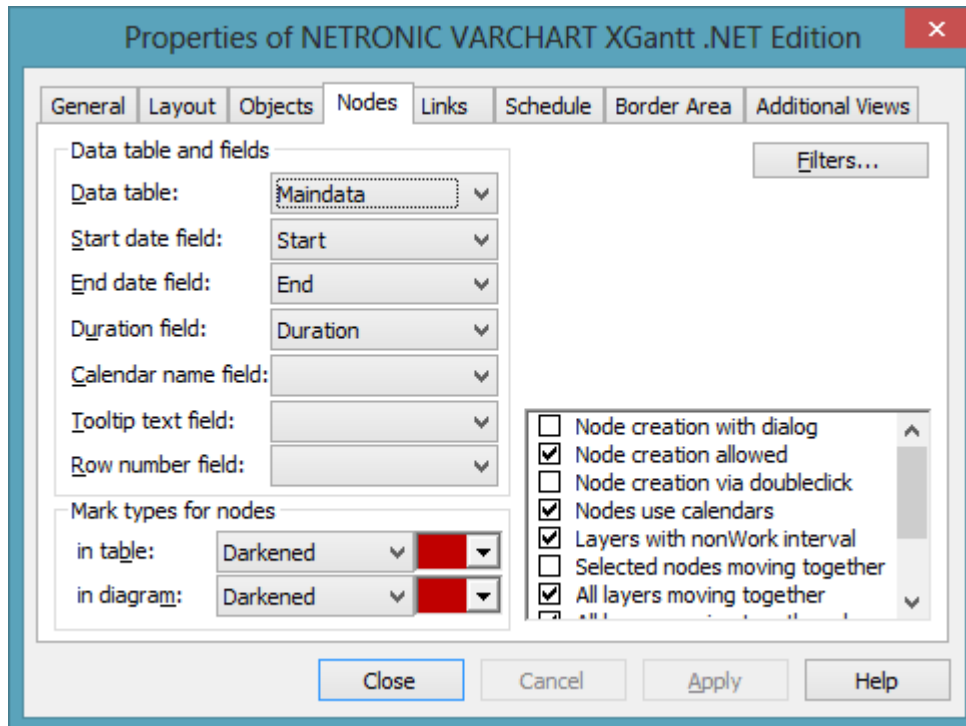
Observe box position

Activate this check box, if the box positions are to be observed as exactly as possible. Alternatively, the available space will be divided proportionally between all elements in the row.

Observe box size

Activate this check box, if the box sizes are to be observed as exactly as possible. The chart may be enlarged and/or the texts in the boxes may be clipped.

4.4 The "Nodes" Property Page



Data table

Select the data table which shall be used for the representation of the nodes.

This feature can also be set by the property **VcGantt.NodesDataTable-Name**.

Start date field

Please select the data field to store the start date of an interactively created node. Only date fields are offered in the combo box.

This feature can also be set by the property **VcGantt.NodeStartDateData-FieldIndex**.

End date field

Please select the data field to store the finish of an interactively created node. Only date fields are offered in the combo box.

This feature can also be set by the property **VcGantt.NodeEndDateData-FieldIndex**.

Duration field

Please select a data field to store the duration of an interactively created layer. Only numeric data fields are available.

This feature can also be set by the property **VcGantt.NodeDurationDataFieldIndex**.

Calendar name field

If you wish to use an individual calendar for a node, you can select the data field to store the name of the calendar. For this, the check box **Nodes use calendar** needs to be activated. Beside, the calendars need to have been created before loading the nodes.

This feature can also be set by the property **VcGantt.NodeCalendarNameDataFieldIndex**.

Row number field

Please select a data field which stores the row number of the node. The modifications only become effective after having carried out an update by using the method **VcGantt.UpdateRowNumberFields**

This feature can also be set by the property **VcGantt.NodeRowNumberDataFieldIndex**.

Tooltip text field

The data field specified here is only important for the VMF export. If you show a VMF file by the WebViewer software and there right-click on a node, the contents of the selected data field will be shown as a tooltip. No further settings are required.

To show tooltips in your application, please activate the check box **VcToolTipTextSupplying events** on the **General** property page or set the VcGantt property **ToolTipTextSupplyingEventEnabled** = True and specify the text to be displayed in the **VcToolTipTextSupplying** event.

Mark type for nodes in table

Use the left field to specify whether node marks are used in the table and, if desired, select the type of node marking from the list:

- No Mark
- Surround inside

- Invert
- Darken (by 25%)
- Brighten (by 25%)
- Pickmarks inside

The field to the right lets you select a color for the marking type.

Mark type for nodes in diagram

Use the left field to specify whether node marks are used in the diagram and, if desired, select the type of node marking from the list:

- No Mark
- Surround
- Surround inside
- Invert
- Darken (by 25%)
- Brighten (by 25%)
- Pickmarks
- Pickmarks inside

The field to the right lets you select a color for the marking type.

Filters

This button lets you open the **Administrate Filters** dialog box. The filter settings that pre-select the nodes can only be entered at runtime by the property **ActiveNodeFilter** of the object **VcGantt**.

Node creation with dialog

If you tick this box, the **Edit Data** dialog box will open automatically when the user creates a new node interactively. After having created a node, the **Edit Data** dialog box can be invoked (even if this option is disabled) either by double-clicking this node or by the corresponding item of the context menu.

This feature can also be set by the property **VcGantt.NodeCreationWithDialog**.

Node creation allowed

If you activate this option the user will be able to create new nodes interactively in an open project. New nodes can be created interactively in the **Creation Mode** (right-click in the diagram area, context menu) or, if the **Node creation via double-click** box was ticked, by double-clicking in the corresponding location of the diagram.

This feature can also be set by the property **VcGantt.NodeCreationAllowed**.

Node creation via double-click

If you tick this box new nodes can be created by double-clicking. A new node created by a double-click will be inserted at the current cursor position and has a duration of a single time unit.

This feature can also be set by the property **VcGantt.NodeCreationViaDoubleClick**.

Nodes use calendars

Tick this box to assign calendars to the nodes. Assigning calendars to nodes has the following effects: The starts and ends of the activities are not positioned on workfree days. The workfree periods are considered when calculating the duration of the activities. Currently, the default is a five-day calendar ("BaseCalendar").

This feature can also be set by the property **VcGantt.NodesUseCalendars**.

If no individual calendar has been assigned per node, the calendar which was defined as active in the CalendarCollection is used.

Layers with nonWork interval

Please activate this check box to have workfree intervals highlighted. They will be displayed as was specified in the **Edit layer** dialog (only rectangle layers).

This feature can also be set by the property **VcGantt.LayersWithNonWorkInterval**

Selected nodes moving together

Please activate this check box to enable all marked nodes to be moved. If you leave it deactivated, only single layers or nodes (depending on whether the

All layers moving together check box was ticked) can be moved by the mouse, even if several nodes have been marked.

This feature can also be set by the property **VcGantt.SelectedNodesMoving-Together**.

All layers moving together

Please tick this check box to move all layers of a marked node in one go. A node can be marked by a mouse click on one of its layers.

If this check box is not ticked, the layers of a marked node can only be moved individually. For moving all layers of the node, please keep the SHIFT key pressed while dragging the node. (For this, the **Move layers as node when shift key pressed allowed** needs to be ticked).

This feature can also be set by the property **VcGantt.AllLayersMoving-Together**.

All layers moving together always

If you tick this check box, all layers of a node can be moved in one go without having to be marked before.

This feature can also be set by the property **VcGantt.AllLayersMoving-TogetherAlways**.

Moving layers as node when shift key pressed allowed

If this box is ticked, all layers of a node can be moved in one go if the Shift key is being pressed while dragging. This feature can also be set at run time by the VcGantt property **VcGantt.MovingLayersAsNodeWithShiftKey-Allowed**.

Use snap targets in interactions

If this box is ticked, the snap target functionality can be used while dragging a node/layer, meaning to specify whether a node/layer "snaps" at the defined snap targets of the respective objects. This feature can also be set at run time by the VcGantt property **UseSnapTargetsInInteractions**.

Show snap lines

Ticking this box enables snap lines to be shown while nodes are being resized or dragged with the snap target mode switched on. These lines help to better recognize the defined snap targets.

This feature can also be set at run time by the **VcGantt** property **ShowSnapLines**.

Show snap targets

Ticking this box enables snap markings to be shown while nodes are being resized or dragged with the snap target mode switched on. These lines help to better recognize the defined snap targets.

This feature can also be set at run time by the **VcGantt** property **ShowSnapMarkings**.

Moving a node vertically via diagram allowed

Tick this box if you want the user to be able to change the order of the activities or their group affiliation by dragging nodes from one row to another in the diagram area. If a node consists of more than one layer, the Shift key needs to be pressed while dragging vertically.

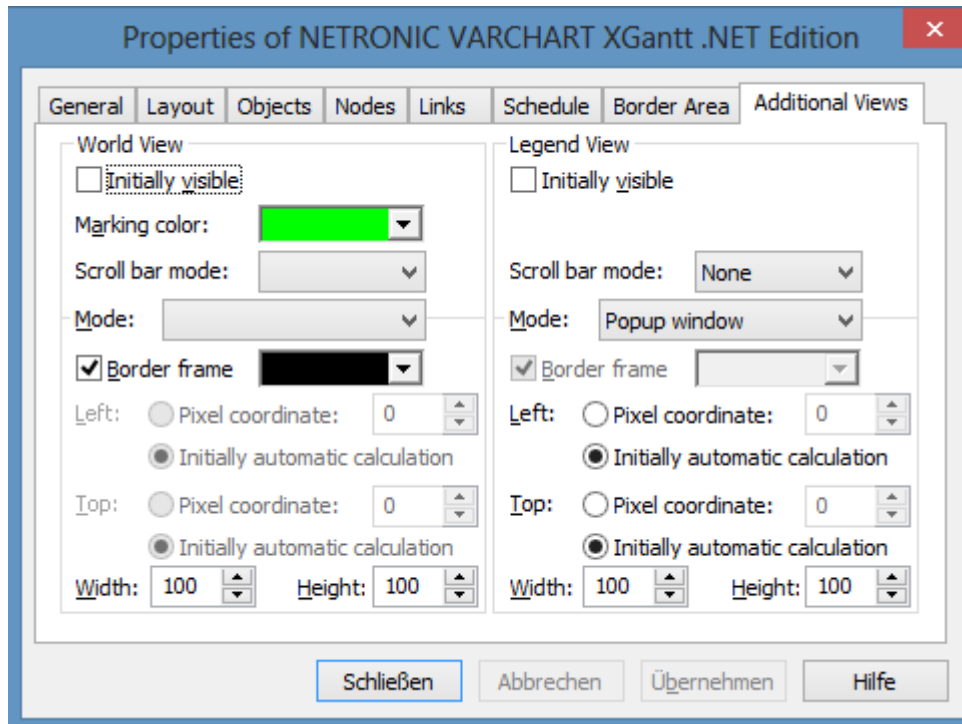
This feature can also be set at run time by the **VcGantt** property **VerticalNodeMovementAllowed**

Moving a node vertically via table allowed

Tick this box if you want the user to be able to change the order of the activities or their group affiliation by dragging nodes from one row to another in the table area. If a node consists of more than one layer, the Shift key needs to be pressed while dragging vertically.

This feature can also be set at run time by the **VcGantt** property **VerticalNodeMovementViaTableAllowed**

4.5 The "Additional Views" Property Page



On this property page you can set the properties of the "world view" and the legend view..

Both views are additional small windows.

The world view displays the diagram completely. Two frames in it indicate the sections actually displayed in the main window. One of them shows the section in the Gantt Graph, the other one shows the histogram section.

The legend view lets you display a legend.

At run time, you can switch on or off both views in the default context menu by clicking **Show world view** or **Show legend view** respectively. You can alternatively use the **Close** button of the title bar to switch off either view.

The description of the possible settings which you find below, is valid for both views, if not stated otherwise.

Initially visible

Activate this check box if the view is to be visible when the program is started.

This property can also be set by the API calls **VcWorldView.Visible** and **VcLegendView.Visible**

Marking color (only World View)

Select the line color of the rectangle that indicates in the World View the currently selected section.

This property can also be set by the API calls **VcWorldView.MarkingColor** and **VcLegendView.MarkingColor**.

Scroll bar mode

You can select a mode of displaying scrollbars. By using scrollbars, empty areas are avoided and there is more space for displaying the chart or the legend.

- **None:** The view always displays the complete chart or legend. Thus empty areas may occur if the view's proportions do not correspond to those of the chart/the legend.
- **Horizontal:** A horizontal scrollbar is displayed if required.
- **Vertical:** A vertical scrollbar is displayed if required.
- **Automatic:** A horizontal or a vertical scrollbar is displayed if required.

This property can also be set by the API calls **VcWorldView.ScrollBarMode** and **VcLegendView.ScrollBarMode**.

Mode

You can select a mode of displaying the the view:

- **Fixed at left side:** The view appears on the left side of the control window. The width can be varied, whereas the position and the height are fixed.
- **Fixed at right side:** The view appears on the right side of the control window. The width can be varied, whereas the position and the height are fixed.
- **Fixed at top side:** The view is displayed in the top section of the control window. The height can be varied, whereas the position and the width are fixed.
- **Fixed at bottom side:** The view is displayed in the bottom section of the control window. The height can be varied, whereas the position and the width are fixed.
- **Position not fixed:** The view is a subwindow of the parent window of the control. It can be positioned anywhere and has no fixed size. The parent window can be modified by the property **VcWorldView.ParentHWnd**.

- **Popup window:** The view is a popup window that has its own frame. The user can modify its position and extension, open it by using the default context menu, and close it by the **Close** button in the frame.

This property can also be set by the API calls **VcWorldView.Mode** and **VcLegendView.Mode**.

Border frame

*Not active if the mode **Popup window** has been selected.* Activate this check box if the view is to have a frame and select a color in the drop down list..

This options can also be set by the API calls **VcWorldView.Border** and **VcWorldView.Border.Color** or **VcLegendView.Border** and **VcLegendView.Border.Color**

Left

*Only active if the mode **Position not fixed** or **Popup window** has been selected.* Select the left position of the view. There are two possibilities:

1. Specify a **Pixel coordinate** value. Note that this is a system coordinate.
2. Select the **Initially automatic calculation** option.

This property can also be set by the API calls **VcWorldView.Left** and **VcLegendView.Left**

Top

*Only active if the mode **Position not fixed** or **Popup window** has been selected.* Select the top position of the view. There are two possibilities:

1. Specify a **Pixel coordinate** value. Note that this is a system coordinate.
2. Select the **Initially automatic calculation** option.

This property can also be set by the API calls **VcWorldView.Top** and **VcLegendView.Top**

Width

*Not active if the mode **Fixed at left/right side** has been selected.* Select the horizontal extension of the view. Note that the pixel coordinate is a system coordinate.

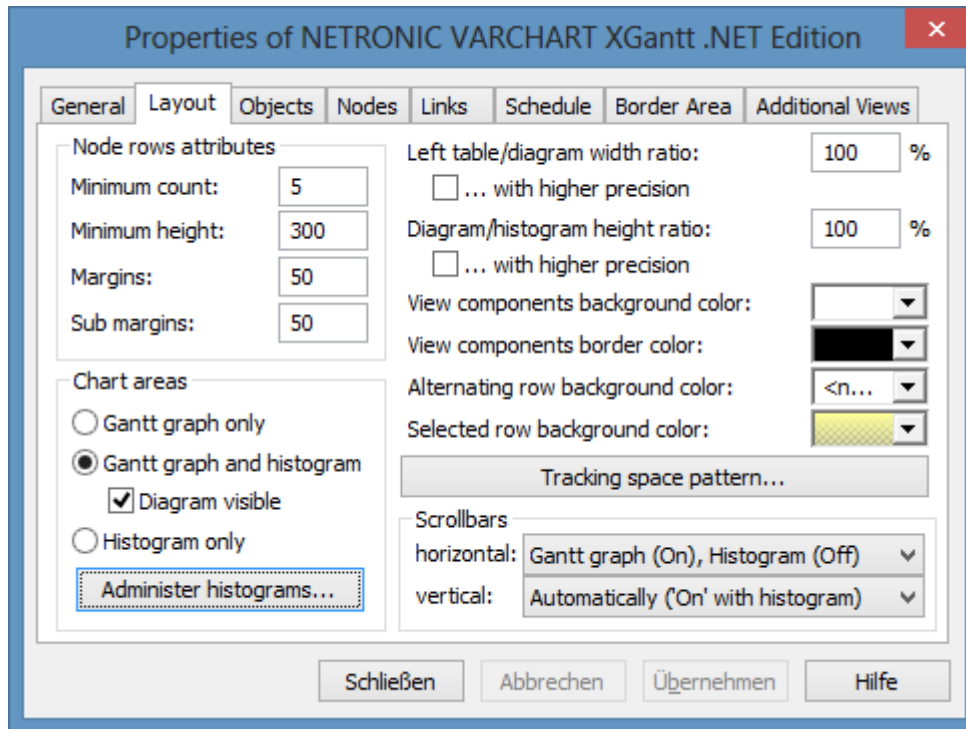
This property can also be set by the API calls **VcWorldView.Width** and **VcLegendView.Width**

Height

*Not active if the mode **Fixed at left/right side** has been selected.* Select the vertical extension of the view. Note that the pixel coordinate is a system coordinate.

This property can also be set by the API calls **VcWorldView.Height** and **VcLegendView.Height**

4.6 The "Layout" Property Page



On this property page you can establish and modify the layout of the chart.

Minimum count

Specify how many node rows are to be displayed in the diagram area at the program start.

This feature can also be set by the property **VcGantt.NumberOfInitialRowCount**.

Minimum height

Specify the minimum height of the node rows in 1/100 mm. This property can also be set at run time by the property **MinimumRowHeight** of the **VcGantt** object. The values allowed to be set range between 2 and 1000.

The minimum row height only takes effect if there is no activity in the row or if existing activities do not exceed the minimum row height. In all other cases the row height automatically adapts to the space required by the activities.

This feature can also be set by the property **VcGantt.MinimumRowHeight**.

Margins

Specify the minimum vertical spacing between the node and the upper or lower node row border in 1/100 mm.

This feature can also be set by the property **VcGantt.RowMargins**.

Sub margins

This property lets you set or retrieve the vertical width between the sub rows. The sub rows only exist if groups are optimized and nodes of this group are arranged in several sub rows to prevent them from overlapping.

This feature can also be set by the property **VcGantt.SubRowMargins**.

Chart areas

Specify what the diagram is supposed to display:

- the Gantt diagram only
- the Gantt diagram and the histogram (a check box allows switching on or off the visibility of the diagram)
- the histogram only.

Administer histograms

The **Administer Histograms** dialog will appear.

Left table/diagram width ratio

Specify the ratio (in %) of the table width to the width of the total diagram (table area plus diagram area) at the start of the program. In order to display the table completely on the start, enter the value "-1".

This feature can also be set by the property **VcGantt.LeftTableDiagram-WidthRatio**.

...higher precision

Activate this property to enable the usage of the more accurate methods **Left-TableDiagramWidthRatioEx** and **RightTableDiagramWidthRatioEx** or the event **VcTableWidthChangingEx** that all return a value of the type "Double" to calculate the ratio between table and diagram.

If this property is not activated, the methods **LeftTableDiagramWidthRatio** and **RightTableDiagramWidthRatio** or the event **VcTableWidthChanging** will be used.

This feature can also be set by the **VcGantt.UseHigherTableDiagramWidthRatioPrecision** property.

Diagram/histogram height ratio

Specify the ratio (in %) of the height of the diagram area (histogram excluded) to the height of the histogram at the start of the program. In order to display the histogram completely on the start, set the value "-1".

This feature can also be set by the property **VcGantt.DiagramHistogramHeightRatio**.

...with higher precision

Tick this box to enable the usage of the more accurate method **DiagramHistogramHeightRatioEx** or the event **VcHistogramHeightChangingEx** that return a value of the type "Double" to calculate the width ratio between diagram and histogram.

If this property is set to the default value "False", the method **DiagramHistogramHeightRatio** or the event **VcHistogramHeight** are used.

This feature can also be set by the **VcGantt.UseHigherDiagramHistogramHeightRatioPrecision** property.

View components background color

This field lets you select the diagram background color. If you combine this property with the **Alternating row background color**, you can generate a color pattern that alternates linewise.

This feature can also be set by the property **VcGantt.DiagramBackgroundColor** or **VcGantt.ViewComponentsBackgroundColor**.

View components border color

This field lets you select the frame color for all panes at a time.

This feature can also be set by the property **VcGantt.ViewComponentsBorderColor**.

Alternating row background color

This field lets you set a second background color to the diagram, which alternates linewise with the **Diagram background color**.

This feature can also be set by the property **VcGantt.DiagramAlternating-RowBackgroundColor**.

Selected row background color

This field lets you set a background color to the selected row of the diagram.

This feature can also be set by the property **VcGantt.SelectedRowBackgroundColor**.

Tracking space pattern

This button opens the dialog **Edit Pattern Attributes** where you can specify the layout of the the free area, sometimes showing up briefly at the top or bottom margin during LiveUpdate interactions.

This feature can also be set by the according properties **VcGantt.Tracking-SpaceBackgroundColor**, **VcGantt.TrackingSpacePattern** und **VcGantt.-TrackingSpacePatternColor**.

Scrollbars

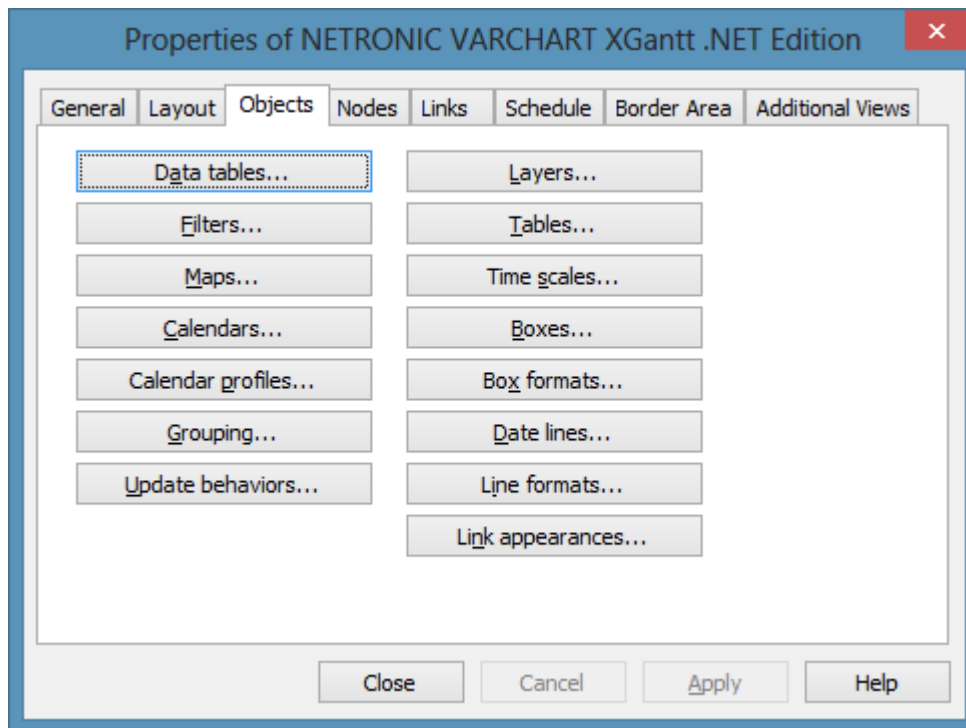
By these options you can set the horizontal and vertical scroll bars. For the horizontal scroll bar, you can choose between the below options:

1. **Gantt graph (on), Histogram (off)** - the horizontal scroll bar is located between the Gantt graph and the histogram
2. **Gantt graph (off), Histogram (on)** - the horizontal scroll bar is located below the histogram
3. **None** - there is no horizontal scroll bar.

For the vertical scroll bar, you can choose between the below options:

1. **Automatically (but 'On' with histogram)** - a vertical scroll bar will be switched on right of Gantt graph if required; another one is always on right of the histogram.
2. **On** - both, the vertical scroll bar right of the Gantt graph and the one right of the histogram are switched on
3. **Off** - both vertical scroll bars are switched off.

4.7 The "Objects" Property Page



Data tables

Opens the dialog **Administrative Data Tables**.

Filters

Opens the **Administrative Filters** dialog box.

Maps

Opens the dialog **Administrative Maps**.

Calendars

Opens the dialog **Specify Calendars**.

Calendar profiles

Opens the dialog **Administrative Calendar Profiles**.

Grouping

Opens the dialog **Grouping**.

Update behaviors

Opens the dialog **Administrate update behaviors**.

Layers

Opens the **Specify Bar Appearance** dialog box.

Tables

Opens the **Specify Table** dialog box.

Time scales

Opens the **Specify Time Scale** dialog box.

Boxes

Opens the dialog **Administrate Boxes**.

Box formats

Opens the dialog **Administrate Box Formats**.

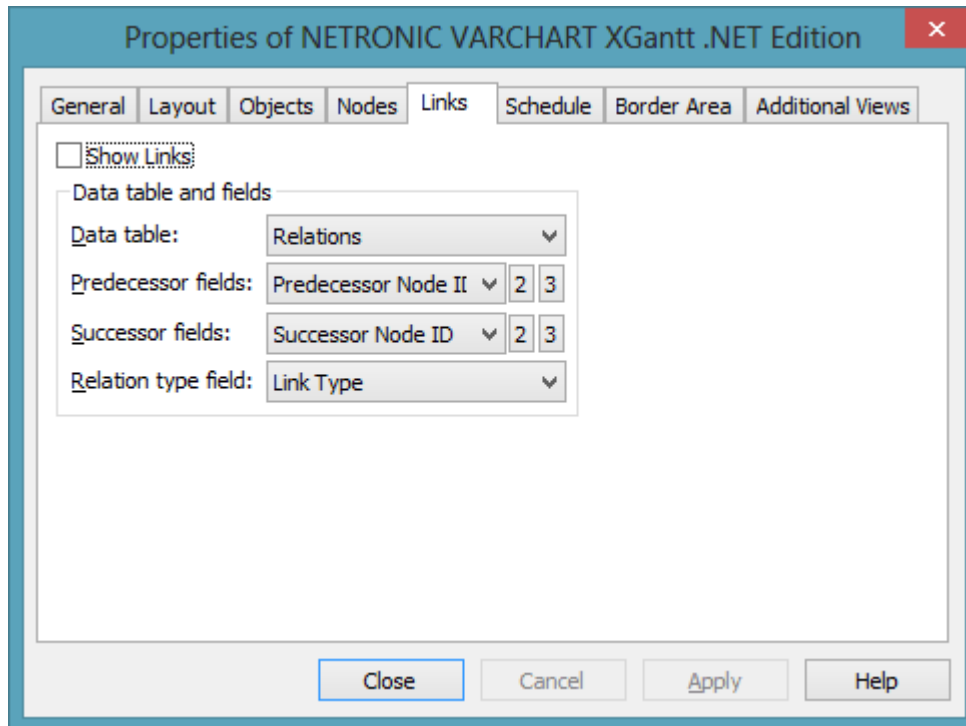
Date lines

Opens the **Specify Date Lines** dialog box.

Line formats

This button lets you open the dialog **Administrate Line Formats**.

4.8 The "Links" Property Page



This property page lets you display links between nodes and establish and modify the appearance of the links.

Show Links

This check box lets you specify whether links and phantom lines representing the links while dragging are to be displayed. This feature can be also set by the API property **VcLinkAppearance.Visible** - but only for the links, not for the lines.

Data table

Select a data table which contains the fields for the relations. This feature can also be set by the property **VcGantt.LinksDataTableName**.

Predecessor field

This field lets you set the data field or fields from the data table selected above to which the identification of the predecessor node of the link is/are stored.

This feature can also be set by the property **VcGantt.LinksPredecessor-DataFieldIndex**.

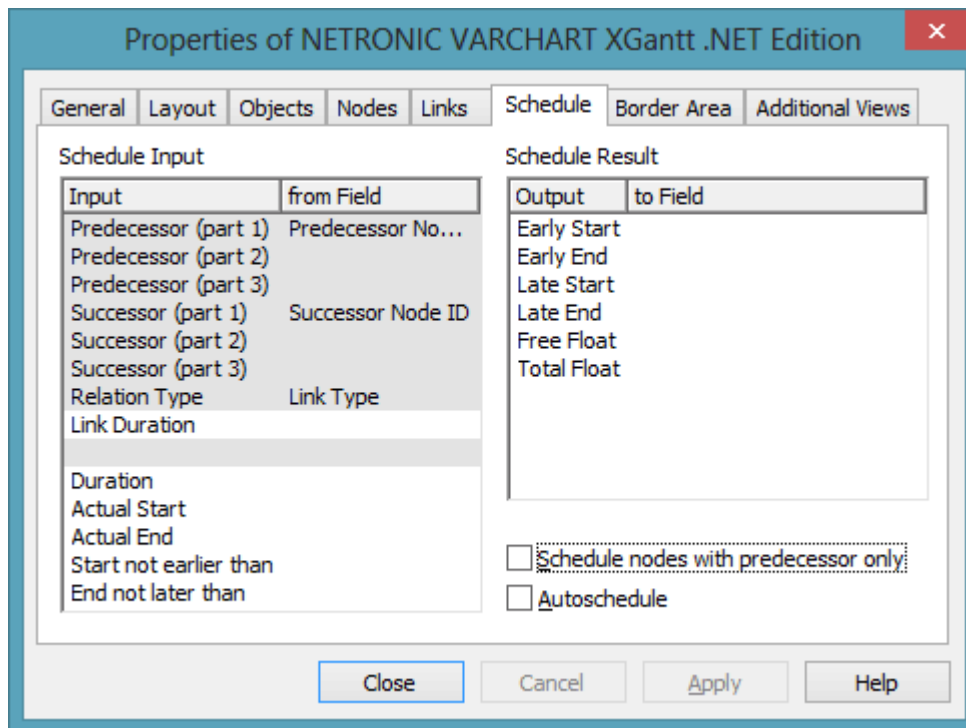
Successor field

This field lets you set the data field or fields from the data table selected above to which the identification of the successor node of the link is/are stored. This feature can also be set by the property **VcGantt.Links-SuccessorDataFieldIndex**.

Relation type field

Select the data field that contains the relation type. This feature can also be set by the property **VcGantt.LinkTypeDataFieldIndex**.

4.9 The "Schedule" Property Page



This property page lets you adapt the date calculation settings of VARCHART XGantt to your interface by specifying the data fields that you want to use for the input (**Schedule Input**) and output (**Schedule Result**) of the scheduler (also see "Important Concepts: Scheduling").

Schedule Input

Please select for each entry of the column, from which field its contents is to be loaded. The scheduler uses the data fields of the data tables of nodes and links previously set. The calculations of the scheduler are based on the project start, their logic dependencies and the project start given. The fields **Predecessor** and **Successor** cannot be edited by the **Schedule Input** table. They merely display the settings of the **Links** property page.

Schedule Result

Specify for each result to which field it is to be stored. The scheduler stores only to data fields of the **Maindata** table. The early/late start and end dates plus the total float and free float are calculated from the duration of the activities, the logical dependencies and the project start.

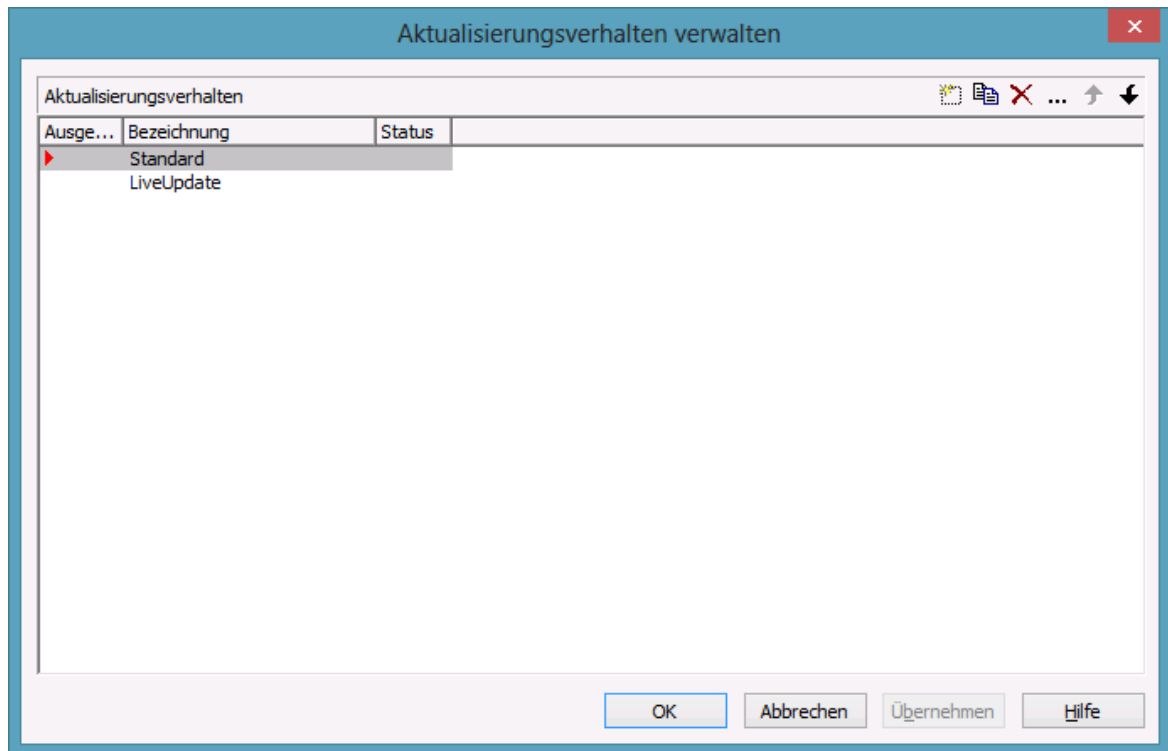
Schedule nodes with predecessor only

If you activate this check box, only those nodes will be scheduled that have a predecessor node, otherwise all nodes will be scheduled. A project start set will be disregarded when scheduling in the first case.

Autoschedule

If this option is activated, the duration of the depending dates will be recalculated automatically each time a link is created or deleted or if an activity is modified.

4.10 The "Administrate Update Behaviors" Dialog Box



Click on the corresponding button on the **Objects** property page to open this dialog. Here you can create, copy, delete and shift individual update behaviors.

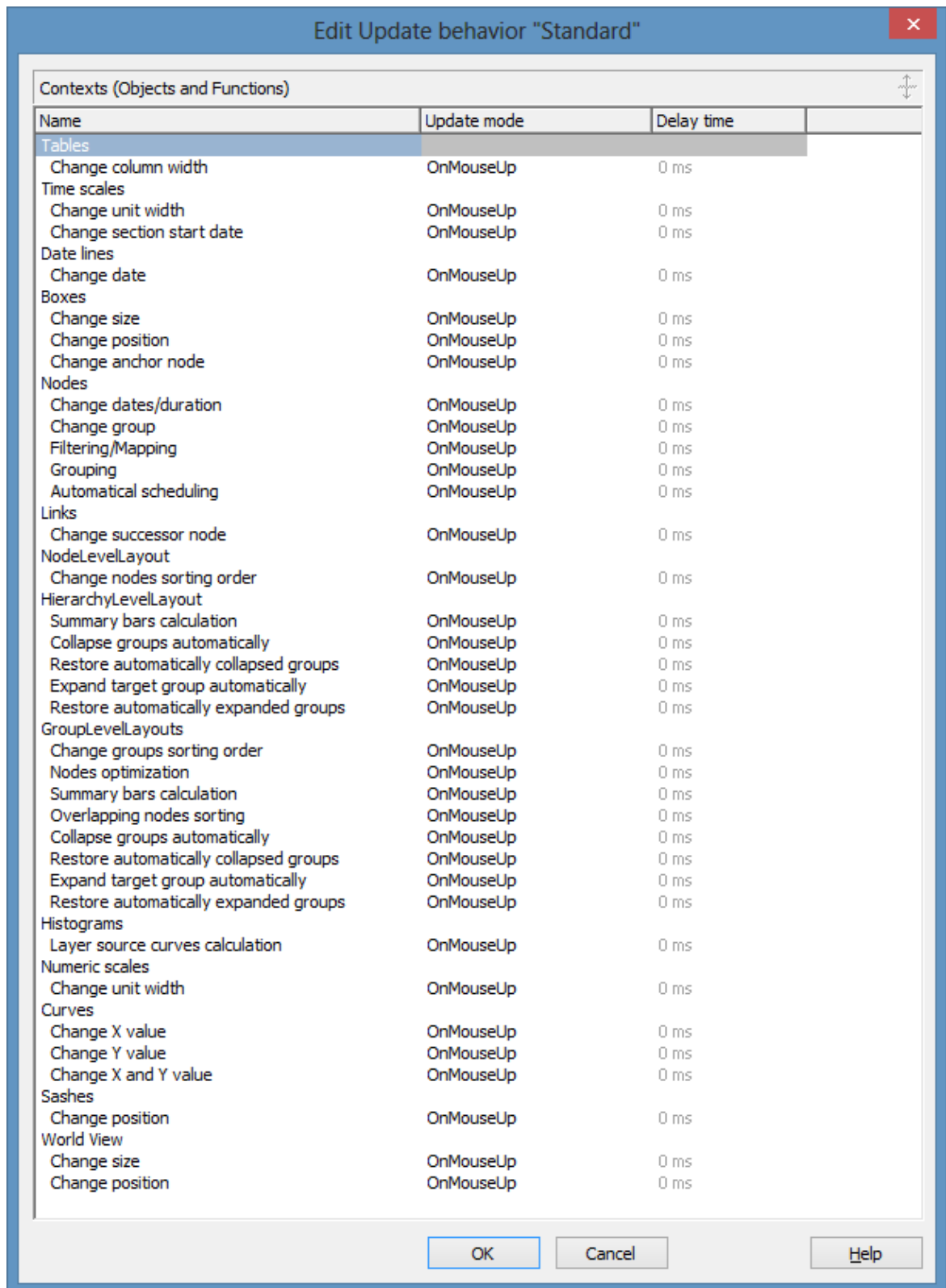
Update behavior

- **Selected:** A red triangle indicates the currently selected update behavior
- **Name:** Lists the names of all existing update behaviors. The names of the individually created ones can be edited.
- **Status:** In the **Status** column each update behavior that has been added (🌟) and/or modified (🔴) since the dialog box was opened is marked by a symbol.

Add / copy / delete / edit / promote / demote update behavior

🌟 📄 ✖ ⬆ ⬇ These buttons let you create, copy or delete update behaviors or move them by one position up or down in the list, respectively.

4.11 The "Edit Update Behaviors" Dialog Box



This dialog can be reached from the <!Administrate Update Behaviors dialog and allows to switch update modes or to modify

Name

Lists the names of all tables and relating functions that are affected by the live update. The names can **not** be edited.

Update mode

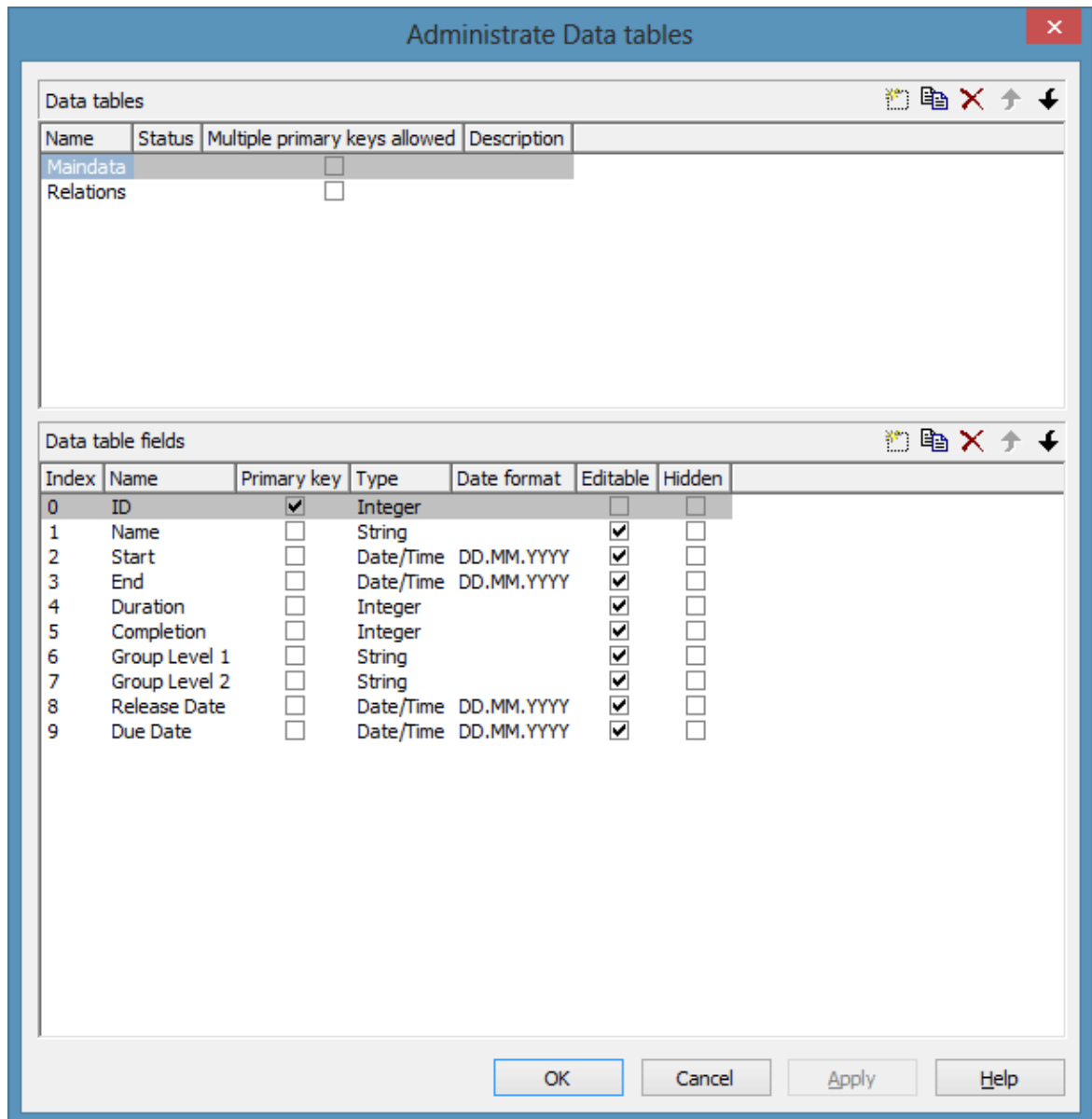
Here you can select a cursor action on which the live update is to take place. This is only possible if you are editing an individually created update behavior created.

Delay time

Here you can set the delay time after which the modified objects of the live update visually are to appear while the mouse cursor is moving.

Setting this property is only possible if the **Update Mode** was set to **OnPauseWhileMouseMoving**



4.12 The "Administrate Data Tables" Dialog Box



You can reach this dialog via the property page **Objects**. Here you can create and edit data tables and their data fields.

Data tables

- **Name:** Lists the names of all existing data tables. The names can be edited.

- **Status:** In the **Status** column each data table that has been added () and/or modified () since the dialog box was opened is marked by a symbol.
- **Multiple primary keys allowed:** Here you can define whether the primary key for your table consists of **one** or **more (maximum 3)** fields. As soon as you have checked the box **Multiple primary keys allowed** you can select up to three data fields for the primary key in the **Data table fields** section. The box **Multiple primary keys allowed** can only be unchecked if no more than one field is selected as primary key in the **Data table fields** section.
- **Description:** Here you can describe the data table.

Add / copy / delete / edit / promote / demote data table



By these buttons you can create, copy or delete data tables or move them by one position up or down in the list, respectively.

Data Table Fields

Here you can create and edit data table fields for the selected data table.

- **Index:** The index of the data fields cannot be modified, since internally, it serves as a reference. In the API, data fields are referred to by the index.
- **Name:** This column displays the names of the fields of the data table. You can modify the field names after clicking on them.
- **Primary Key:** This check box allows to select a data field from the column to be the primary key of the data record.
- **Type:** This field allows to set the data type of the data field selected. You can choose between:

String

Integer

Date/Time

Double

- **Date format:** If the type **Date/Time** has been selected, you can specify the date format for the corresponding data field here. Choose a predefined date format or define your own date format (for example DD.MMM.YY hh:mm). You can compose the format of the following strings:

YY or **YYYY** (two-digit or four-digit figure for the year), **MM** or **MMM** (two-digit figure or three-digit character string for the month), **DD** (two-digit figure for the day), **hh** (two-digit figure for the hour), **mm** (two-digit figure for the minute), **ss** (two-digit figure for the second).

Please note that the date format set here needs to be the same as defined for your node dates.

The date format set here only is relevant for entering data, but not for displaying data.

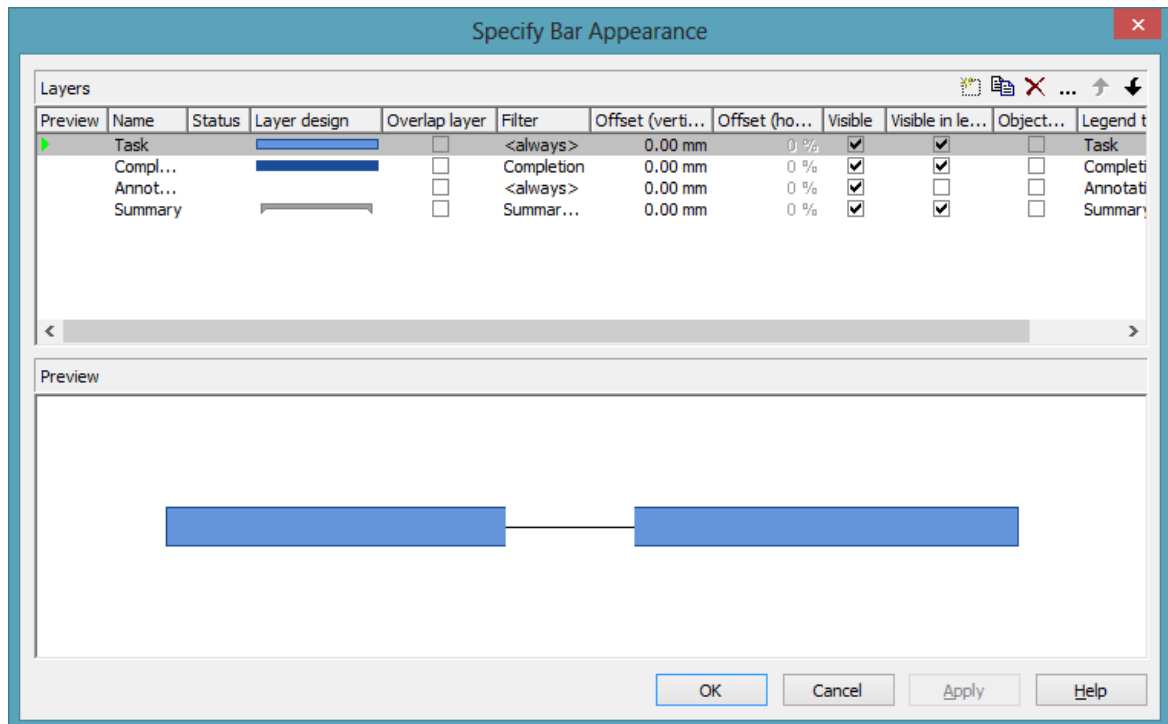
- **Editable:** Please activate this check box for all data table fields that shall be editable in the dialog **Edit Data**.
- **Hidden:** Please activate this check box for all data table fields that shall be hidden in the dialog **Edit Data**.
- **Relationship:** This field allows to define a relationship to another table. The data records of this table will be related to the data records of the other table by the field defined as the primary key. This is why only those tables are offered for selection for which a primary key was defined.

Add / copy / delete / edit / promote / demote data table field



By these buttons you can create, copy or delete data table fields or move them by one position up or down in the list, respectively.

4.13 The "Specify Bar Appearance" Dialog Box



Activities are represented by bars. The graphical representation of a bar is defined by a bar appearance. The graphical representation is composed by layers that are dynamically assigned to activities by filters. A layer is the graphical representation of a single date (symbol layers) or a pair of dates (rectangle layers or line layers). The dates are provided by data fields that are specified in the **Edit Layer** dialog box.

Layers are composed by graphical attributes (shape, line color, pattern, etc.) and an annotation. In addition, they can be of different heights and offset to ensure that all layers assigned to an activity are visible.

If a bar is represented by more than one layer, the layers are drawn consecutively, allowing to overlap. The layer at the top of the **Layer** list is drawn first; the layer at the bottom of the **Layer** list is drawn last and may overlap the ones previously drawn. The final bar appearance results from the graphical display of all layers, the filters of which allow the activity to be displayed.

Layer

Define one layer per line in the table.

Preview



The layers marked by a small arrowhead in the **Preview** column are displayed in the preview window.

The green arrowhead marks the layer on which the cursor is currently positioned. The layer is temporarily displayed in the preview window, that is, as long as the cursor is on the layer. By clicking on the arrowhead you can make it turn red, and vice versa. A red arrowhead indicates that a layer is displayed permanently in the preview window.

Name

Lists the names of all layers that were defined. The names can be edited.

Status

In this column all layers added () and/or modified () after the dialog box was opened are marked by a symbol.

Layer design

This list contains appearances of layers. To modify the appearance of a layer, click on the **Edit layer** button above the list or double-click on the **Layer design** entry to get to the **Edit Layer** dialog box where you can define the graphical attributes and the annotation of the layer.

Overlap layer

In the mode **All nodes in one row** and not **optimized**, you can display overlapping layers by an overlap layer. There is only one appearance to all overlap layers. No filter can be used for it.

Filter

The filter associated to a layer controls the activities that are displayed by the layer. To assign a filter to a layer, mark the **Filter** field. Two buttons will appear:



Open the select box that lists the available filters and select one.



Alternatively, click on the **Edit** button in the **Filter** field to get to the **Administrative Filters** dialog box where you can edit, copy, define or delete a filter.

Examples of filters: "Standard", "Critical", "Milestone". The chosen filter stipulates the condition that an activity must fulfil in order to apply the layer. For example, if you choose the "Critical" filter for the "Early" layer, the "Early" layer will only be displayed in critical activities.

Offset (vertical)

The vertical offset from the central horizontal line of a bar is to be specified in millimetres. Positive values cause the layer to be shifted upwards, negative values will shift the layer downwards.

If you mark the **Offset (vertical)** field of a layer, two buttons will appear with an arrow pointing upwards and downwards to increase or decrease the vertical offset of the selected layer, respectively.



By the second button you can get to the **Configure Mapping** dialog box. Here you can set vertical offsets in dependence of data.



If a vertical offset was mapped, there will be a bold display of the arrow on the button.

Offset (horizontal)

(only for symbol layers) When you mark the **Offset (horizontal)** field of a symbol layer, two buttons appear with an arrow pointing upwards and downwards respectively. You can use the arrows to increase or decrease the horizontal offset against the layer date (-50 to +50 %).

Visible

Uncheck this box if you want the layer to be invisible. You can use this feature to hide a layer without deleting it.

Visible in legend

Check this box if you want the layer to be displayed in the legend.

ObjectDraw events

Tick this box to enable the events **VcObjectDrawing** and **VcObjectDrawn** for nodes which are displayed by this layer.

Legend text

Define a legend text for the layer.

Add layer



A new layer is created.

Copy layer



A copy of the selected layer under a new name is created.

Delete layer



Deletes the selected layer.

Edit layer



You will reach the **Edit Layer** dialog box.

Promote/Demote layer

If a node comprises more than one layer, the layers are stacked on top of one another. The top layer in the table is drawn first. The lower the position of a layer in the table, the more layers it overlaps, i.e. the order of the layers in the table is the order in which they will be drawn in the diagram.



The selected layer will be moved up one position in the table and one position towards the background in the diagram. The layer at the top of the table is overlapped by all other layers.

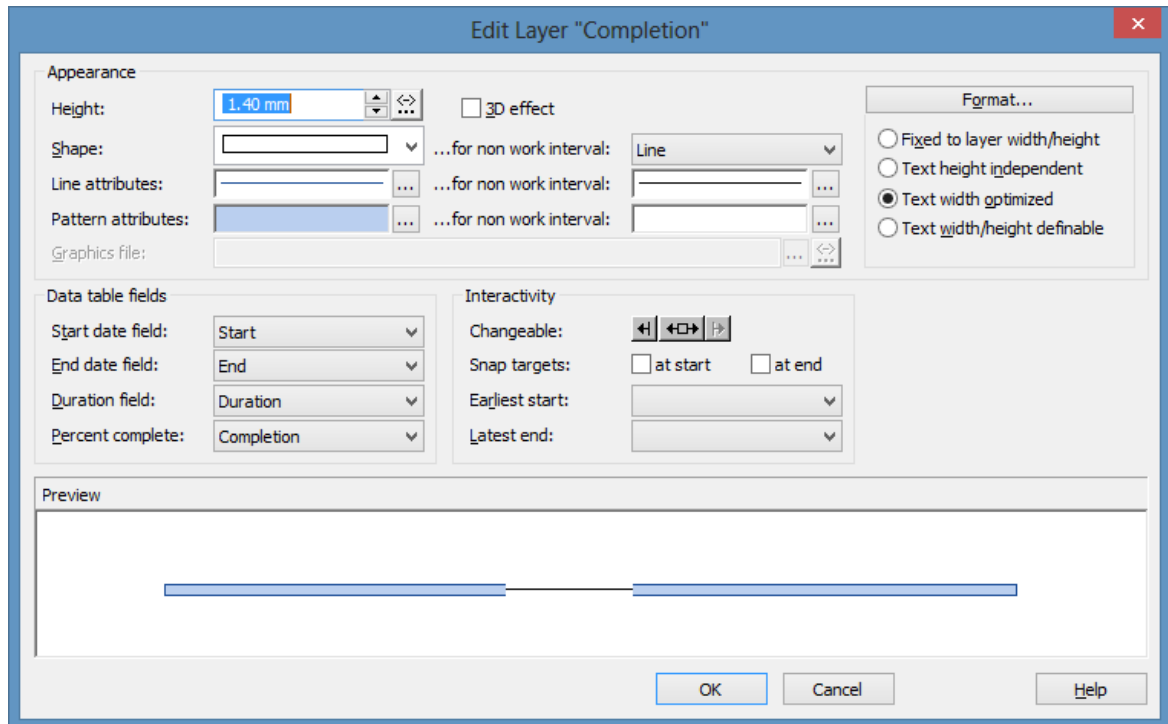


The selected layer will be moved down one position in the table and one position towards the front in the diagram. The layer at the bottom of the table overlaps all other layers.

Preview window

The preview window displays the layers that are marked in the **preview** column, including their overlaps caused by the drawing priority and by offsets.

4.14 The "Edit Layer" Dialog Box



This dialog box you can get to via the **Specify Bar Appearance** dialog box. The name of the layer edited is displayed in the headline.

Height

Here you can define the height of the layer in millimetres either by directly entering the desired value into the field or by clicking on either of the two arrows pointing upwards and downwards.



By clicking on this button you reach the **Configure Mapping** dialog box. It allows to assign heights to layers data-dependent.



If a mapping has been configured, the arrow on the button will appear solid.

3D-Effect

Decide whether or not the layer should be given a 3-dimensional perspective.

Shape

Select from the list a shape for the layer. You can choose between:

- **Bitmap layer:** you can browse for a bitmap file in the **Graphics file** field.)
- **Invisible symbol:** only the layer annotation will be visible. The layer also will not be displayed in the legend.
- **Rectangle layer**
- **Wedge-shaped layer:** wedge ascending or descending
- **Line layer**
- Various types of **symbol layers**.

Rectangle, wedge-shaped and line layers are used to show timespans. Wedge-shaped layers are useful for visualising increasing and decreasing activities, e. g. during the project start or end. Symbol layers are used to show specific points in time.

Non work interval shape

Select the form to be displayed for the non work intervals of rectangle layers. Before, the **Layers with NonWork interval** option on the **Nodes** property page has to be ticked.

The drop down list offers the forms <rectangle>, <line>, <empty area> and <no>, <no> having the effect of showing a continuous layer. Together with the above mentioned option, one can chose for certain layers to show non work intervals and for others not.

Line attributes

The line type of the layer frame is displayed here. To change it, click on the **Edit** button (⋮). Then the **Line Attributes** dialog box will open.


Line attributes for non work intervals

Specify the lines for non work interval layer. Click on ⋮ to open the **Edit line attributes** dialog.

Pattern attributes

Here you can see the currently set layer pattern. Click on ⋮ to open the **Edit pattern attributes** dialog where you can specify pattern, pattern color or background color.

Pattern attributes for non work interval



Specify pattern and fill color for non work interval layers. Click on  to open the **Edit pattern attributes** dialog.

Graphics file

*(only activated, when for **Shape** the option <Bitmap layer> has been specified)* Select a graphics file to visualize the layer.

Relative path names can also be set. If a relative file name was specified, at run time the first folder to be searched will be the one in the path set by the VARCHART property **FilePath**. If it is not found searching will continue in the current directory of the application and in the installation directory of the VARCHART Control.

 Click on this button to open the **Select Graphics File** dialog box.

 By this button you can get to the **Configure Mapping** dialog box where you can configure a mapping for the graphics file. If a mapping was configured, the arrow on the button will be displayed in bold ().

The color of the pixel in the left upper corner of the graphics will be replaced by the diagram color, i. e. this color will appear transparent.

Fixed to layer width/height

If you select this option, the height and width of the layer annotation will be fixed to the height and width of the layer.

Text height independent

If you select this option, the height of an annotation outside the layer will be independent of the layer height, whereas its width will depend on the layer width. The height of annotation inside the layer always is restricted by the layer height.

Text width optimized

If you select this option, the width of an annotation outside the layer will be independent of the layer width, whereas its height will depend on the layer height. The width of annotation inside the layer always is restricted by the layer width.

Text width/height definable

If you select this option, the annotation width and height will be independent of the layer width or height respectively. Then you can specify for each field the width and the number of lines individually in the **Edit Layer Format** dialog or by the properties **MinimumWidth** and **TextLineCount** in objects of the type **VcLayerFormatField**.

Start date field

Specify the start date of the selected layer, e.g. Early Start, Late Start, Scheduled Start.

Format

Opens the **Edit Layer Format** dialog.

End date field

In the end field line, specify the end date of the selected layer, e.g. Early Finish, Late Finish, Scheduled Finish.

To define a rectangle or line layer you need to specify a start and end field or a duration. If both an end field and a duration are specified, the duration entry overrides the end field entry. When an interaction occurs, not only the duration field will be updated, but also the end field.

Duration field

The unit of the duration will be interpreted in dependency on the time unit specified on the **General** property page. From the list, select the data field that contains the duration of the selected layer.

Percent complete

(not activated for symbol and bitmap layers) If you want the current layer to display the percentage degree of completion of an activity, select the data field that contains the percentage degree of completion of the selected layer.

The end date visualized by the layer is calculated from the start date field, the end date field or the duration respectively and the percent complete value. The data of the activity will not be changed.

Changeable

These options allow to set whether the user can move by the mouse a layer completely, the start of a layer and/or the end of a layer.

You can enable/disable three options to the user:

1. The layer start can be moved.
2. The whole layer (i.e. the start and end of the layer together) can be moved.
3. The layer end can be moved.

A button appearing pressed indicates that the options is enabled.

Snap targets

Specify whether the layer defines its start and/or end date as snap target.

Earliest start

Date and time of the selected field are considered the lower limit for the start time of the layer when interactively moving the layer or the node.

This feature can also be set by the property **VcLayer.MinimumStartData-FieldIndex**.

Latest end

Date and time of the selected field are considered the upper limit for the end time of the layer when interactively moving the layer or the node.

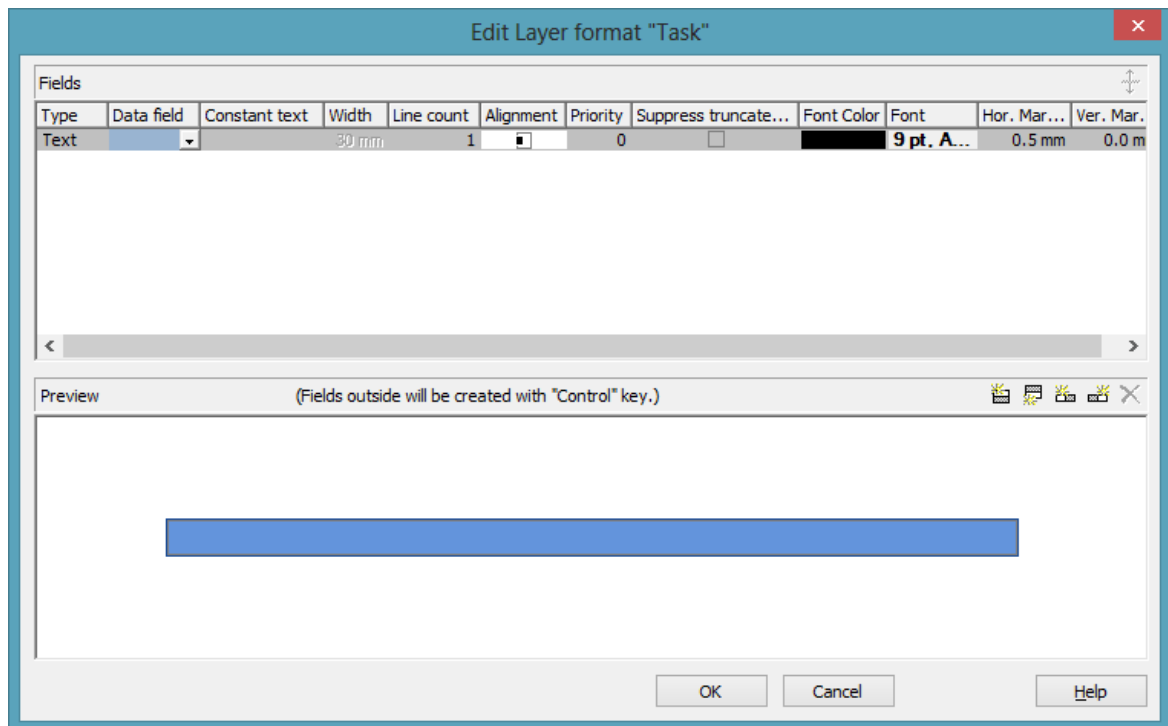
This feature can also be set by the property **VcLayer.MaximumEndData-FieldIndex**.

Preview

In the preview window the layer is displayed with its current settings.

In the preview, bar layers always will be interrupted by a solid line. This line shows how the layer will be displayed at run time, if workfree intervals are highlighted and if a calendar is assigned to the nodes. (These settings are made on the **Nodes** property page. Please note that they do not influence how the layer is displayed in the preview window of the **Edit Layer** dialog.)

4.15 The "Edit Layer Format" Dialog Box



You can get to this dialog box via the **Format** button of the **Edit Layer** dialog box.

Type

The field type (text) is displayed here.

Data field

Select the data field the content of which is to be displayed in the current field. In addition to the data fields defined in the data definition table, you can select the option <Row number> to display the number of the row which contains the layer.

If the content of a data field does not fit into the current field, the excess characters will be clipped in the diagram.

Constant text

(only if no data field was specified) Type a constant text to be displayed in the current field.

Width

Specify the width for the selected field (in mm). The maximum width of a field is 90 mm:

Note: Only editable if **Text width/height definable** was selected in the dialog **Edit Layer**.

Line count

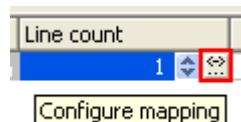
Specify the number of lines of text that can be displayed in the current field.

Note: Only editable if **Text width/height definable** was selected in the dialog **Edit Layer**.

For outside fields of a layer only: You can set the number of text lines dynamically, i.e. in dependence of the length of the text string. For this, two options exist:

1. You can have the number of lines calculated directly, store the results to a field and use them here
2. You can put down the number of lines in a map and assign it here

Case 1: You can have the number of lines calculated by the method **VcLayerFormatField.CalculateLineCount(...)** and store the results to a field. The field can be assigned by the **Configure mapping** dialog, which is to be invoked by pressing the right button that shows a double-headed arrow in the field **Line Count**:



In the dialog popping up, please select a data field from the top selection box and leave the map selection box below empty.

Case 2: For using a map, the map needs to be created and filled before it can be assigned; beside, the map type **vcNumberMap** is to be used. In a map of that type numbers are allocated to character strings. If the character strings put down here are found in a data field (still to be designated), the allocated number of lines will be displayed. Maps can be generated by the property page **Objects** and the button **Maps...**. In the **Configure mapping** dialog you can select a data field and a map, thus designating the data field the content of which is to be compared to the character strings of the map. You can view the content of the selected map in the dialog and modify it in continuative dialogs.

Alignment

Specify the alignment of the content of the selected field (left, centered, right).

Priority

Specify the priority of the layer field. Priority values between -9 and +9 are allowed. If the total width of the layer is too small to show the contents of all layer fields, the priority of the layer field determinates if its content is displayed. The content of the field of highest priority is displayed first, if possible, completely. The contents of fields of lower priorities are displayed subsequently. If a field content cannot be displayed completely, it will be suppressed or truncated (depending on the setting in **Suppress truncated text**).

Suppress truncated text

Specify whether a text that does not fit into the field is to be suppressed or truncated.

Font Color

Indicates the font color for the current field. If you click on the field, two buttons will appear:



By the **arrow** button you can open the color picker to select a font color.



By the second button you can get to the **Configure Mapping** dialog box. It allows to assign colors in dependence of data.



If colors were mapped, the arrow on the button will appear solid.

Font

Indicates the font style of the field. If you click on the field, two buttons will appear:



The Windows **Font** dialog box will appear.



By the second button you can get to the **Configure Mapping** dialog box. It allows to assign fonts in dependence of data.




If fonts were mapped, the arrow on the button will appear solid.

Apply selected property to all fields

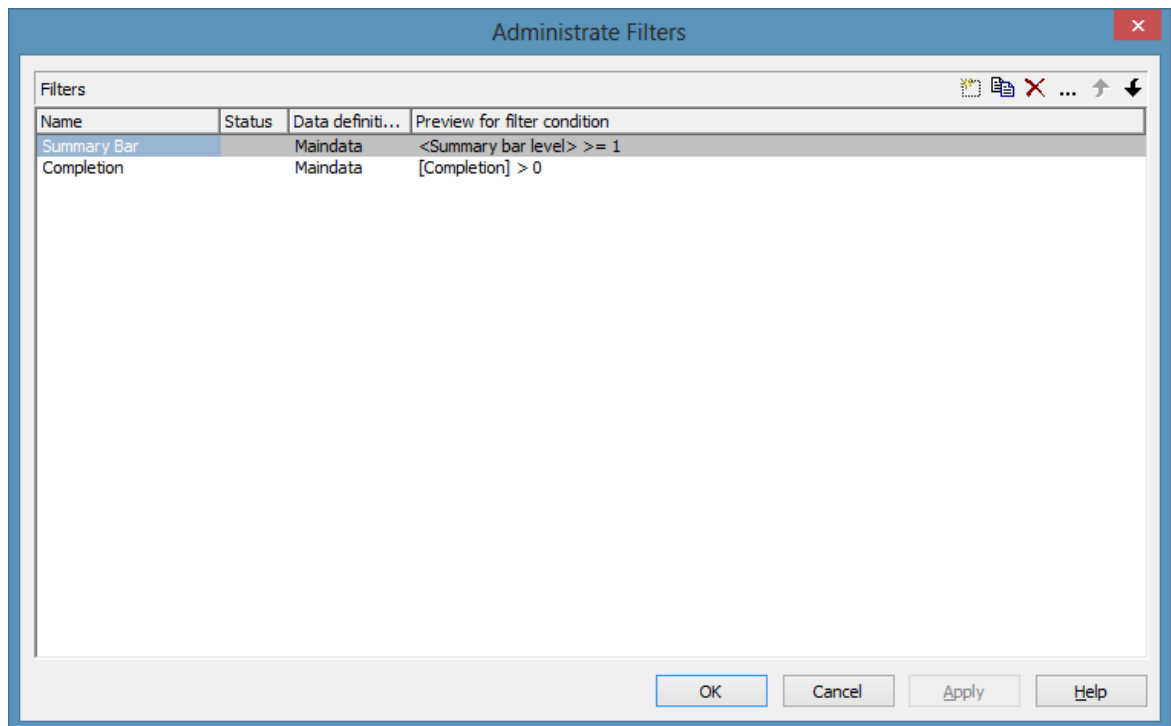
 Applies the marked property to all fields.

Preview

The current fields are displayed in the preview window. If you click on a field, you can modify its attributes in the **Fields** table.

 By the buttons above the preview window you can add new fields or delete the marked field. If you want to add new fields outside of the layer, press the **Ctrl** key in addition. You also can use the **Del** key to delete fields.

4.16 The "Administrate Filters" Dialog Box





You can get to this dialog box

- via the **Objects** property page
- for layers: via the **Specify Bar Appearance** dialog box
- for table formats: via **Edit Table** dialog box
- for links: via the **Filter** button of the **Link** property page
- for histogram curves: via the **Filter** combo box of the **Edit Histogram** dialog
- for nodes: via the **Filter** button of the **Nodes** property page.

Name

Lists the names of all existing filters. The names can be edited.

Status

In the **Status** column each filter that has been added () and/or modified () since the dialog box was opened is marked by a symbol.

Data definition table

This column shows the data definition table (**Maindata** or **Relations**) for each filter and is only shown if the check box **Extended data tables enabled** on the property page **General** is not ticked.

Preview for the filter condition

This column shows the criteria of each filter. The criteria cannot be edited here. To modify the filter criteria, click on the **Edit filter** button.

Add filter



A new filter will be created. You can modify its default name by double-clicking and editing it. New filters are created context-sensitively, i. e. the data definition table always will be specified automatically.

Copy filter



Copies the selected filter.

Delete filter



The marked filter in the list will be deleted. You can only delete filters that are not currently used.

Edit filter



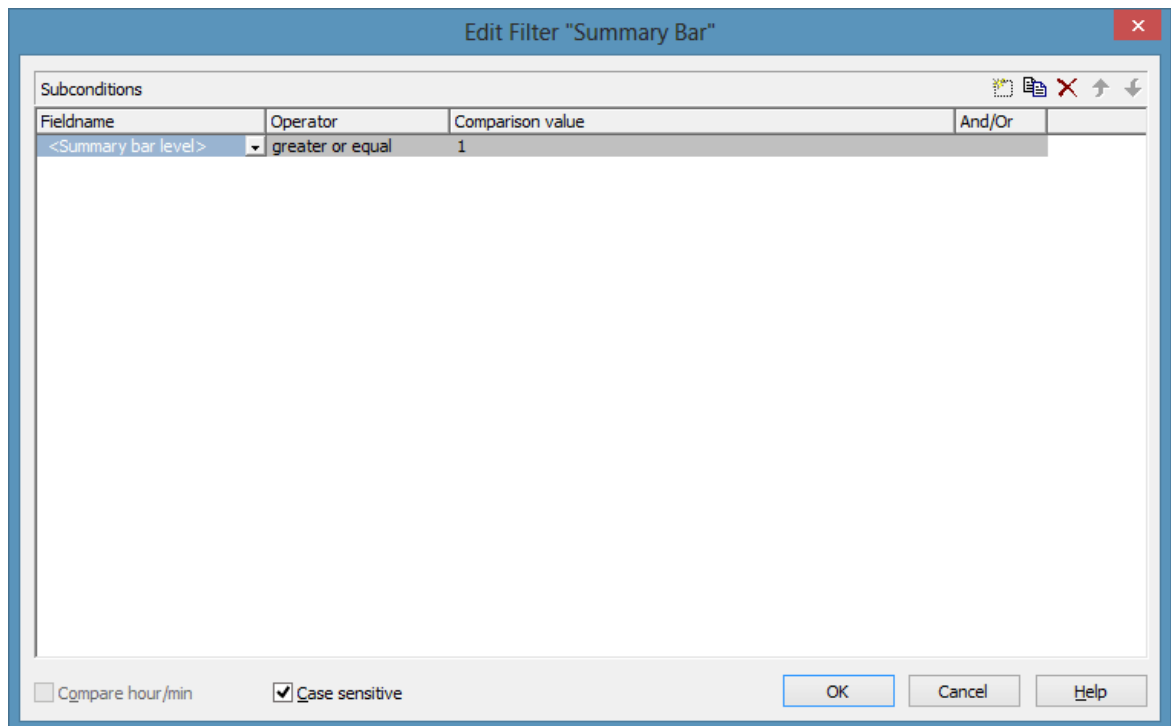
Press the **Edit filter** button to view or modify the criteria of a filter. The **Edit Filter** dialog box will appear where you can edit the criteria of the corresponding filter.

Promote / demote filter



By these buttons you can move the filter by one position up or down in the list.

4.17 The "Edit Filter" Dialog Box



You can get to this dialog box either

- by the **Objects** property page
- or by the **Administrate Node Appearances** dialog box
- or by the **Administrate Link Appearances** dialog box, where you can activate the **Administrate Filters** dialog box and then click on the **Edit filter** button. The head line of this dialog box displays the name of the filter being edited.

Add subcondition

 Inserts a new line for a subcondition above the selected line.


Copy subcondition

 Copies the selected subcondition.

Delete subcondition

 Deletes the selected subcondition.

Evaluate subcondition earlier/later

 If a filter consists of several subconditions, they are evaluated one by one, starting by the top of the list.

You can click on the **Evaluate subcondition earlier/later** button to move a selected subcondition upward or downward by one position in the table to have it worked off earlier or later.

Fieldname

This list contains all data fields available to be compared to the comparison value. It also contains some predefined settings:

- The <summary bar level> entry can be used for displaying summary bars in Gantt diagrams. For example, you can specify a filter containing the condition "<summary bar level> greater or equal 1" and assign it to a layer (e.g. "Summary level 1") in order to display summary bars for level 1. Please note that the option **Summary bars** has to be activated on the **Sorting** property page.
- Filters containing the <grouping level> setting can be used for example in the **Edit Table** dialog (for Gantt diagrams) as row filters for basic rows.
- <Gantt: collapsed>: to collapse groups
- <Gantt: nodes in separate rows>: to display all nodes in separate rows
- <Gantt: nodes overlaid>: to allow for overlapping nodes
- <Gantt: row>: to define filters for single rows
- <Gantt: summary node>: definition of summary bars
- <Node Read Only>: filters that select for nodes that are defined as read only.

This feature can also be set at run time by the VcFilterSubCondition property **DataFieldIndex**.

Operator

The operator compares the value of a data field with a comparison value.

Comparison value

This column shows the current comparison value. The **Comparison value** select box lists all fields (in square brackets) that can be used as comparison values. The type of the data fields offered as comparison values correspond

to the data type of the data field specified in the **Fieldname** column. For example, if the data field "Early Start" is specified in the **Fieldname** column, for the comparison value you can select either a date field (e. g. "Early End") or the <today> option or the <input> option.

With the help of the <input> option you can specify a variable filter. In variable filters only the field name and the operator are specified, but not the comparison value. You can specify the comparison value when necessary. You can use a variable filter when you open a project and want to select the activities to be displayed.

Dates need to be entered in the format defined on the **General** property page. If you have selected a date field in the **Fieldname** field, two arrow buttons will appear as soon as you click on this field. The first arrow button lets you open a combobox with all available date data fields. The other arrow button opens a Date dialog box from which you can select a date by mouse-click. You can also edit the date direct.

Numeric values or texts must be typed manually into the **Comparison value** field.

With the operators "equal" and "unequal" you can use wildcards in text fields:

*: no sign or any number of signs

?: exactly one sign

If you do not want to use the signs * or ? as wildcards, but want to search for these signs, you have to set a backslash in front of them:

*: *

\?: ?

If the backslash does not follow a * or ?, the program searches for the sign \.

Examples:

Activity 1 : Name = "Construction"

Activity 2 : Name = "*Construction"

Possible filters for activity 1:

[Name] = C*

[Name] = C?nstruction

Possible filters for activity 2:

[Name] = *C*

[Name] = **

[Name] = ?C*

And/Or

This column shows the logical connection of two subconditions in the table.

Choose the AND operator to connect the current subcondition and the next subcondition in the table to select only those objects that fulfil both subconditions. Choose the OR operator to select those objects that fulfil at least one of the subconditions.

If you have formulated several subconditions, linking them partly with AND and partly with OR, the AND links will be processed first. (AND links are stronger than OR links).

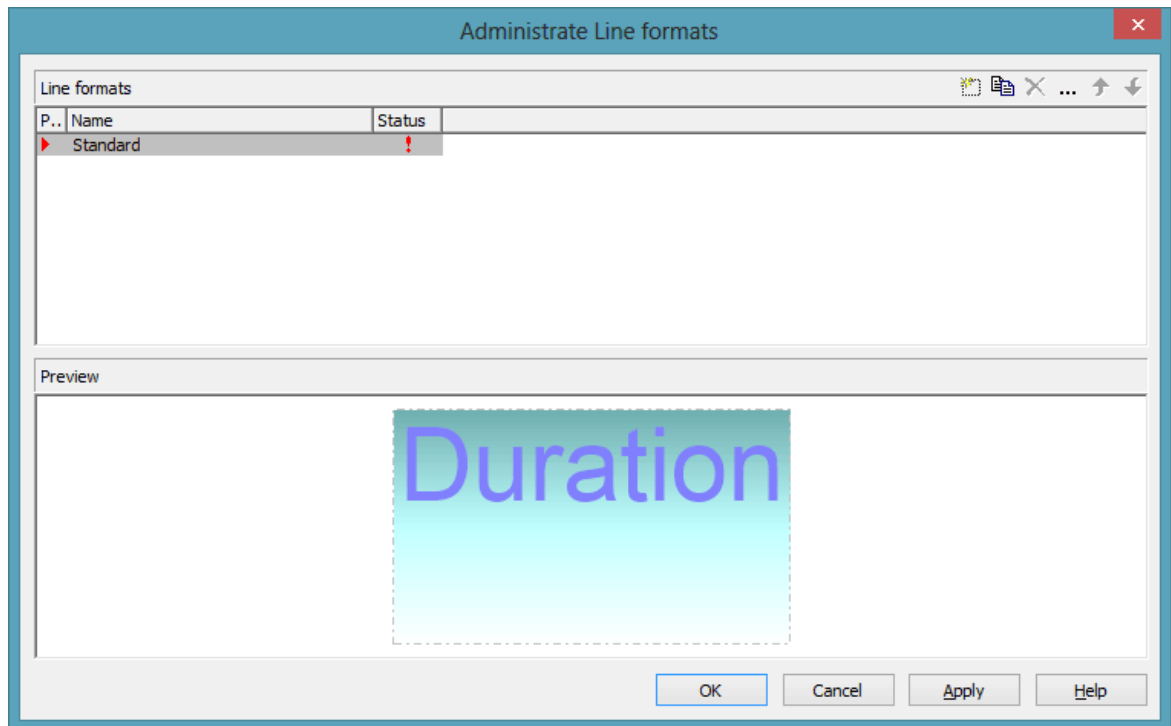
Compare hour/min

Activate this check box if the hours and minutes of a date are to be considered when dates are compared.


Case sensitive

Activate this check box if the comparison of the entries is to be case-sensitive.

4.18 The "Administrate Line formats" Dialog Box



You can get to this dialog box

- by clicking the corresponding button on the **Objects** property page
- by clicking  in the **Line format** field of the **Administrate Line grids** dialog.



Preview

In this column a red triangle marks the line format which is displayed in the preview below.

Name

Lists the names of all existing line formats. The names can be edited.

Status

In the **Status** column each line format that has been added () and/or modified () since the dialog box was opened is marked by a symbol.

Add line format



A new line format will be created. You can modify its default name by double-clicking and editing it.

Copy line format



Copies the selected line format.

Delete line format



The marked filter in the list will be deleted. You can only delete filters that are not currently used.

Edit Line format



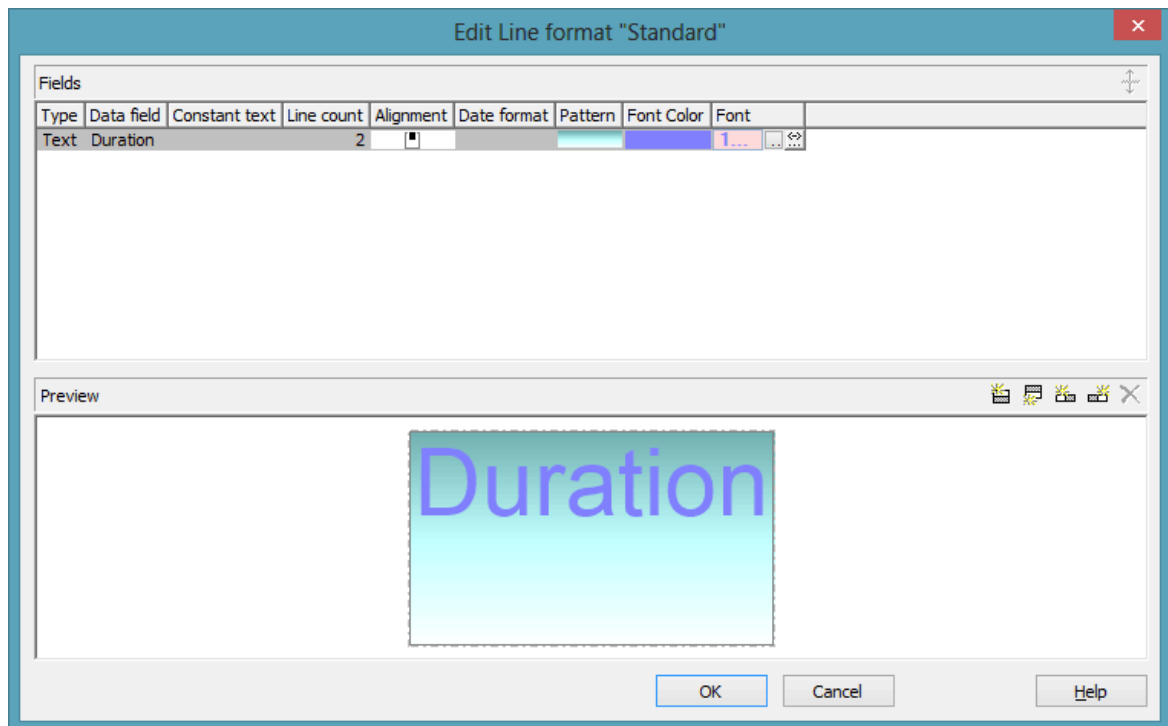
Opens the dialog **Edit Line format** which lets you specify the attributes of the line format such as color, pattern etc.

Promote / demote line format



By these buttons you can move the line format by one position up or down in the list.

4.19 The "Edit Line format" Dialog Box



You can get to this dialog box

- by clicking the button **Line formats** on the **Objects** property page and then the button in the **Administrative line formats** dialog
- by clicking in the **Line format** field of the **Administrative Line grids** dialog

Type

The field type (text) is displayed here.

Data field

Select the data field whose content is to be used as line grid annotation. In addition to the data fields defined in the data definition, you can select the entries `<Date>` or `<Group title>`: The current date or the group title (if grouping is switched on) is displayed.

If the content of a data field does not fit into the current field, the excess will be cropped in the diagram.

Constant Text

(only if no data field has been specified) Type a constant text to be displayed in the current field.

Line Count

Specify the number of lines of text that can be displayed in the current field.

Alignment

Specify the alignment of the content of the selected field (left, centered, right).

Date format


If you have selected <Date> as data field for the annotations, you can specify the date format here. To compose the date you can use the following tokens:


- D: first letter of the day of the week (not adjustable)
- TD: Day of the Week (adjustable by using the event **VcTextEntrySupplying**)
- DD: two-digit figure for the day of the month: 01-31
- DDD: first three letters of the day of the week (not adjustable)
- M: first letter of the name of the month (not adjustable)
- TM: name of the month (adjustable by using the event **VcTextEntrySupplying**)
- MM: two-digit figure for the month: 01-12
- MMM: first three letters of the name of the month (not adjustable)
- YY: two-digit figure for the year
- YYYY: four-digit figure for the year
- WW: two-digit figure for the number of the calendar week: 01-53
- TW: text for "calendar week" (adjustable by using the event **VcTextEntrySupplying**)
- Q: one-digit figure for the quarter: 1-4
- TQ: name of quarter (adjustable by using the event **VcTextEntrySupplying**)
- hh: two-digit figure for the hour in 24 hours format: 00-23



- HH: two-digit figure for the hour in 12 hours format: 01-12
- Th: Text of "o' clock" (adjustable by using the event **VcTextEntrySupplying**)
- TH: "am" or "pm" (adjustable by using the event **VcTextEntrySupplying**)
- mm two-digit figure for the minute: 00-59
- ss: two-digit figure for the second: 00-59
- TS: short date format, as defined in the regional settings of the windows control panel
- TL: long date format, as defined in the regional settings of the windows control panel
- TT: time format, as defined in the regional settings of the windows control panel
- xC/XC: You can set a maximum ten-place, simple upward counting from a reference date onward, for example "15:05:07:16:00", which equals 15 months, 5 days, 7 hours, 16 minutes, 0 seconds. The notation is: **xC44:C33:C22:C11:C00**. In written language: Show at least 2 digits for the counters 4...0 and a preceding "-" symbol if the value is negative. The separators are variable and can be replaced by other separators symbols. "x" means: Display a preceding "-" symbol if the value is negative, but no "+" symbol if it is positive. "X" means: Display a preceding "-" symbol if the value is negative and a "+" symbol for positive values. In the dialog **Edit Time Scale Section...** the check boxes **Use reference date** and **Adjust major ticks to reference date** need to be ticked, also, the parameter **Serial annotation** has to be set to **No**. In the application the reference date is set at run time by the call **VcRibbon.set ReferenceDate**, overriding any settings in the dialog.

Note: Characters which are not to be interpreted as part of the date should be preceded by a backslash '\'. '\\' for instance results in '\'. The special characters: ':, /, -' and **blank** don't need '\' as prefix

Pattern


Here you can select the fill pattern and color for the current field. By clicking on  you open the **Edit pattern attributes** dialog where you can specify a pattern, a background color and, if needed, a second pattern color



by clicking on . You can define your own colors in addition to the ones suggested. Transparent colors are also available.

By clicking on  you open the **Configure Mapping** dialog box. Here you can configure data-dependent patterns and colors. If a mapping has been configured, the arrow on the button will be displayed in bold ().

Font Color

Indicates the font color for the current field. If you click on the field, two buttons will appear:



 by the arrow button you can open the Color picker to select a font color.

 by the second button you reach the **Configure Mapping** dialog box. Here you can configure data-dependent font colors. If a mapping has been configured, the arrow on the button will be displayed in bold ().


Font

Indicates the font style for the current field. If you click on the field, two buttons will appear:

 The Windows **Font** dialog box will appear.


 by the second button you reach the **Configure Mapping** dialog box. Here you can configure data-dependent fonts. If a mapping has been configured, the arrow on the button will be displayed in bold ().

Apply selected property to all fields

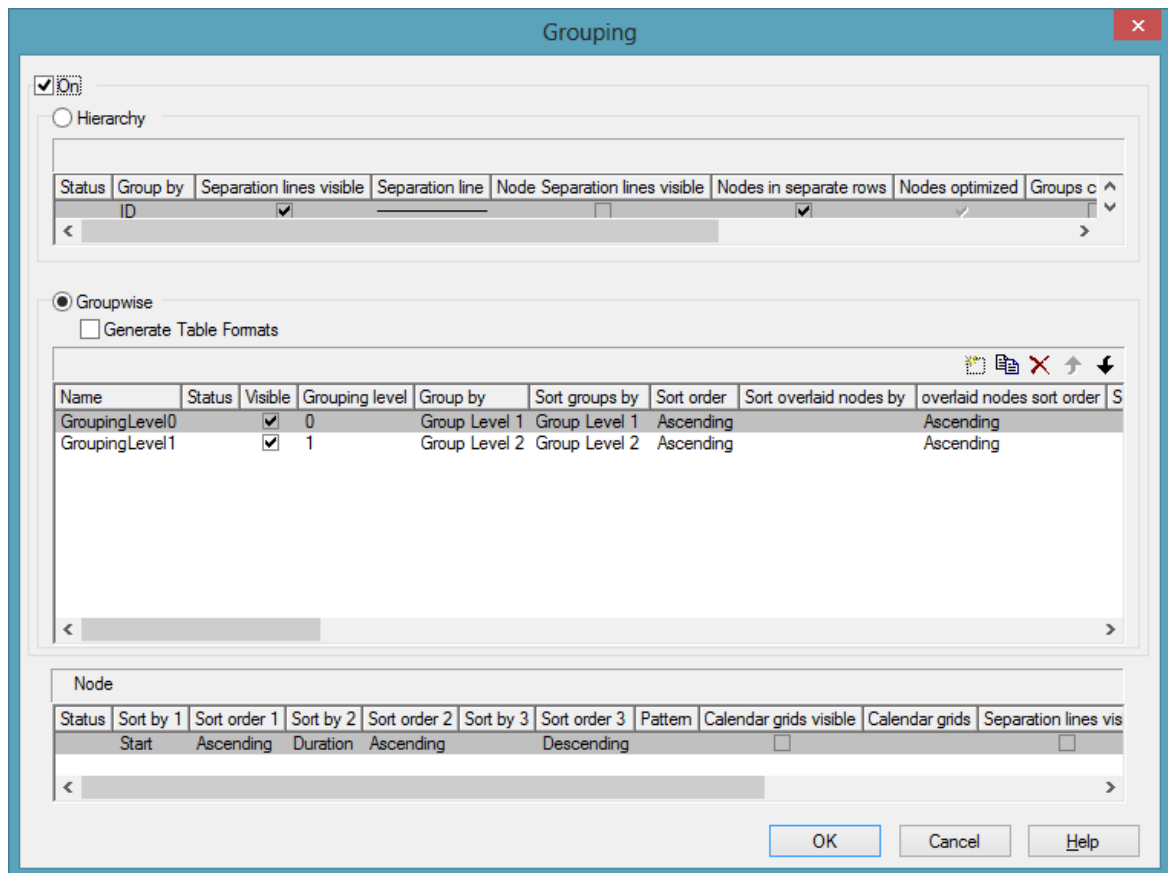
 Applies the marked property to all fields.

Preview

The current fields are displayed in the preview window. If you click on a field, you can modify its attributes in the **Fields** table.

 With the help of the buttons above the preview window you can add new fields or delete the marked field. You also can use the **Del** button to delete fields.

4.20 The "Grouping" Dialog Box



In this dialog you can set options to hierarchical and grouping arrangements of nodes, sorting of nodes and to the layout of these structures.

The dialog shows three different sections: **Hierarchy**, **Groupwise** and **Nodes**, where you can set the corresponding options.

On

The grouping of nodes either in the form of a hierarchy (according to a hierarchy code) or in the form of grouping according to different criteria are switched on or off.

> Hierarchy

If you activate this radio button, the activities will be arranged in a hierarchy, according to a hierarchy code. In the code, hierarchy levels are separated by dots. If you select this option, the section **Groupwise** automatically will become inactive.

In the table below the **Hierarchy** button you can make further settings concerning the hierarchical arrangement.

Group By


Select the data field which contains the code by which the activities are arranged.

Separation lines visible

Tick this box to display separating lines between different hierarchical levels.

This feature can also be set by the property **VcHierarchyLevelLayout.SeparationLinesVisible**.

Separation line

By clicking on  you can open the dialog **Line attributes** and specify the style of the separation lines.

The line attributes can be also set by the corresponding properties **VcHierarchyLevelLayout.SeparationLineColor**, **VcHierarchyLevelLayout.SeparationLineThickness** and **VcHierarchyLevelLayout.SeparationLineType**.

Nodes in headers

Specify whether each node of a group will be displayed in a separate row or not.

If this option is activated, the table section of the activities is suppressed, so you will need to use the layer format or tooltip to identify the activities for the user.

Nodes overlaid

Specify whether the node layout on this hierarchy level is to be optimized or if nodes overlap.

Bodies collapsed

If you select this option, all hierarchy levels from the second one downward will be displayed collapsed on the start of the program. They can be expanded interactively after the start.

Summary Bar

If you tick this box, summary bars will be displayed in all levels. If you want to display summary bars only for special levels, you have to define a layer with an appropriate filter condition (<Summary bar level> = ...).

This feature can also be set by the property **VcHierarchyLevelLayout.-SummaryBarsVisible**.

Collapse groups automatically

If you tick this box, every group save the one just being touched will be collapsed when a node or a group is being moved interactively.

Restore automatically collapsed groups

When this check box is ticked every group that was automatically collapsed before is restored again when a node/ a group is being moved interactively.

Expand target group automatically

When this check box is ticked the target group is expanded automatically when a node/a group is being moved interactively.

Restore automatically expanded group

When this check box is ticked every group that was automatically expanded before is restored again when a node/a group is being moved interactively.

Pagebreak after Group

After clicking on , the following options can be selected:

- **None:** no page break will be inserted
- **On page full:** if a group would be separated by a page break, the page break will be inserted after the preceeding group already
- **After each group:** a page break is inserted after each group

This features can also be set by the property **VcHierarchyLevelLayout.-PageBreakMode**.

Maximal level for pagebreaks







Here you can specify up to which hierarchy level page breaks after each group are to be carried out. If the level is set to 4, for example, no page break will be carried out after level 4.

If the level is set to the default -1, page breaks are carried out on each level.

This feature can also be set by the property **VcHierarchyLevelLayout.-LevelMaximumForPagebreaks**.

> Groupwise

If you activate this radio button, the activities will be arranged in groups (grouped by different criteria) and the section **Hierarchy** automatically will become inactive.

In the area below the <bGroupwise button you can set all further grouping options - mostly concerning the layout (pattern, calendar grid, line grid etc.). You can define different settings for each grouping level. By clicking on the corresponding buttons       levels can be created, deleted, copied or the order of the levels can be changed.

Generate Table Formats

If this check box is activated, for each grouping level an own table format will be created: Subtitle_n, Collapsed_n. The formats probably have to be adapted by the dialog **Edit table format**, especially the data field.

If this check box is not activated, no table formats will be created for new grouping levels. You may have to create them yourself, if required. This option is helpful, because it allows to get along with only two table formats for grouping (Subtitle and Collapsed) that you can modify by maps and filters.

Group node visible

Tick this box to display bars in the diagram for those groups coming from a separate group data table. For that purpose you also have to tick the **Extended data tables enabled** option on the **General** property page before.

Name

Specify a name for the corresponding grouping level.

Visible

Specify whether or not the groups of this level are to be displayed.

Grouping level

The level, for which the settings of this line are valid is displayed here. You can change the order of the levels by clicking on the corresponding arrow buttons above the table.

Group by

Select the data field by which the activities on the current grouping level are to be grouped. If you leave this field blank, the activities on the current grouping level will not be grouped.

Groups sorted by

Select the data field by which the groups should be sorted when the program is started. If you do not set anything here, the sequence of the nodes will derive from the sequence of loading.

Sort order

Set the sorting order (ascending or descending) on the current grouping level.

Sort overlapping nodes by

Select the data field by which the nodes of a group that are put in a single row are to be sorted. If you do not set anything here, the sequence of the nodes will derive from the start date and the duration of the activities, i.e. the earliest and the shortest activities will be farthest in front. This property can only apply if the property **VcGroupLevelLayout.NodesArranged-Optimized** was set to **False**.

Overlapping nodes sort order

Set the sorting order (ascending or descending) of the overlapping nodes.

Sort optimized nodes by



Select the data field by which the nodes of a group that are put in a single row are to be sorted. If you do not set anything here, the sequence of the

nodes will derive from the start date and the duration of the activities, i.e. the earliest and the shortest activities will be farthest in front. This property can only apply if the property **VcGroupLevelLayout.NodesArranged-Optimized** was set to **True**.

Optimized nodes sort order

Set the sorting order (ascending or descending) of the optimized nodes.

Pattern

If you click on  you open the dialog **Pattern attributes**. Here you can specify the background pattern and two pattern colors of the group title row as well as by clicking on  assign the respective property in dependence on data.

Calendar

Select the data field that contains the name of a calendar, which should be used for the group node.



Calendar grid visible

Specify whether a calendar grid is displayed.

Calendar grid with subgroups

Specify, whether the calendar grid shall be displayed for subgroups as well.

Calendar grids

By clicking on  you can select a calendar grid for the group or create a new one in the **Administrate Calendar grids** dialog which you can open by clicking on . For further information about calendar grids see the chapter **The Administrate Calendar grids** dialog.

If you select <From Scale> the first not visible calendar grid from the time scale will be displayed.



Line grid visible

Specify whether a line grid is displayed.

Line grid with subgroups

Specify, whether the line grid shall be displayed for subgroups as well.

Line grids

By clicking on  you can select a line grid for the grouping level or create a new one in the **Administrate Line grids** dialog which you can open by clicking on . For further information about line grids see chapter **The Administrate Line grids** dialog.



Date lines visible

Specify whether a line grid is displayed.

Date lines with subgroups

Specify, whether the date line shall be displayed for subgroups as well.

Date lines

By clicking on  you can select a date line for the grouping level or create a new one in the **Specify Date lines** dialog which you can open by clicking on . For further information about line grids see chapter **The Specify Date lines** dialog.

Separation lines visible

Tick this box to display separating lines between different groups.

Separation lines at top

If you tick this box, the separation line will be drawn above a group (instead of below).

Separation line

You can edit the appearance of the separating lines after clicking on the **Edit** button.

Nodes in headers

Specify whether each node of a group will be displayed in a separate row or not.

If this option is activated, the table section of the activities is suppressed, so you will need to use the layer format or tooltip to identify the activities for the user.

Nodes overlaid

Specify whether the node layout on this group level is to be optimized or if nodes overlap.

Bodies collapsed

If you select this option, the groups will be displayed initially collapsed, i. e. only the group titles will be visible, but not the nodes.

Modifications allowed

If you tick this box, the user can collapse expanded groups and vice versa. The user can collapse/expand groups by double-clicking on the group heading in the table section, by clicking once on the minus or plus symbol next to the group heading or by the context menu of a group.

Summary Bar

If you tick this box, summary bars will be displayed. To specify summary bars for a specific level, you have to define a layer with an appropriate filter condition (<Sum bar level = ...).

Moving groups vertically via tables

When this check box is ticked you can change the order of groups by drag interactions in the table area.

Collapse groups automatically

If you tick this box, every group save the one just being touched will be collapsed when a node is being moved interactively.

Restore automatically collapsed groups

When this check box is ticked every group that was automatically collapsed before is restored again when a node is being moved interactively.

Expand target group automatically

When this check box is ticked the target group is expanded automatically when a node is being moved interactively.

Restore automatically expanded group

When this check box is ticked every group that was automatically expanded before is restored again when a node is being moved interactively.

Moving groups vertically via diagram

When this check box is ticked you can change the order of groups by drag interactions in the diagram area.

Pagebreak after Group

After clicking on , the following options can be selected:

- **None:** no page break will be inserted
- **On page full:** if a group would be separated by a page break, the page break will be inserted after the preceeding group already
- **After each group:** a page break is inserted after each group

This features can also be set by the property **VcGroupLevelLayout.Page-BreakMode**.

> Nodes

The below settings describe the options that you can select for grouped or ungrouped nodes concerning in particular sorting options as well as the layout of the node rows.



Note: Please note that the settings for the sorting of the activites are only valid when opening the diagram. If you want to sort the activities again later, please use the VcGantt method **SortNodes**.

Sort by 1 to 3

Specify the data fields by which the activities are to be sorted when the diagram is opened. You can sort the activities by up to three data fields, in ascending or descending order respectively (**Sort Order 1 to 3**).

If you specified a data field by which the activities are to be grouped (**Grouping by**), each group will be sorted separately.

Pattern

If you click on  you open the dialog **Pattern attributes**. Here you can specify the background pattern and two pattern colors of the node line as well as by clicking on  assign the respective property in dependence on data.

Calendar grid visible

Specify whether a calendar grid is displayed.

Separation lines visible

Specify whether a separation line is displayed.

Separation lines at top

If you tick this box, a separation line will be drawn above a node (instead of below).

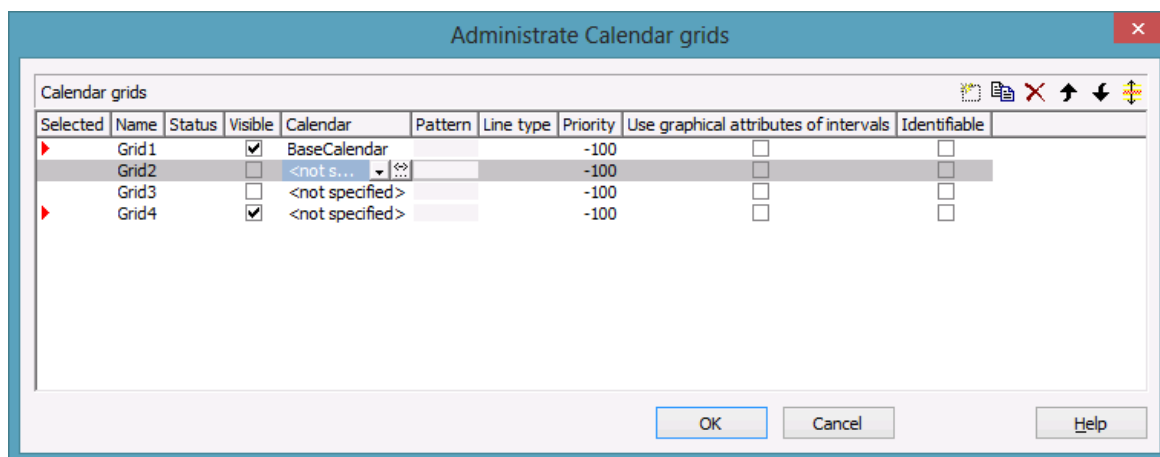
Separation line


The layout of the separation lines can be edited in the **Line attributes** dialog box which appears when you click on the **Edit** button.


Separation lines step size




Specify after how many activities a separating line is drawn.

4.21 The "Administrate Calendar grids" Dialog Box



You can get to this dialog by clicking on  in the field **Calendar grids** in the dialog **Grouping**, section **Groupwise**.

By clicking on the corresponding buttons  you can add, copy or delete calendar grids.

The   arrow buttons allow to move a calendar grid by one line down or up, while the  button lets you assign the feature just activated to all calendar grids listed.

The below features can be set to calendar grids:



Selected

By clicking on this field you can select this calendar grid to apply to the grouping level. A red arrow indicates that this calendar grid was selected.

Name

Enter a name for the calendar grid.

Status

In this column each calendar grid that was added () and/or modified () after opening the dialog box is marked by a symbol.


Visible

Activate this check box for the calendar grids to be displayed.


Calendar

The calendar selected here will apply to all groups of this level. If no calendar is selected here, the calendar of the level to which the calendar grid was assigned will apply.

Pattern

Select the fill pattern and color for the calendar grid. By clicking on  you open the **Edit pattern attributes** dialog where you can specify a pattern, a background color and, if needed, a second pattern color. You can define your own colors in addition to the ones suggested. Transparent colors are also available.

Line type

When clicking on this button () , the **Line attributes of calendar grid** dialog box will appear, where you can enter the settings of the border lines of the calendar grid.

Priority

Lets you set the priority of a calendar grid. It refers to other calendar grids and to layers (> 0: in front of the layers, < 0: behind the layers).

Use graphical attributes of intervals

Specify whether the graphical attributes that have been set for the intervals are to be displayed.

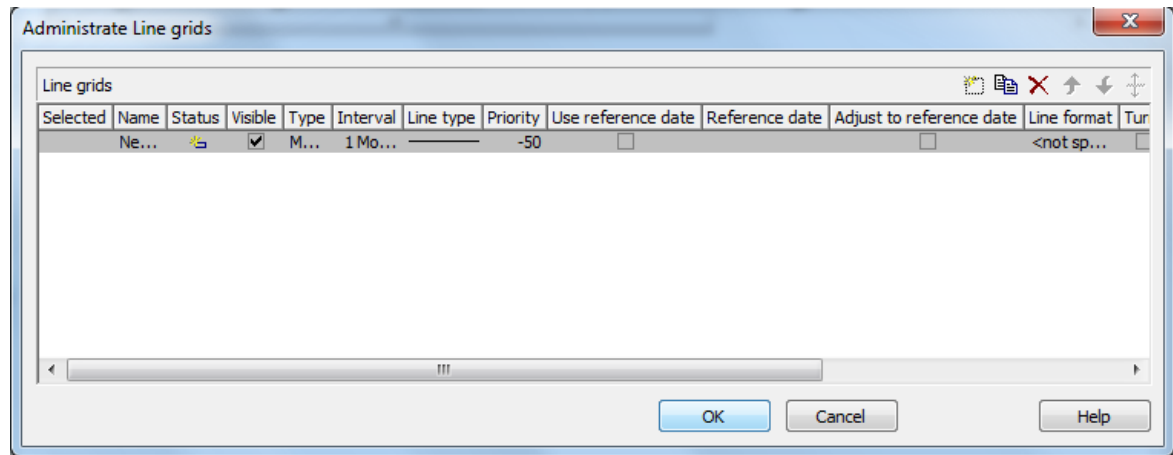
Identifiable


This option allows to set whether or not the calendar grid can be identified by the VcGantt method **IdentifyObjectAt**. A tool tip text for instance can only appear if a calendar grid can be identified; the same is valid for the context menu popping up on right-clicking the mouse. For a tool tip text to appear, the corresponding interval also has to be identifiable; please see the **Calendar grid** section in the **Edit time scale section** dialog.


Snap targets start/end




Tick this check box to have the calendar grid's relevant positions defined as "snap targets" for nodes/layers to be moved.

4.22 The "Administrative Line grids" Dialog Box



You can get to this dialog by clicking on  in the field **Line grids** in the dialog **Grouping**, section **Groupwise**.

By clicking on the corresponding buttons  you can add, copy or delete line grids.

The   arrow buttons allow to move a line grid by one line down or up, while the  button lets you assign the feature just activated to all line grids listed.

The below features can be set to line grids:



Selected

By clicking on this field you can select this line grid to apply to the grouping level. A red arrow indicates that this calendar grid was selected.

Name

Enter a name for the line grid.

Status

In this column each line grid that was added () and/or modified () after opening the dialog box is marked by a symbol.

Visible

Tick this check box for the line grids to be displayed

Type

Lets you set the basic unit of the line grid, e.g. days, weeks, etc.

Interval

Lets you set the size of the interval between the grid lines as an integer multiple of the basic unit of the grid.

Line type

When clicking on the button in this field, the **Line attributes of line grid** dialog box will appear, where you can set shape and color of the borderlines of the line grid.

Priority

Lets you set the priority of a line grid. It refers to other line grids and to layers (> 0 : in front of the layers, < 0 : behind the layers).

Use reference date

Tick this check box if the start value of the line grid should coincide with the reference date selected.

Reference date



Select the reference date from the date picker.

Adjust to reference date

Tick this check box to position the line grid on a different value of the time unit, i.e. the one defined by the reference date, for example on 13:17 of a day.

If this option is not selected, the lines of a line grid are positioned on the beginning of a time unit, for example on 00:00 h of a day.

Line format

By clicking on  you can select a line format for the line grid or create a new one in the **Administrate Line formats** dialog which you can open by clicking on . For further information about line formats see the chapter **The Administrate Line formats** dialog.

Turn

If you tick this check box, the annotations at the lines of the date line grid can be turned by 90 degrees (vertically).

Alignment

Here you can specify the horizontal alignment of the line annotations.

At top

Tick this check box to position the annotations of the lines in the line grid at the top of the Gantt graph.

At center

Tick this check box to position the annotations of the lines in the line grid at the center of the Gantt graph.

At bottom

Tick this check box to position the annotations of the lines in the line grid at the bottom of the Gantt graph.

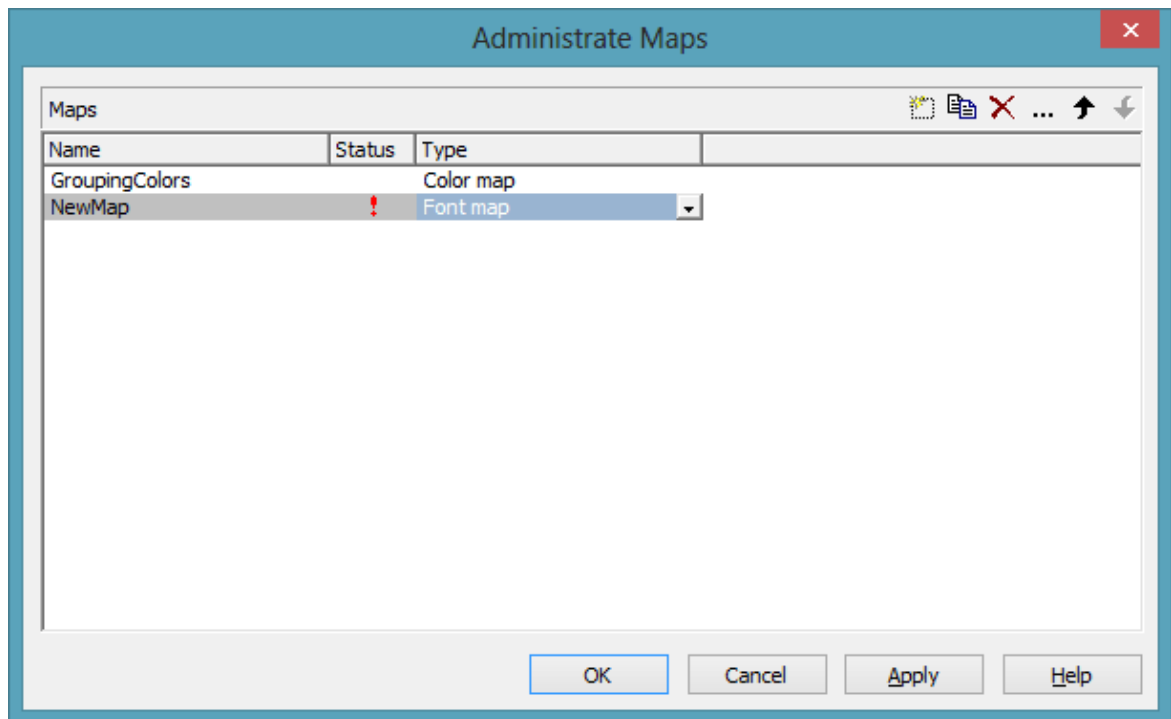
Observe DST

Tick this check box to have daylight saving time observed.

Snap target

Tick this check box to have the line grid's relevant positions defined as "snap targets" for nodes/layers to be moved.

4.23 The "Administrate Maps" Dialog Box



You can invoke this dialog by clicking the **Maps** button either on the **Objects** property page or in the **Configure Mapping** dialog box.

Name

This column lists the names of all existing maps. All names can be edited.

Status

In the **Status** column each map that has been added (🌟) and/or modified (🚨) since the dialog box was opened is marked by a symbol.


Type

Select the map type:


- Color maps
- Pattern maps
- Graphics file maps
- Fonts
- Millimetres

- Number map


Add map

 A new map will be created. You can modify its default name by double-clicking and editing it.

Copy map

 Copies the selected map.


Delete map

 The marked map in the list will be deleted. You can only delete maps that are not currently used.

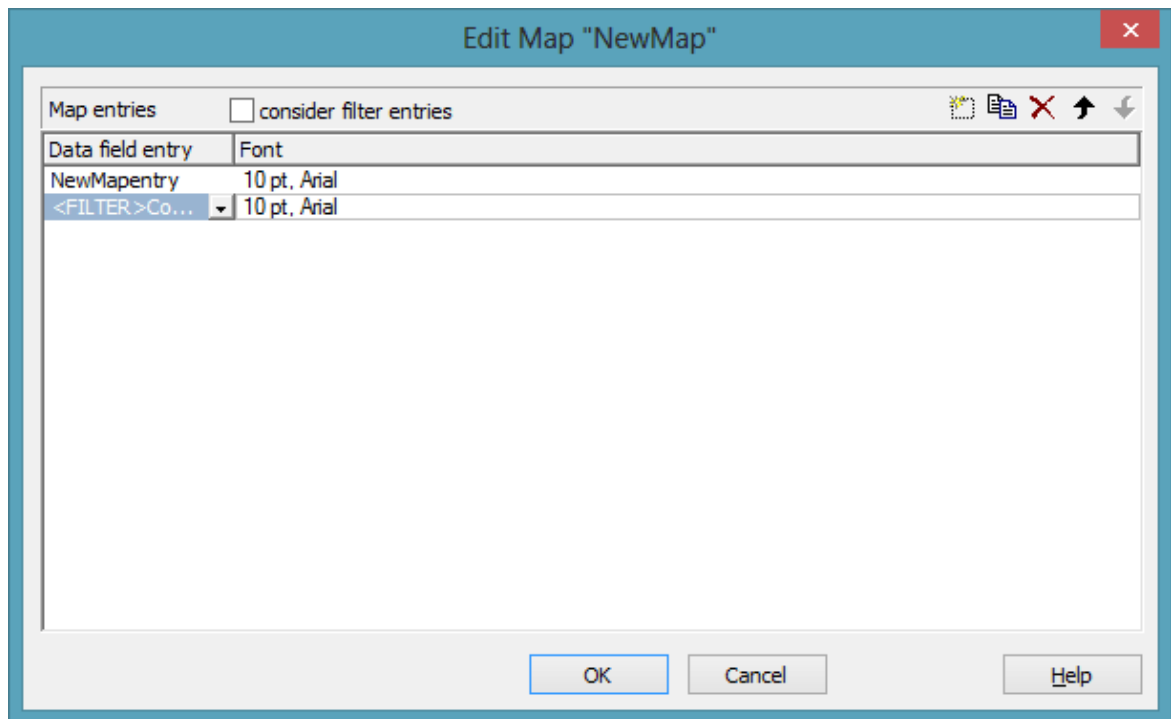
Edit map

 The **Edit Map** dialog box will appear.

Promote / demote map

 By these buttons you can move the map by one position up or down in the list.

4.24 The "Edit Map" Dialog Box



You invoke this dialog box by clicking the **Edit map** button () of the **Administrate Maps** dialog box.

In a map you can set up to 150 allocations. If you wish to set more allocations, please create a new map, e. g. as a copy of an existing one.

consider filter entries

If you have ticked this check box, not only the single values from the list of data field entries are considered as keys but also the filters which can be selected from the drop down list. Thus you can not only specify a single value as key but also a range of values.

Data field entry

Specify the entries of the data field selected for which colors or patterns and legend texts are to be assigned.

Color/Pattern

Assign colors or patterns to the data field entries. To do so, click on the corresponding field. A dialog box will open opens that lets you select a color or a pattern, respectively. The color dialog box also offers transparent colors.

Legend text

Enter a legend text for each data field entry.

Add map entry



A new map entry will be created. You can modify its default name by double-clicking and editing it.

Copy map entry



Copies the selected map entry.

Delete map entry



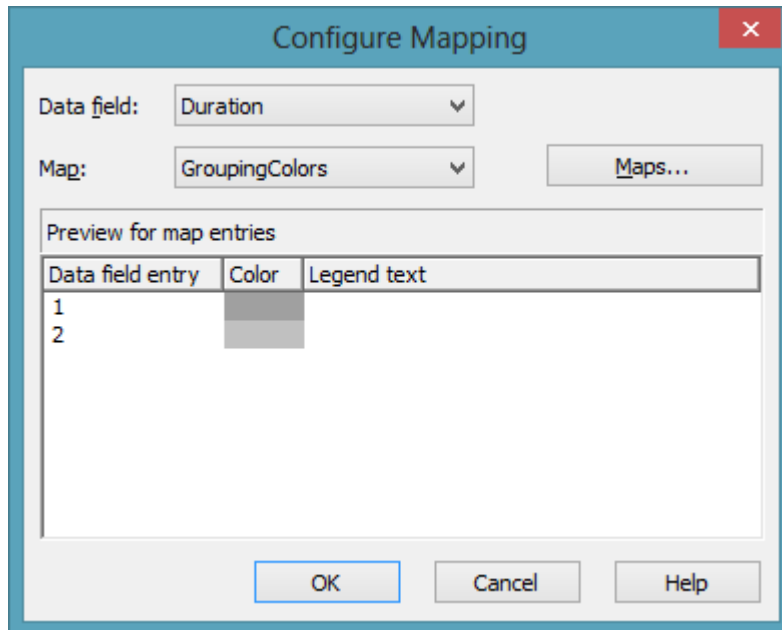
The marked map entry in the list will be deleted. You can only delete map entries that are not currently used.


promote / demote map entry



The selected map entry can be moved by one position up or down in the list.

4.25 The "Configure Mapping" Dialog Box



In this dialog box you can assign a map to a data field. You will get to it by clicking on the button  for the desired attribute in various dialogs, e.g. the dialog **Edit layer**.

Data field

Select the data field the entries of which control the desired attributes of the current object.

Map

(only activated if a data field has been specified) Select the map that depending on its type assigns the corresponding attributes to each data field entry.

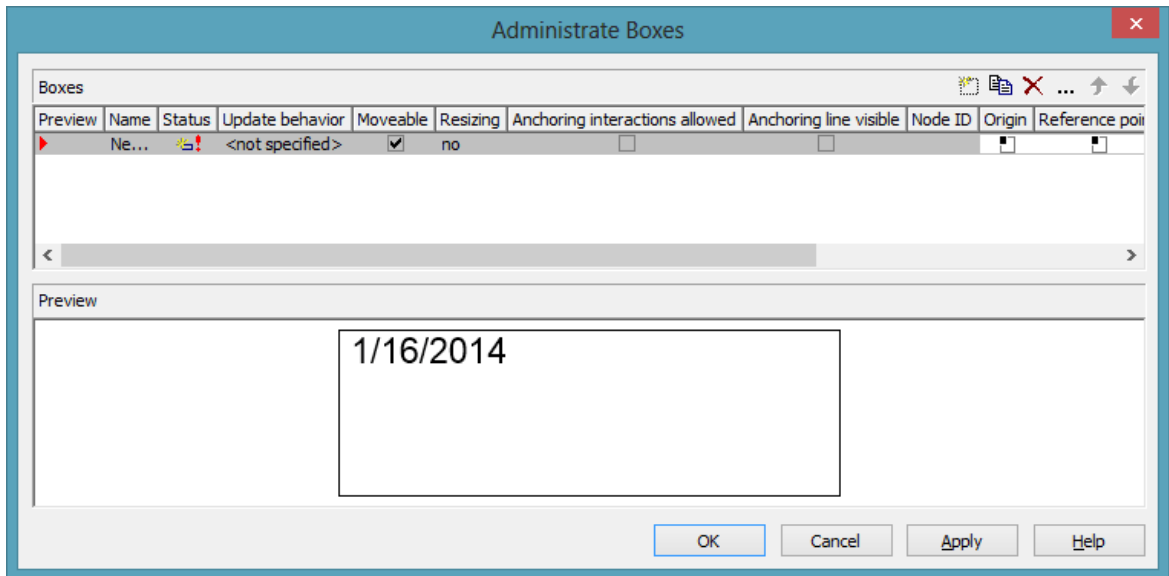
Maps

Opens the **Administrate Maps** dialog box, where you can create, edit, copy or delete maps.

Preview for map entries

The preview shows the selected map: the data field entries and the attributes assigned to them.

4.26 The "Administrate Boxes" Dialog Box



You can get to this dialog box by the **Objects** property page. In the diagram area, boxes can be displayed, that you can administer by the above dialog.



Preview

The box marked in the **Preview** column is displayed in the preview window.

Name

Lists the names of all existing boxes. The names can be edited.

Status

In the **Status** column all boxes added () and / or modified () after the dialog box was opened are marked by a symbol.

Update behavior

Select an update behavior for this box. Leaving the setting to <not selected> means that the setting for boxes made in the **Edit Update behavior** dialog will apply

Moveable

By moving a box its offset will be modified. Activate this check box if the box is to be moveable in the diagram at run time. Deactivate the check box if you do not want the box to be moved at run time.

Resizing

Here you can specify whether the size of a box can be modified interactively. You can select whether only height, only width or both height and width can be modified. When the pointer is placed on the frame of the box, its form changes to a double-headed arrow. Now hold the left mouse button pressed and change width and/or height by moving the mouse in the desired direction.

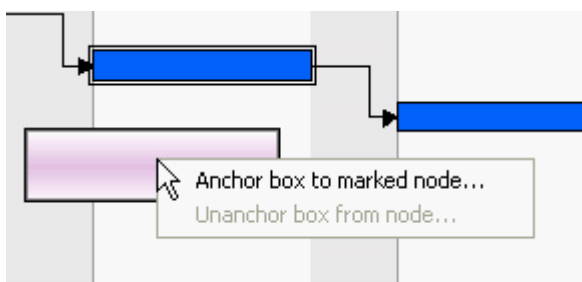
Tipp If you have selected **width and height** you can place the pointer on the corner of the box and both dimensions can be modified at the same time.

Anchoring interactions allowed

Specify whether anchoring interactions (by mouse or over context menu) are possible. Thus the user can tie boxes to nodes or untie them again.

Boxes can be anchored to nodes either interactively (mouse + Shift key or context menu) or by using the corresponding API properties and methods.

- **Anchoring by mouse:** Point with the mouse to the box you want to tie to a node and press the Shift key. A little anchor appears. Hold the Shift key pressed and draw a line between the box and the desired node. The box is now anchored to the node. If you have ticked the check box **Anchoring line visible** a line is displayed. Follow the same steps to untie the box again.
- **Anchoring over contextmenu:** Mark the node to which you want to anchor the box and select **Anchor box to marked node** from the context menu of the box. If the context menu does not pop up, you have to activate the option **Show context menu for the box** on the property page **General**.



Select **Unanchor box from node** to untie the box again.

If you want to tie the box to another node carry out the same steps as described above, either by mouse or over context menu.

- **Anchoring by API** Please see the API Reference Guide for a detailed description of the property **AnchoringInteractionsAllowed** and the method **AnchorToNode** of the object **VcBox**

A box which was anchored can still be moved interactively (provided that you have ticked the check box **Moveable**).

If you move a node which is anchored to a box, the box is moved as well. If the node is collapsed, the box is collapsed as well, thus becoming invisible. When the node is expanded the box is visible again.

When a box is tied interactively to a node, its position on the screen will be maintained. The offset values which are used as basis are converted according to the reference points (Origin, ReferencePoint). If, for example, a box with a certain offset refers to a chart at the top left (origin) and then is anchored to a node, an offset to the top left node is calculated automatically. This makes sure that the position on the screen will not be altered. If the box is untied from the node the calculation is carried out backwards.

This method is applied as well when using the API property **AnchorToNode** but not when setting the property **NodeID**.

Anchoring line visible

Specify whether a line between the reference points (origin, reference point) of a node and of a box which are anchored is displayed.

Node ID

Here you can enter a string which is interpreted as Node ID and is used for identifying the node to which the respective box shall be tied. An empty string implicates that the box will not be anchored to a node.

Note: It is neither checked whether the syntax of the string is correct nor whether the node exists. If the node does not exist, no anchoring will take place.

Origin

By the properties **Origin**, **Reference point**, **X Offset** and **Y Offset** you can position a box in the diagram area. The relative position of the boxes is independent of the current diagram size.

Specify the origin, i. e. the point of the diagram from which the offset to the reference point of the box is measured. Possible values: top left, top centered, top right, centered left, centered centered, centered right, bottom left, bottom centered, bottom right.

Reference point

Set the reference point of the box, i. e. the point of the box from which the offset to the origin is measured. Possible values: top left, top centered, top right, centered left, centered centered, centered right, bottom left, bottom centered, bottom right.

X Offset

Set the distance between origin and reference point in x direction.

Y Offset

Set the distance between origin and reference point in y direction.

Frame

If you click on the **Frame** field, an **Edit** button will appear that lets you open the **Line Attributes** dialog box. In the dialog box you can specify the type, the thickness and the color of the box frame line.

Priority

Set the drawing priority of the box in relation to other objects in the diagram (nodes, grids, etc.). The priority of nodes is 0. If the priority of boxes is higher than the one of nodes, the boxes may hide the nodes and may thus inhibit interactive access.

Visible

Activate this check box if the box is to be visible at run time.

Box format

The current box format of the box is displayed here. If you click this field, two buttons will appear:



From the select box you can choose a box format.



By the **Edit** button you can get to the **Administrate Box Formats** dialog box.

Add box



A new box will be created. You can modify its default name by double-clicking and editing it.

Copy box



The Box selected will be copied.

Delete box



The box marked in the list will be deleted.

Edit box



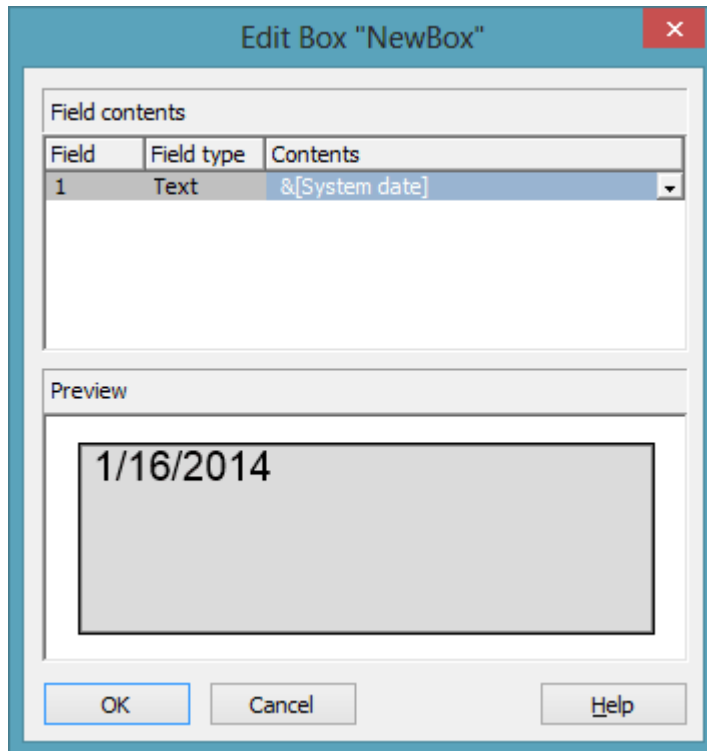
The **Edit Box** dialog box will appear.

Promote / demote box



By these buttons you can move the box by one position up or down in the list.

4.27 The "Edit Box" Dialog Box



You can get to this dialog by the **Objects** property page and the dialog box **Administrate Boxes** by clicking on the the **Edit box** button. This dialog box will also appear at run time when double-clicking on a box.

Field

This column contains the numbers of the box fields. (The number of fields depends on the selected box format.)

Field Type

This column displays the field types (text or graphics).

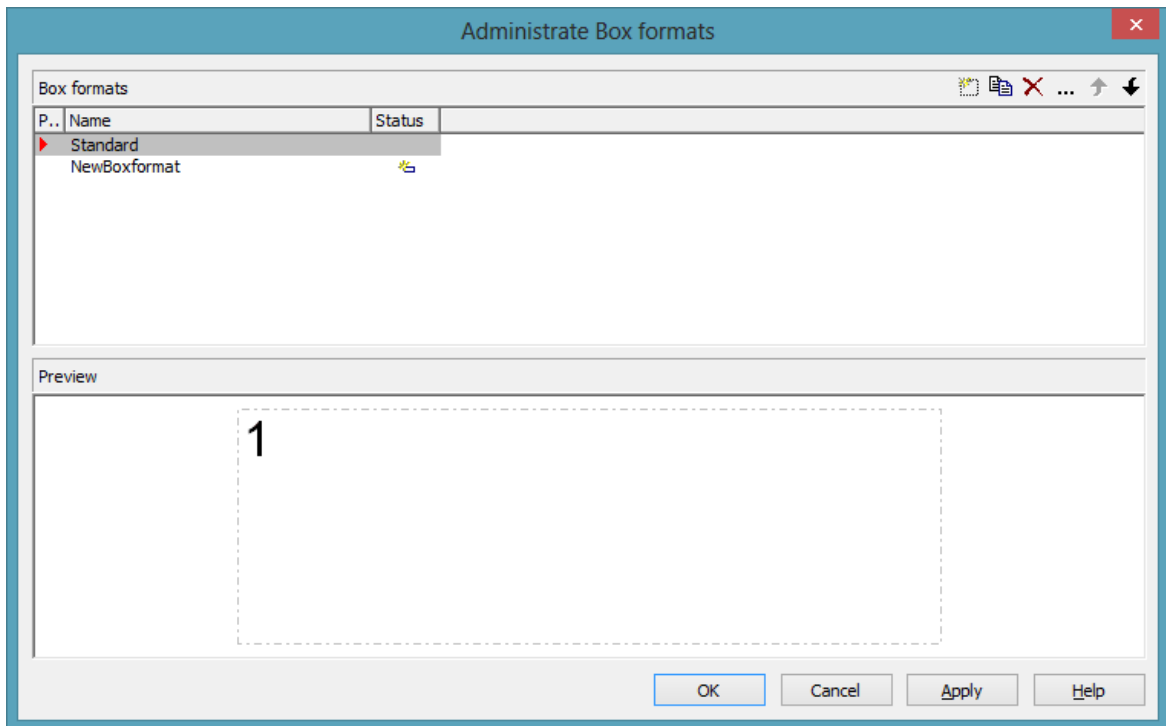
Contents

Type the contents of the field or a graphics file name here.

If a text field contains more than one line, you can use "`\n`" in the text string to separate two lines of the text field (Example: "Line1\nLine2"). Otherwise the lines will be separated at blanks.

Graphics formats available: WMF, JPG, BMP, GIF, PCX, PNG, TIF.

4.28 The "Administrate Box Formats" Dialog Box



This dialog you can get to by the **Objects** property page.



Preview

The preview window shows the box format marked in the **Preview** column.


Name

Lists the names of all existing formats. The names can be edited.

Status

In the **Status** column the formats added () or modified () after the dialog box was opened are marked by a symbol.


Add box format

 A new format will be created. You can change its default name by double-clicking and editing it.

Copy box format

 The marked format will be copied.



Delete box format

 The marked format in the list will be deleted. You can only delete formats that are not being used.

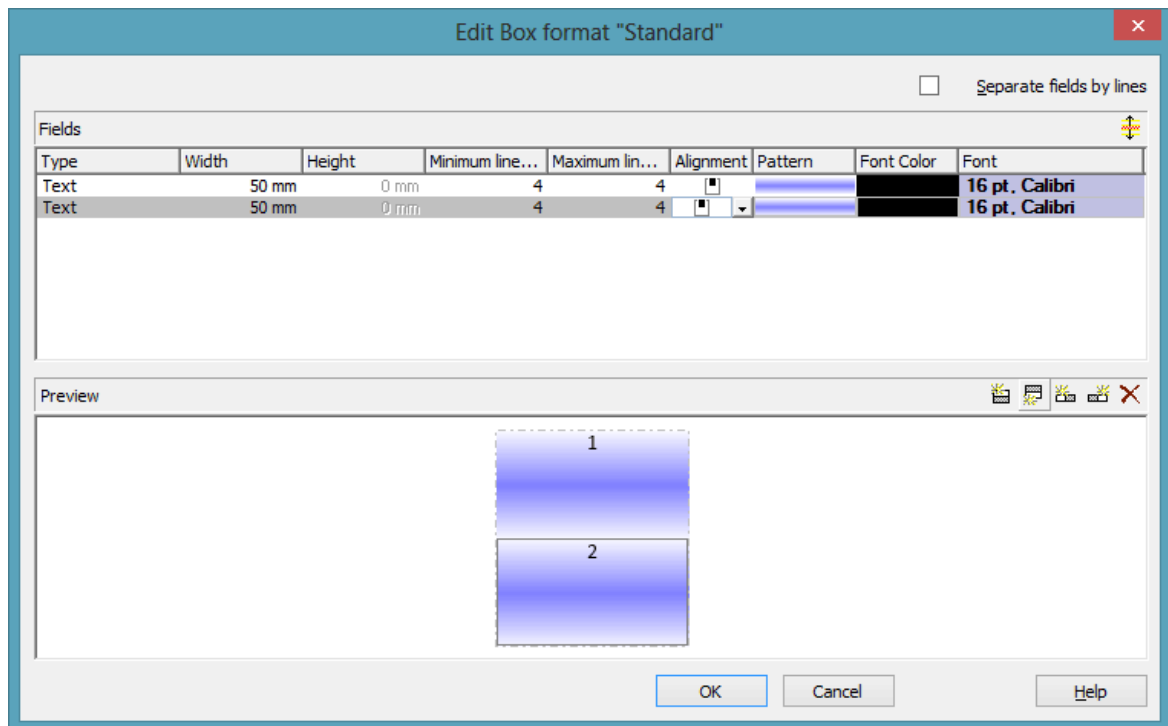
Edit box format

 You will get to the **Edit Box Format** dialog box.

Promote / demote box format

  By these buttons you can move the selected format by one position upward or downward in the list.

4.29 The "Edit Box Format" Dialog Box



This dialog box will appear if you activate the **Administrate Box Formats** dialog box on the **Objects** property page and then click on the **Edit box format** button.

Separate fields by lines

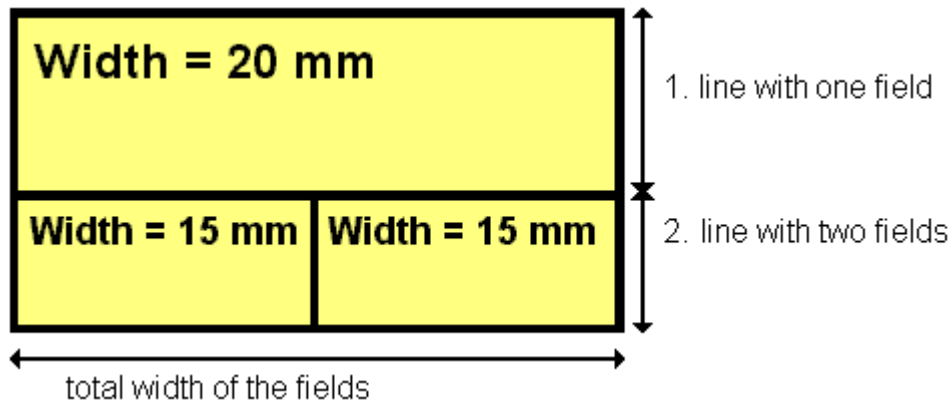
Activate this check box if the box fields are to be separated by lines.

Type

Select the field type: text or graphics.

Width

Specify the width for the selected field (in mm). The maximum width of a field is 200 mm. If the rows are split into two or more fields and the total widths of the rows vary, the total width will be equal to the width of the widest row.



Height

(only for the type graphics) Specify the minimum height for the selected field (in mm). The maximum height is 200 mm.


Minimum/Maximum line count

(only for the type text) Specify the minimum/maximum number of lines of text that can be displayed in the current field. Each field can contain a maximum of nine lines of text.

Alignment

Specify the alignment of the content of the selected field (9 possibilities).

Pattern

Select the fill pattern and color for the current field. By clicking on  you open the **Edit pattern attributes** dialog where you can specify a pattern, a background color and, if needed, a second pattern color. You can define your own colors in addition to the ones suggested. Also, transparent colors are available.

Font Color

(only for the type text) Indicates the font color for the current field.



By the **arrow** button you can open the color picker to select a font color.

Font

(only for the type text) Indicates the font style for the current field.


 The Windows **Font** dialog box will appear.

Apply selected property to all fields

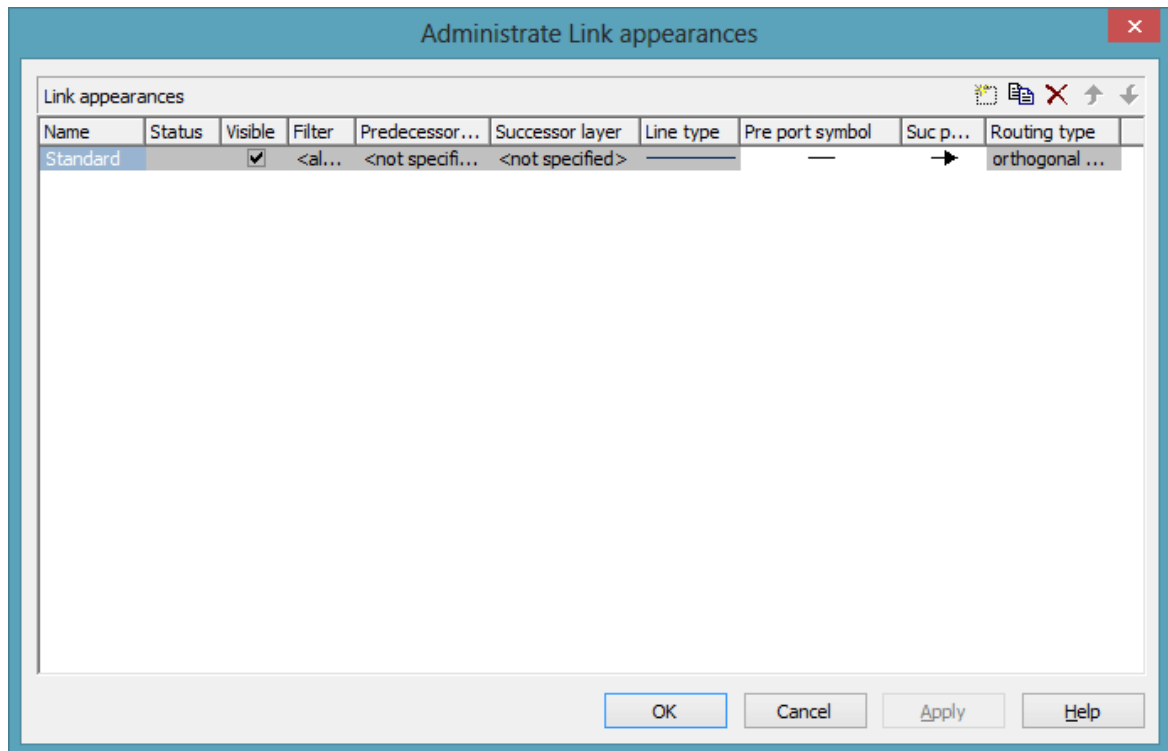
 Applies the marked property to all fields.

Preview

The current fields of the box format are displayed in the preview window. If you click on a field, you can modify its attributes in the **Fields** table.

 With the help of the buttons above the preview window you can add new fields or delete the marked field. You also can use the Del button to delete fields.

4.30 The "Administrate Link Appearances" Dialog Box





You can get to this dialog by clicking the **Link appearances** button on the **Objects** property page.

Name

This column displays the names of the link appearances available. The names can be edited.

This feature can also be set by the property **LinkAppearanceName**.

Status

In the **Status** column each link appearance that has been added () and/or modified () since the dialog box was opened is marked by a symbol.

Visible

This check box lets you specify whether the links between the nodes should be displayed. This feature can be also set by the property **VcLinkAppearance.Visible**.

Filter

This column displays the filter used for a link appearance. From the select box you can select an appropriate filter.

This feature can also be set by the property **VcLinkAppearance.Filter-Name**.

Predecessor layer

Specify to which layer of the predecessor node the link is to be drawn. If the selected layer is not assigned to a node, the link will be drawn to the first visible layer of this node.

This feature can also be set by the property **VcLinkAppearance.PredecessorLayerName**.

Successor layer

Specify to which layer of the successor node the link is to be drawn. If the layer selected is not assigned to a node, the link will be drawn to the first visible layer of this node.

This feature can also be set by the property **VcLinkAppearance.Successor-LayerName**.

Line type

Clicking on an entry in this column will cause an **Edit** button to occur, by which you can get to the **Edit Line attributes** dialog box. There you can set type, thickness and color of the line.

This feature can also be set by the property **VcLinkAppearance.LineType**.

Pre port symbol

Select a port symbol for a link that visually accentuates the junction of the link and the predecessor node.

This feature can also be set by the property **VcLinkAppearance.-PredecessorPortSymbol**.

Suc port symbol

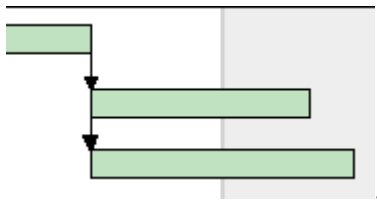
Select a port symbol for a link that visually accentuates the junction of the link and the successor node.

This feature can also be set by the property **VcLinkAppearance.Successor-PortSymbol**.

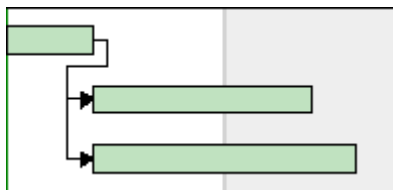
Routing type

This field allows to select a routing type. As the first row of the table containing the link appearance types is reserved for the default link appearance, the item <not specified> is selectable only from the second row on. If <not specified> has been selected, a routing type is used which is further up the list of the LinkAppearance objects.

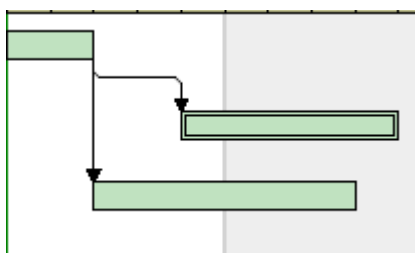
The routing type can also be set by the **VcLinkAppearance** property **RoutingType**.



Straight-lined link type



Orthogonal link type



Orthogonal distinguishable link type

Add link appearance



A new link appearance will be created. You can modify its default name by double-clicking and editing it.

Copy link appearance



Copies the selected link appearance.

Delete link appearance



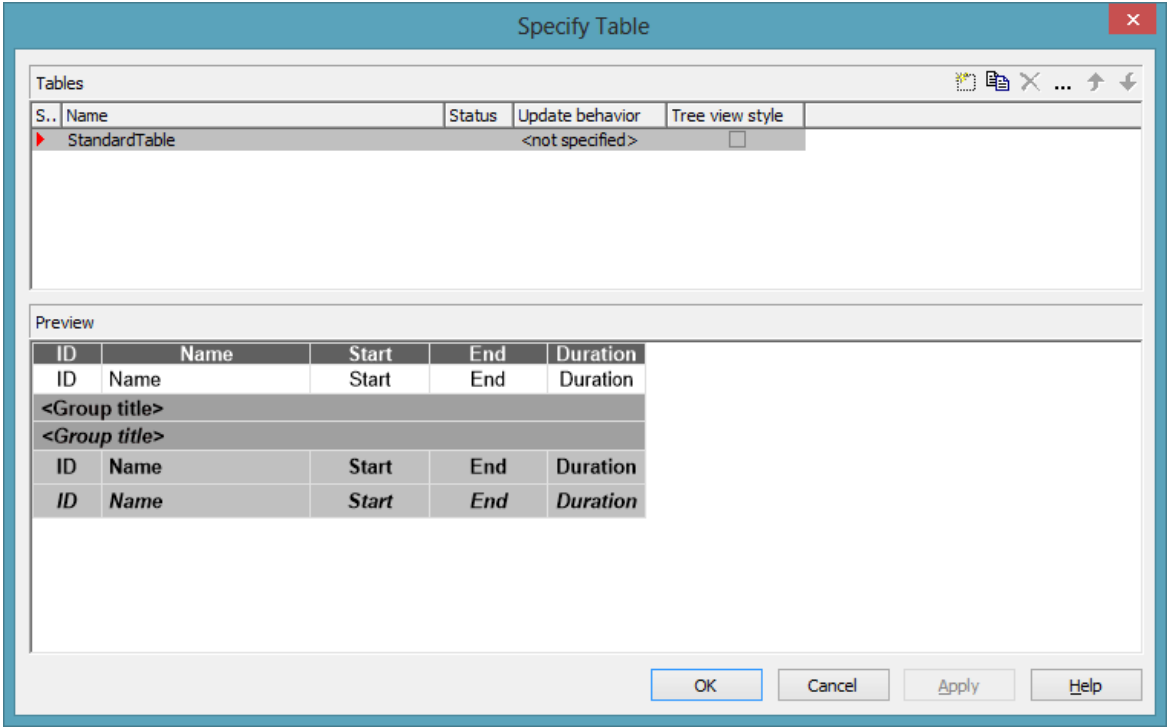
The marked link appearance in the list will be deleted. You can only delete link appearances that are not currently used.

Promote / demote link appearance



By these buttons you can move the link appearance by one position up or down in the list.

4.31 The "Specify Table" Dialog Box



In this dialog box you can establish and administer tables.



Preview

The table marked by a small red arrow in the **Preview** column is displayed in the preview window in the lower half of the dialog above. It simultaneously is the table presently edited.

Name

Lists the names of all tables that are defined. The names can be edited.

Status

In this column each table that has been added () and/or modified () since the dialog box was opened is marked by a symbol.

Update behavior

Select an update behavior for this table. Leaving the setting to <not selected> means that the setting for tables made in the **Edit Update behavior** dialog will apply

Tree view style

If this check box is activated, nodes will be arranged in tree view style, with lines tracing the logical tree structure. In either case, plus or minus symbols mark levels.

ID	Name
[-]	
[-]	Snyder
	Smith

ID	Name
[-]	
1	Snyder
2	Smith

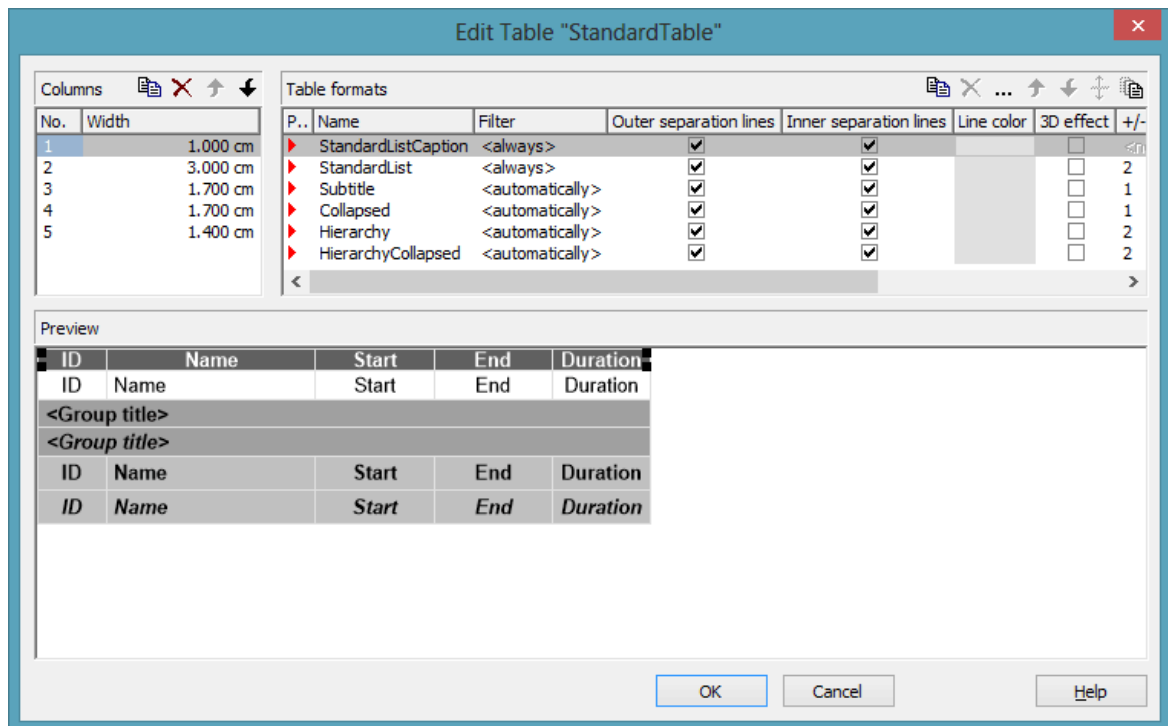
Pictures above: a group with and without the tree view style set

Add / copy / delete / edit / promote / demote table;



By these buttons you can create, copy or delete the marked table or move it by one position up or down in the list, respectively. The latter may serve to sort the names and thus contribute to improved clarity but has no function in terms of priority.

4.32 The "Edit Table" Dialog Box



In this dialog box you can edit a table.

Columns

The **Columns** list contains the **No.** and the **Width** of each table column. The width can be varied by steps of 1 mm in the range from 0 to 10 cm.

You can define 100 columns at maximum. The sequence of the table columns in the **Columns** list corresponds to the sequence of the table columns in the chart.





The buttons above the **Columns** list allow to copy or delete table columns or to modify their position in the list.

Table Formats



The **Table Formats** list lets you specify different table formats:


- **Preview:** A table format marked by a red arrow is displayed in the preview window.
- **Name:** A table format by default has a name. **StandardListCaption** is the name of the table format of the table caption. The names can be edited only for the table formats **ListFormat2**, **ListFormat3** and for all table formats that you have specified yourself.

- **Filter:** A table format is combined with a filter that selects the activities to which the table format is to apply. When several filters of this list apply to an activity, the table format of the highest priority will be used. The sequence of the filters in the list of the **Table formats** field of the dialog box inversely corresponds to their priority: the top filter has lowest priority. Four pre-defined filters exist. The format of the <interfaceNode> filter applies to nodes interfacing the nodes selected. The <never> filter never applies. It practically serves as a template for copying. The <automatically> filter applies to nodes of the same group level; the level is to be specified. The <always> filter collects all nodes that were not selected by other filters. It makes sense to put it at the top; in addition, it cannot be deleted.
- **Outer/Inner separation lines:** Specify whether the table fields are to be separated by lines outside and/or inside the table fields.
- **Line Color:** You can assign a line color to a format.
- **3D effect:** Specify whether the table fields are to be highlighted by a 3D effect.
- **+/- column:** Specify whether in a column + or - shall be displayed for collapsing or expanding subordinated lines. Select the appropriate column from the drop down list.
- **Indent column:** Specify the column to be indented. This only works if there are lines (nodes) subordinated to this line (node). Then the first subordinated line will be indented. If the **automatically** filter is assigned, the column in which +/- is displayed will be indented.
- **Indent width:** Specify by how much (in mm) the column shall be indented.

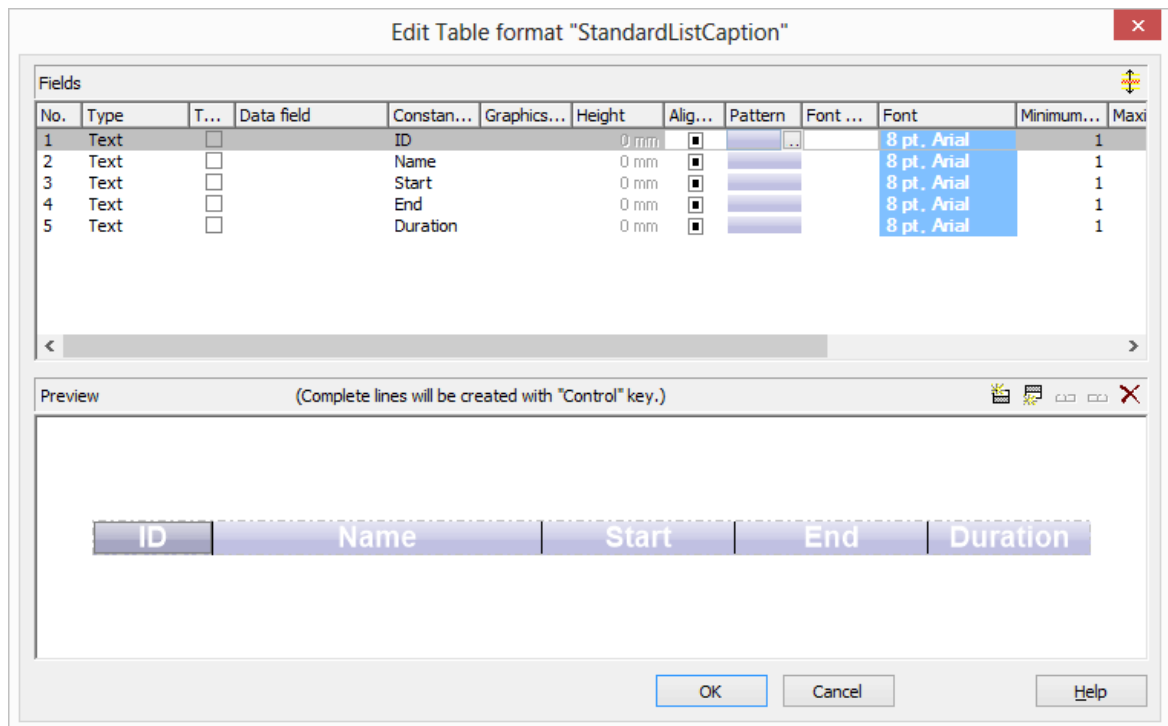
  ... By using these buttons at the top of the **Table Formats** list you can copy or delete table formats or open the **Edit Table Format** dialog.

Note: For the table format **StandardListCaption** (table caption) attributes cannot be assigned using maps.

  By using these buttons you can move the table formats in the list, except for the first and the second one that are immobile.

 If you have changed the attributes **Outer separation lines** or **Inner separation lines** of a table format and then click on this button, the changed attribute will be applied to all table formats.

4.33 The "Edit Table Format" Dialog Box



In this dialog box you can edit a table format (row type).

No.

Number of the table format field: This number cannot be edited. It is used as an index that allows to access the table format field by API calls.

If you create a new table format field in the preview window, the index preliminarily will receive "?" instead of a number. The "?" will be replaced by a number when the dialog is left by **OK**, which can be verified when re-opening the dialog.

Type

Please select the field type: **text**, **graphics** or **multi-state**. Multi-state fields are used for example to trigger a rotating sequence of different states and of the associated data fields when clicked.

Text/graphics combined

If this check box is activated, in the table format field a text and a graphics can be combined as follows:

- **Type:** Text, **Text/graphics combined:** no: Only text will be displayed (as specified for **Data field** or for **Constant text**).
- **Type:** Graphics, **Text/graphics combined:** no: Only a graphics will be displayed (as specified for **Graphics file name**).
- **Type:** Text, **Text/graphics combined:** yes: Text (as specified for **Data field** or for **Constant text**) and a graphics (as specified for **Graphics file name**) will be displayed.
- **Type:** Graphics, **Text/graphics combined:** yes: Only a graphics will be displayed (as specified for **Graphics file name**). Text (as specified for **Data field**) is visible only in a tooltip. If possible, it will be displayed as hyperlink.

Data field

Select the data field the content of which is to be displayed in the current field. In addition to the data fields defined in the data definition table, you can select one of the following options:

- <Group title>: the code specified for the current grouping level
- <Row number>: consecutively numbered rows

If the content of a data field does not fit into the field, excess characters will be truncated when displayed.


Constant Text

(only if no data field has been specified) Type a constant text to be displayed in the current field.


Graphics file name

Indicates the name and the directory of the graphics file to be displayed in the selected table format field.


If you click on a **Graphics file name** field, two buttons will appear:

Click on the first button  to open the Windows dialog box **Choose Graphics File**. It lets you select the graphics file to be displayed in the selected table format field.

If a relative file name was chosen, at run time the file will at first be searched in the path set by the VcGantt property **FilePath**. If it is not found there, it will be searched in the current directory of the application and in the installation directory of the VARCHART XGantt control.

 Click this button to use a map for displaying graphics in table format fields depending on the node data. The **Configure Mapping** dialog box will open which lets you combine a map and a node data field, the map assigning graphics files in dependence of the data field entries.

If in the **Configure Mapping** dialog box only a data field, but no map is selected, the content of the data field will be used as a name of a graphics file. If in the data field or in the map no valid graphics file name can be found, the file name specified in the **Symbol file field** will be used.

After a node data field and a map have been combined, the arrow on the second button will turn to bold: .

 When you leave the **Symbol File Name** field, a symbol indicates that a map was assigned to a data field.

When the graphics is displayed, the color of the pixel in the top left corner will be replaced by the color of the diagram background, and so will all pixels of the same color. Therefore all pixels of the graphics that show the same color as the top left corner pixel are transparent.



Height



(only for the type graphics) Specify the minimum height for the selected field (in mm). The maximum height is 99 mm.

Alignment

Specify the alignment of the content of the selected field (9 possibilities).

Pattern

This field lets you set the default background pattern and colors of the table format. By clicking on  you open the **Edit pattern attributes** dialog where you can specify a pattern, a background color and, if needed, a second pattern color by clicking on . You can define your own colors in addition to the ones suggested. Transparent colors are also available.

By clicking on  you open the **Configure Mapping** dialog box. Here you can configure data-dependent patterns and colors. If a mapping has been configured, the arrow on the button will be displayed in bold (.

Font Color

Indicates the font color for the current field. If you click on the field, two buttons will appear:



By the **arrow** button you can open the color picker to select a font color.



By the second button you can get to the **Configure Mapping** dialog box. It allows to assign font colors in dependence of data.



If colors were mapped, the arrow on the button will appear solid.

Font

Indicates the font style for the current field. If you click on the field, two buttons will appear:



The Windows **Font** dialog box will appear.



By the second button you can get to the **Configure Mapping** dialog box. It allows to assign fonts in dependence of data.



If fonts were mapped, the arrow on the button will appear solid.

Minimum/Maximum line count

(only for the type text) Specify the minimum/maximum number of lines of text that can be displayed in the current field. Each field can contain a maximum of nine lines of text.

Spacing

Specify the spacing in percent.

Wrapping

Specify the wrapping of rows.

Hor. Margins (left/right)/ Ver. margins (top/bottom)

Specify the margins of the table format fields.

+/- column

Specify whether + or - for collapsing or showing further lines shall be displayed.

Indent column

Specify whether the column shall be indented.

Preview

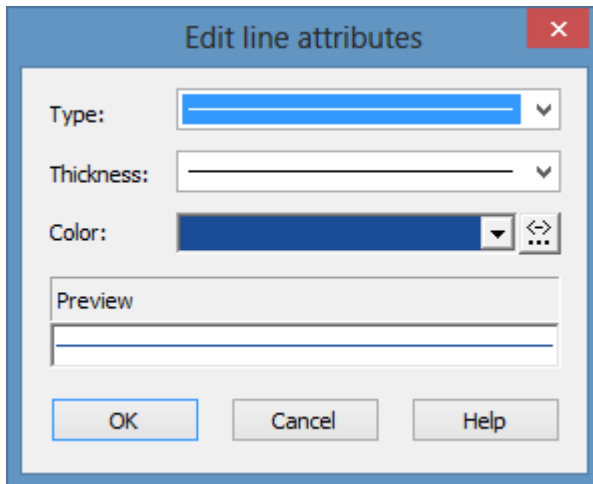
The current fields of the table format are displayed in the preview window. If you click on a field, you can modify its attributes in the **Fields** table.




With the help of the buttons above the preview window you can add new fields or delete the marked field. You also can use the Del button to delete fields.

The first four buttons (for adding new fields) are only activated, if it is actually possible to create a new field beside the field marked. This depends on the number of columns of the current table format specified in the **Edit Table** dialog.

4.34 The "Edit Line Attributes" Dialog Box



This dialog which can in each case be invoked by clicking on  is available for hierarchy and grouping, for calendar grids, for the bar appearance, for filling of curves and the numeric scales in a histogram, for the link appearance, for intervals and for box frames.

Type

Select the line type (dashed, dotted etc.).

Thickness

Define the line thickness.

Color

Select the line color.



This button will open the **Configure Mapping** dialog box where you can specify the line color data-dependent.

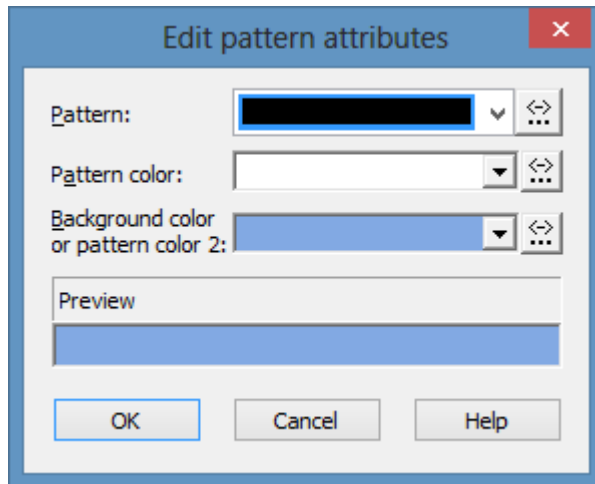


After having mapped the line color, the arrow on the button will appear bold.

Preview

The line appearance based on the current settings is displayed in this field.

4.35 The "Edit Pattern Attributes" Dialog Box



The pattern dialog which can be invoked by clicking on is available for filling of curves in a histogram, calendar grids, group title, intervals, time scale sections, box, line and table formats, layers and for node lines.

This button will open the **Configure Mapping** dialog box where you can specify the pattern, pattern color, background color or background color 2 data-dependent.

After having mapped one/several pattern attributes, the arrow on the button will appear bold.

Pattern

Here you can select a fill pattern.

Pattern color

Select the foreground color of the fill pattern.

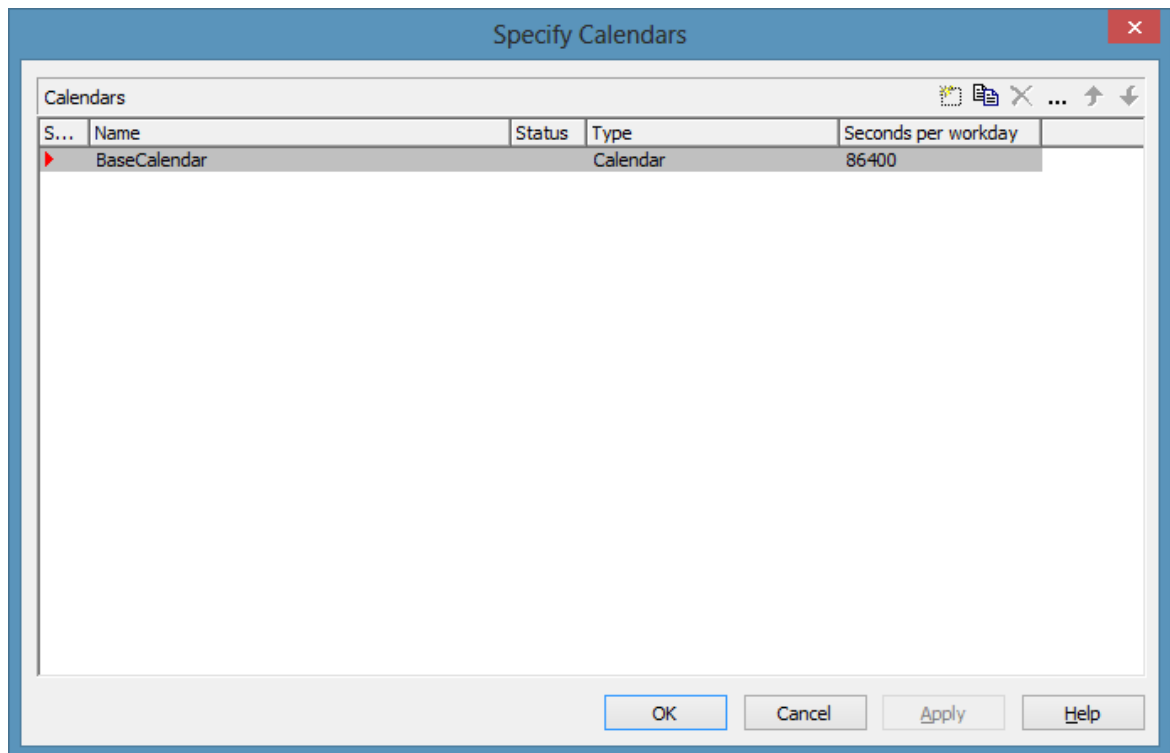
Background color or pattern color 2

Select the background color or a second pattern color.

Preview

The pattern based on the current settings is displayed in this field.

4.36 The "Specify Calendars" Dialog Box



You can get to this dialog via the **Objects** property page. Define one calendar per line in the table.



Selected

The calendar marked by a small arrowhead in the **Selected** column is used for the calendar grid.

Name

Lists the names of all calendars defined.

Status

In the **Status** column each calendar that has been added () and/or modified () since the dialog box was opened is marked by a symbol.

Type

Specify the calendar type. Besides ordinary calendars shifts calendars are available, too.

Seconds per Workday

Specify how much seconds the workday has got.

Add calendar



Click on this button to add a calendar.

Copy calendar



The marked calendar is copied.

Delete calendar



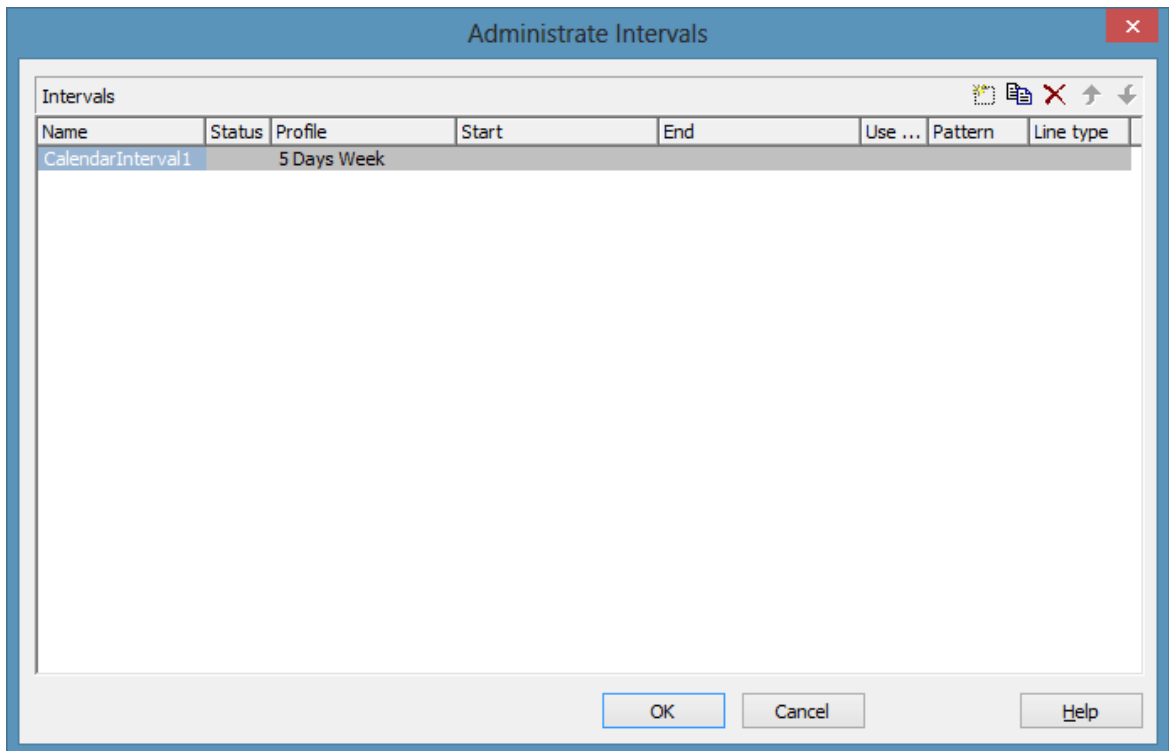
The marked calendar is deleted.

Edit calendar



You will reach the **Edit Calendar** dialog box.

4.37 The "Administrate Intervals" Dialog Box (Calendar)





In this dialog box you can edit intervals.



Name

Lists the names of all intervals. All names can be edited.

Status

In this column each interval that has been added () and/or modified () since the dialog box was opened is marked by a symbol.

Profile

Here you can select a profile for your interval by clicking . If you want to edit the profile click on  beside its name to open the **Administrate Calendar profiles** dialog.


Start/End

In this field you can set the beginning or end of of an interval. The date can be easily entered or modified by using the spin control.


Use graphical attributes

If this option is selected, you can select an display a pattern and a line type for the interval. The option is only active for the profil types <Working time> and <Nonworking time>.

Pattern

Click on  to open the dialog **Edit pattern attributes**.

Line type

Click on  to open the dialog **Edit line attributes**.

Add interval



A new interval will be created. You can modify the marked name by double-clicking and editing it.

Copy interval



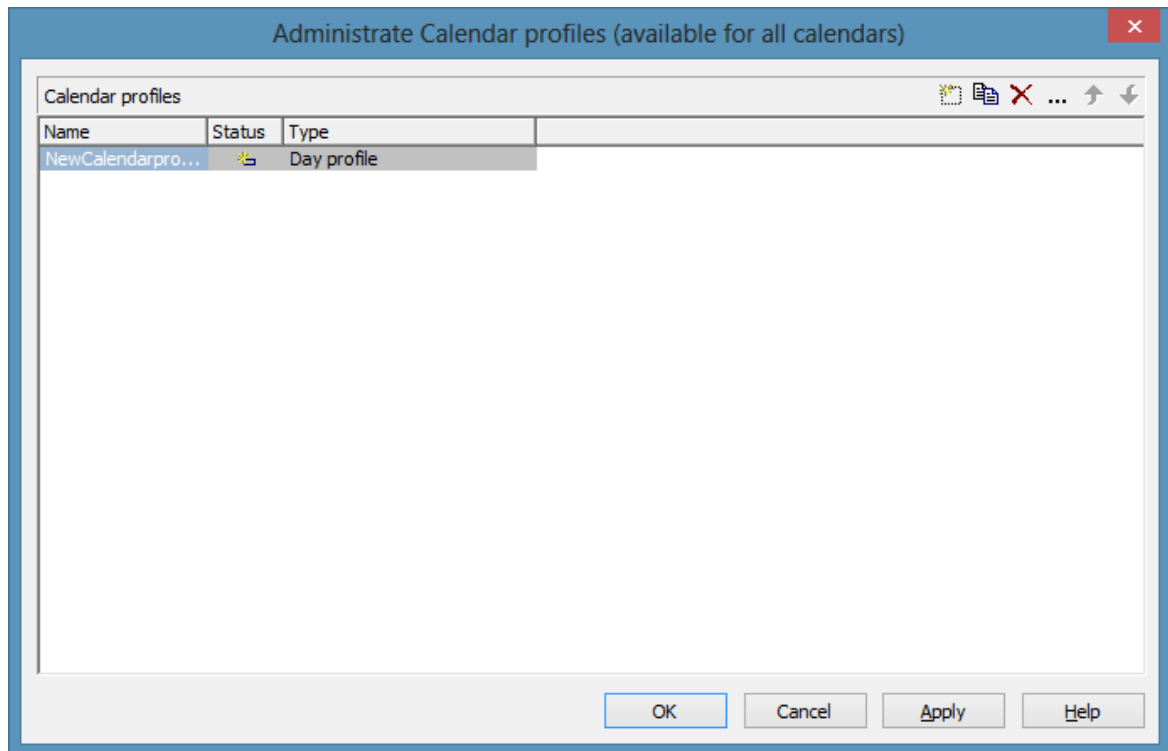
Click on this button to copy the marked interval.

Delete interval



Click on this button to delete the marked interval.

4.38 The "Administrate Calendar Profiles" Dialog Box



In this dialog you can create and modify calendar profiles.

Name

Lists the names of all calendar profiles. All names can be edited.

Status

In this column each calendar profile that has been added () and/or modified () since the dialog box was opened is marked by a symbol.

Type

By clicking you can select the calendar profile type. You can choose between <Day profile>, <Week profile>, <Year profile> and <Variable profile>.

Add calendar profile



A new calendar profile will be created. You can modify the marked name by double-clicking and editing it.

Copy calendar profile



Click on this button to copy the marked calendar profile.

Delete calendar profile



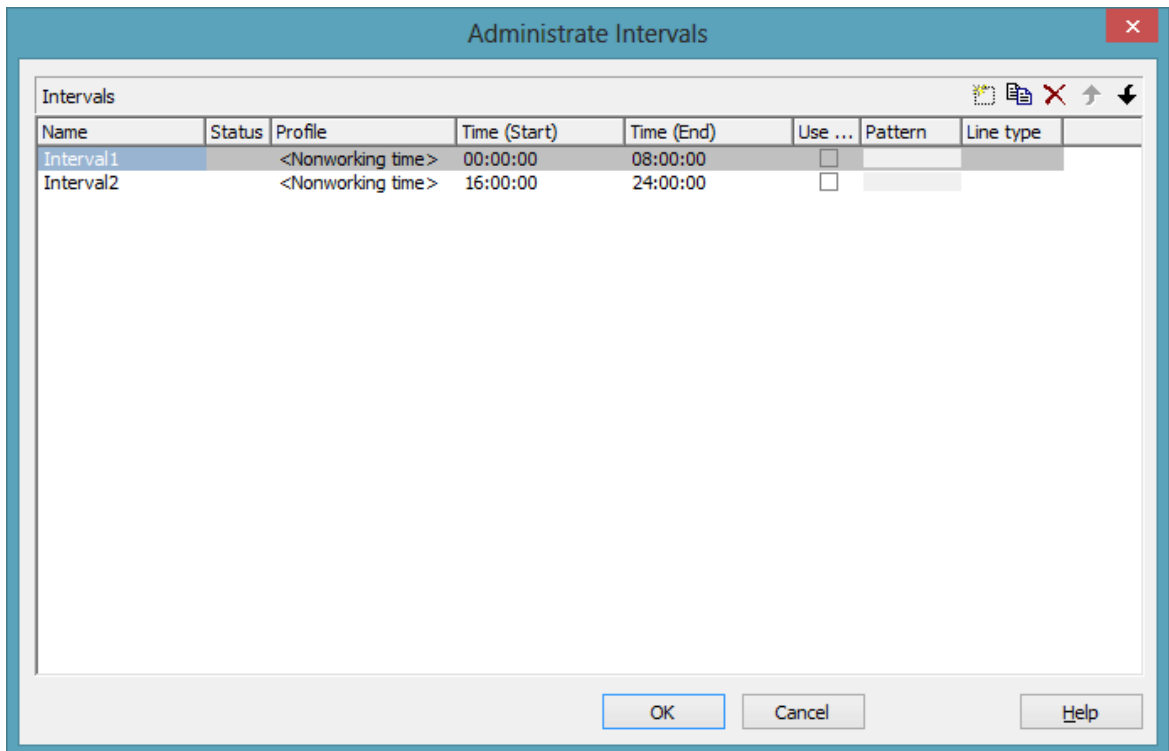
Click on this button to delete the calendar profile.

Edit calendar profile



You will reach the **Administrate Intervals** (Calendar profiles) dialog box.

4.39 The "Administrate Intervals" Dialog Box (Calendar Profiles, Profile Type <Day Profile>)





You can get to this dialog if you activate the dialog box "Administrate Calendar Profiles" on the "Objects" property page, and then click on the "Edit" button of the calendar profile. The different types of profiles offer different setting options. This dialog serves to create and modify intervals of a day profile.

Name

Lists the names of all intervals. All names can be edited.

Status

In this column each interval that has been added () and/or modified () since the dialog box was opened is marked by a symbol.

Profile

Here you can select a profile for your interval by clicking .

Time Start/Time End

In this field you can set the start or end time of an interval by clicking on the arrow buttons.


Use graphical attributes

If this option is selected, you can select an display a pattern and a line type for the interval. The option is only active for the profil types <Working time> and <Nonworking time>.


Pattern

Click on  to open the dialog **Edit pattern attributes**.


Line type

Click on  to open the dialog **Edit line attributes**.

Add interval

 A new interval will be created. You can modify the marked name by double-clicking and editing it.

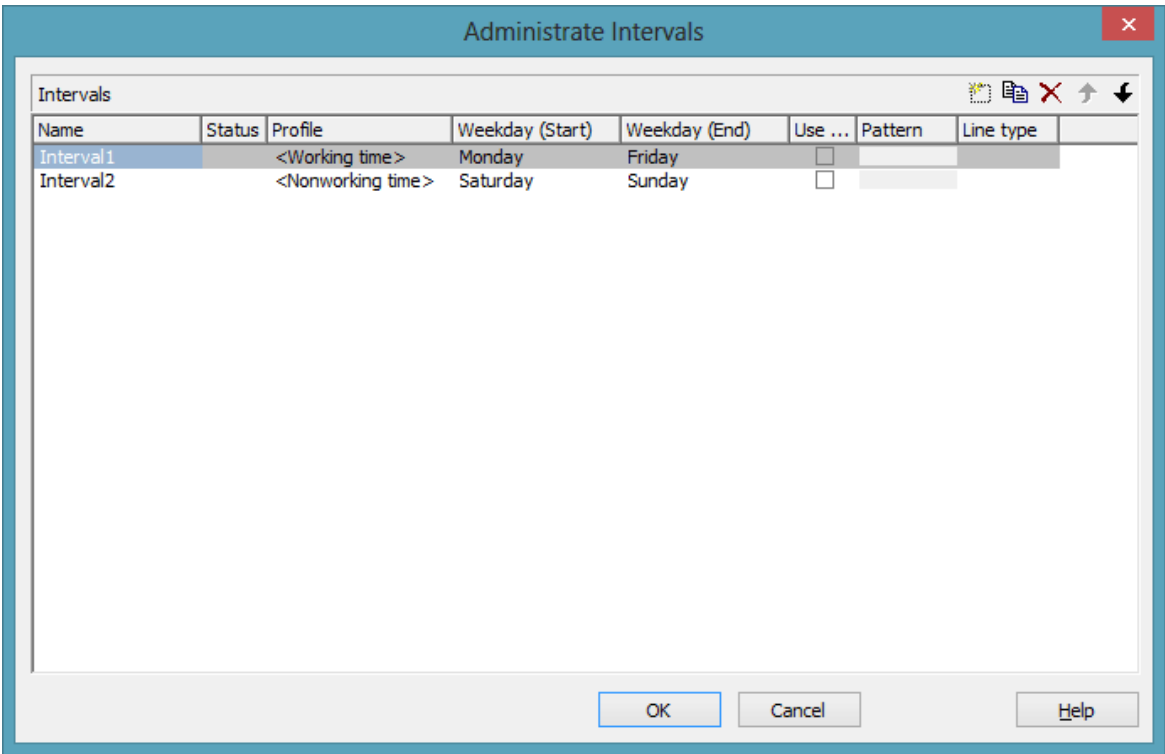
Copy interval

 Click on this button to copy the marked interval.

Delete interval


 Click on this button to delete the marked interval.

4.40 The "Administrate Intervals" Dialog Box (Calendar Profiles, Profile Type <Week Profile>)




You can get to this dialog if you activate the dialog box "Administrate Calendar Profiles" on the "Objects" property page, and then click on the "Edit" button of the calendar profile. The different types of profiles offer different setting options. This dialog serves to create and modify intervals of a week profile.

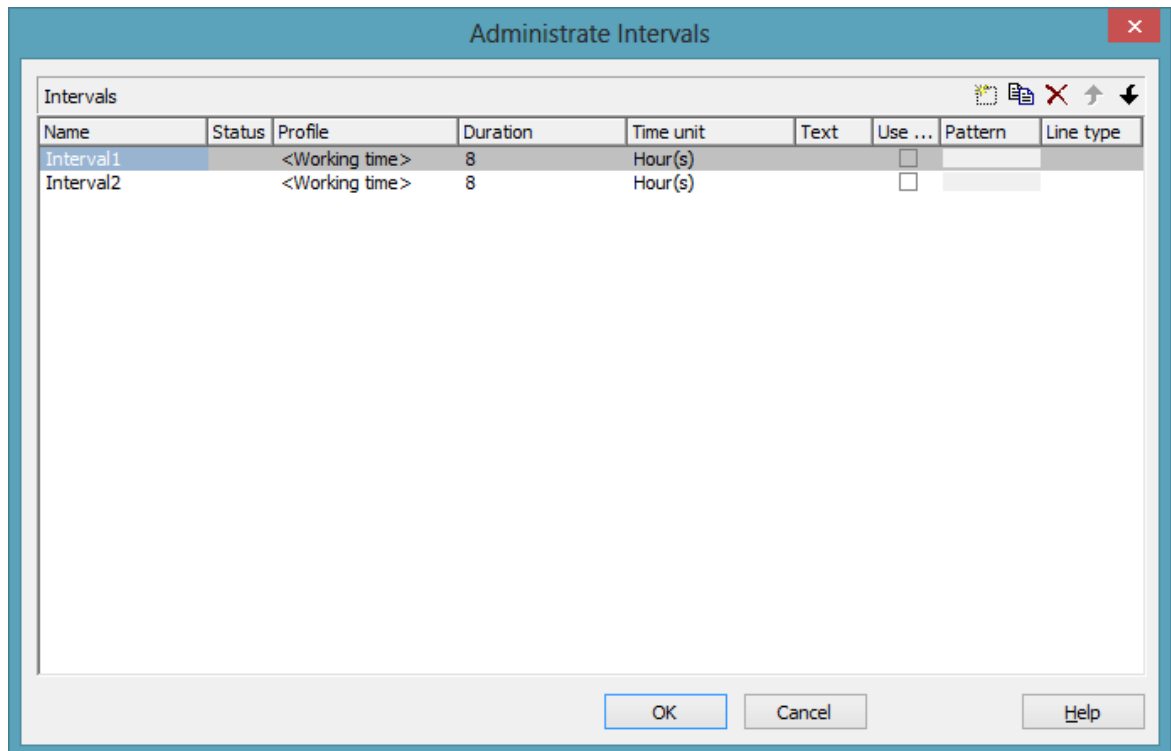
Weekday Start/Weekday End

By clicking  you can set the first/last weekday of the interval.

Weekday Start/Weekday End

By clicking  you can set the first/last weekday of the interval.

4.41 The "Administrate Intervals" Dialog Box (Calendar Profiles, Profile Type <Variable Profile>)



You can get to this dialog if you activate the dialog box "Administrate Calendar Profiles" on the "Objects" property page, and then click on the "Edit" button of the calendar profile. The different types of profiles offer different setting options. This dialog serves to create and modify intervals of a variable profile.

Duration

Here you can specify the duration of the interval. This feature can also be set by the property **VcInterval.Duration**

Time unit

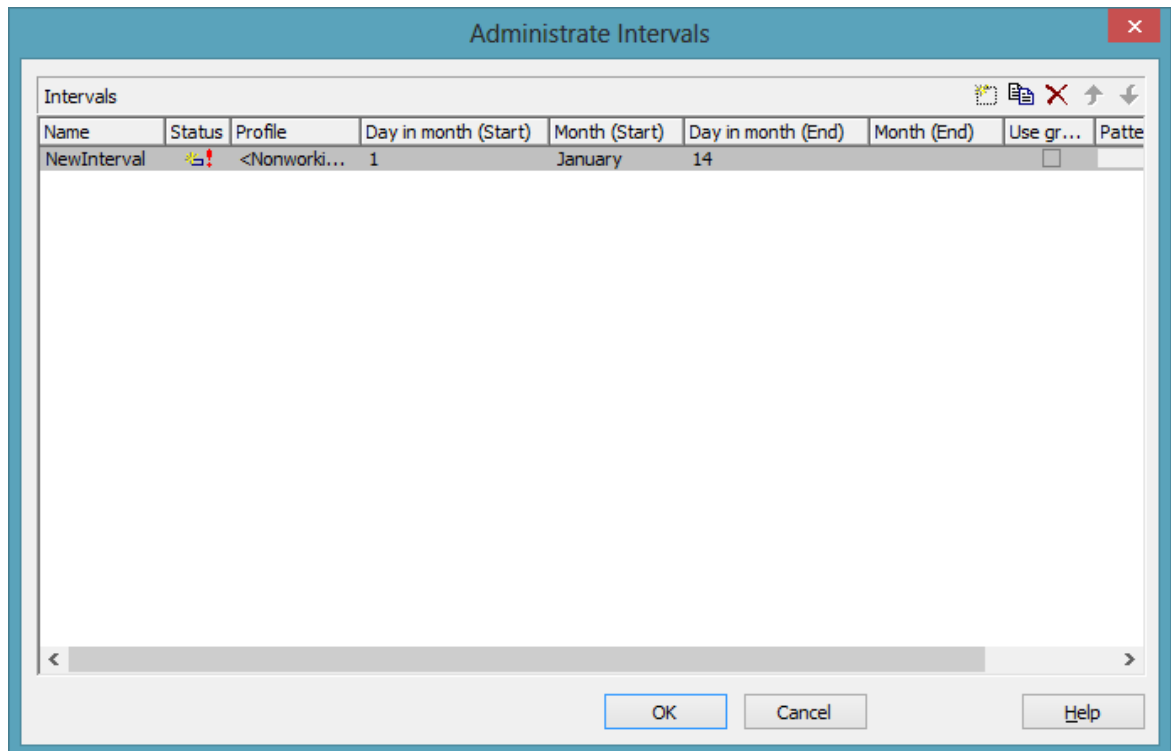
Here you can specify the time unit of the interval. This feature can also be set by the property **VcInterval.TimeUnit**

344 The "Administrate Intervals" Dialog Box (Calendar Profiles, Profile Type <Variable Profile>)

Text


Here you can specify the text of the time ribbon This feature can also be set by the property **VcInterval.Text**

4.42 The "Administrate Intervals" Dialog Box (Calendar Profiles, Profile Type <Year Profile>)




You can get to this dialog if you activate the dialog box "Administrate Calendar Profiles" on the "Objects" property page, and then click on the "Edit" button of the calendar profile. The different types of profiles offer different setting options. This dialog serves to create and modify intervals of a year profile.

Day in month (Start)/Day in month (End)


By clicking  you can set the day in the start/end month of the interval. This feature can also be set by the property **VcInterval.DayInStart/EndMonth**

Month (Start)/Month (End)

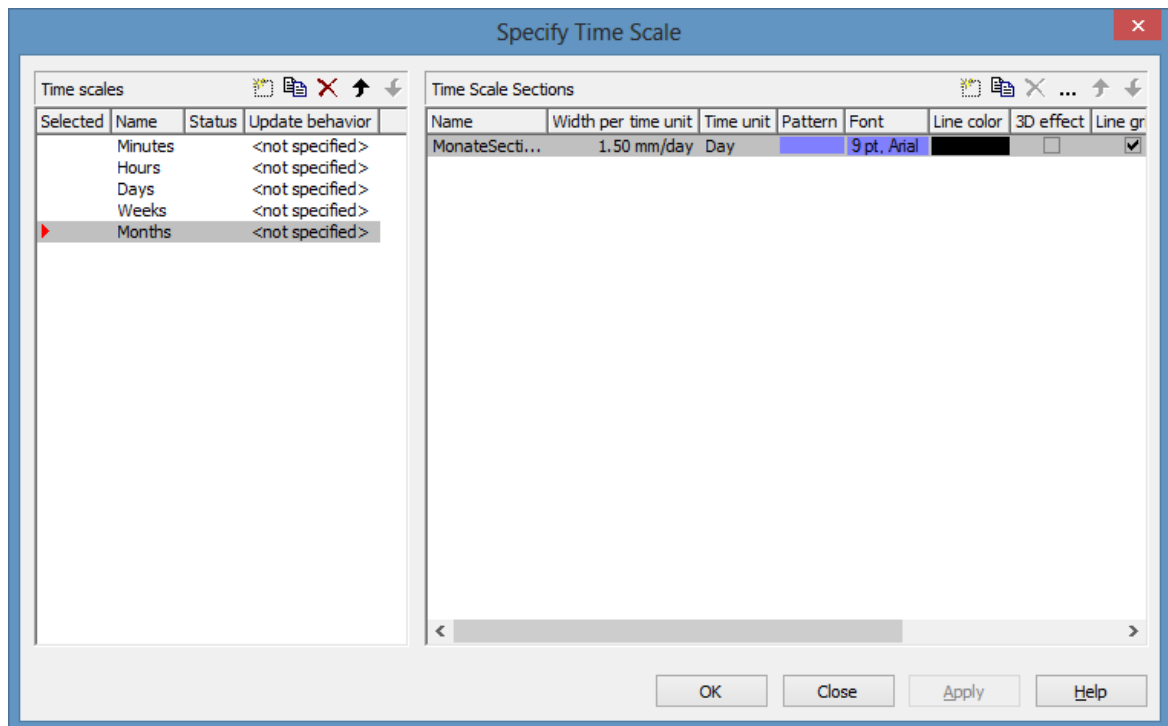
By clicking  you can set the day in the start/end month of the interval. This feature can also be set by the property **VcInterval.Start/EndMonth**

346 The "Administrate Intervals" Dialog Box (Calendar Profiles, Profile Type <Year Profile>)

Month (Start)/Month (End)



By clicking  you can set the day in the start/end month of the interval. This feature can also be set by the property **VcInterval.Start/EndMonth**

4.43 The "Specify Time Scale" Dialog Box








You can reach this dialog box via the **Objects** property page. It allows to establish and modify time scales.

Time scales




- **Selected:** The time scale marked by a small arrowhead in this column is used for the diagram. Please note that the time scale selected here should match the **Time unit** selected on the **General** property page.
- **Update behavior** Select an update behavior for this time scale. Leaving the setting to <not selected> means that the setting for time scales made in the **Edit Update behavior** dialog will apply
- **Name:** Lists the names of all time scales that are defined. The names can be edited.
- **Status:** In this column each time scale that has been added () and/or modified () since the dialog box was opened is marked by a symbol.

Add / copy / delete / edit / promote / demote time scale

     By these buttons you can create, copy or delete time scales or move them by one position up or down in the list, respectively.

Time Scale Sections

The **Sections** table contains all sections specified for the selected time scale. The following properties can be specified:

- **Name** of the section
- **Width per Unit:** Specify the unit width of the active time scale. The basic unit is the smallest unit into which the time scale is divided. You can specify the basic unit width in millimetres in steps of 100th of a millimetre. The maximum width you can assign to the basic unit is 320 mm, the minimum width is 0.01 mm.
- **Unit** of the section: seconds, minutes, hours, days.
- **Pattern:** Click on  to open the **Edit pattern attributes** dialog where you can specify another pattern for the section. If the ribbons had different patterns before, the new pattern will be applied to all sections.
- **Font:** Select the font for the annotation in the section. When you click the first button () , the Color Picker box will appear where you can choose the font color. When you click the second button () , the Windows **Font** dialog box will appear where you can choose the font type. If the ribbons had different fonts (colors or types), the font selected here will be applied to all sections.
- **Line color:** Select a frame color for the time scale.
- **3D-Effect:** This box lets you decide whether the time scale should be assigned a 3D effect (to give it perspective).
- **Line grids:** Specify whether predefined vertical grid lines should be displayed in the diagram area beneath the current section or not.
- **Calendar grids:** Specify whether a predefined calendar grid should be displayed in the diagram area beneath the current section. If you choose to display a calendar grid, weekends and other workfree periods, for example, will be highlighted by vertical areas.
- **Collapse Workfree Periods:** If you select this option, workfree periods will not be displayed in this section. The calendar that defines the workfree periods is selected in the **Specify Calendars** dialog box.

Add/ Copy/ Delete/ Edit/ Promote/Demote time scale section

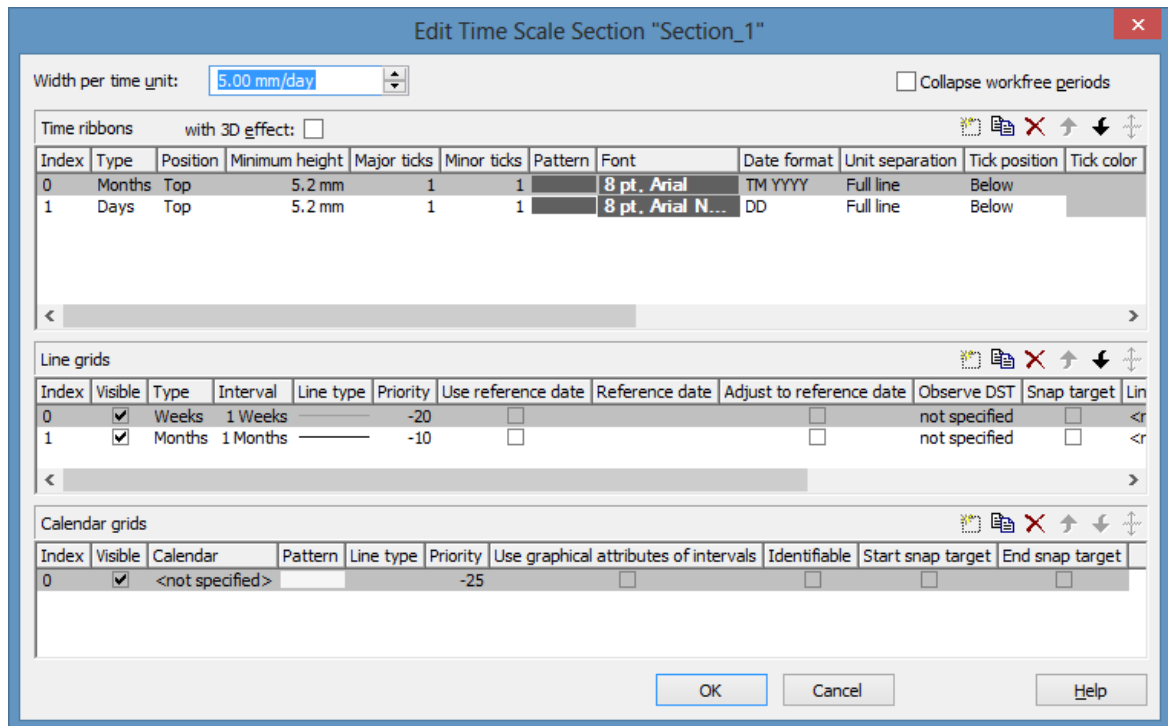


By these buttons you can create, copy, delete or edit time scales or move them in the table, respectively.

The position of the time scale sections in the table corresponds with their position in the diagram.

When creating a new section, all sections will be displayed with nearly the same extent. You can modify their size by using the mouse. The start dates of sections can be set and modified by API calls.

4.44 The "Edit Time Scale Section" Dialog Box



Width per time unit

Specify the width allocated to each unit of the selected section. The new value is transferred to the corresponding field in the **Time Scale Sections** table in the **Specify Time Scale** dialog.

Collapse workfree periods

Here you can set whether or not workfree periods should be displayed in this section. The calendar that defines the workfree periods is the one selected in the **Specify Calendars** dialog box.

When setting this feature, the new value will be copied to the corresponding field in the **Time Scale Sections** table in the **Specify Time Scale** dialog.




Time ribbons

Ribbons serve the purpose of annotating the time scale. A section may have several ribbons (e.g. one showing a monthly and a second one showing a daily scale).



By these buttons you can create, copy and delete ribbons and move them in the table.

The table lets you modify the settings of the ribbons in the selected section:

- **Index:** Displays the serial number of a ribbon (cannot be edited).
- **Type** Lets you set the type of ribbon: seconds, minutes, hours, days, weeks, months, quarters, years, shifts, fiscal quarters, fiscal years.
- **Position:** Lets you specify, whether the ribbon should be displayed at all and if so, whether its position should be at the top or at the bottom of the diagram.
- **Minimum height** Allows to set the minimum height of the ribbon (in mm).
- **Major ticks:** You can set after how many time units a major tick should be displayed, for example after 7 days. (The time unit depends on the ribbon type selected.) The major ticks will be annotated, if sufficient space is available.
- **Minor ticks:** Allows to set after how many time units a minor tick (not annotated) should be displayed, e.g. after one day. The time unit depends on the ribbon type selected.
- **Pattern:** Lets you set the pattern of the ribbon. Click on  to open the **Edit pattern attributes** dialog where you can select a pattern, a color and a second pattern color. If you don't select a new pattern, all ribbons of the time scale section have the pattern specified in the **Specify Time Scale** dialog. If you assign a new pattern to the first ribbon of a section it will be copied to the **Pattern** field in the **Time Scale Sections** table in the **Specify Time Scale** dialog.
- **Font:** Lets you set font specifications to the annotation of the ribbons. If this value is not set, the ribbons of the section will display the font set in the **Specify Time Scale** dialog. To assign a different font color to a ribbon, please click on the drop-down-button () in the ribbon field to get to the color picker. To assign a different font type to a ribbon, please click on the **edit** button () of the ribbon field to get to the Windows **Font** dialog box. The font that you define for the first ribbon of a section will be copied to the **Font** field of the **Sections** table in the **Specify Time Scale** dialog.
- **Date format:** Lets you set the date format to the ribbon. The available formats depend on the selected type of ribbon. To compose the date you can use the following tokens:

- D: first letter of the day of the week (not adjustable)
- TD: Day of the Week (adjustable by using the event **VcTextEntrySupplying**)
- DD: two-digit figure for the day of the month: 01-31
- DDD: first three letters of the day of the week (not adjustable)
- M: first letter of the name of the month (not adjustable)
- TM: name of the month (adjustable by using the event **VcTextEntrySupplying**)
- MM: two-digit figure for the month: 01-12
- MMM: first three letters of the name of the month (not adjustable)
- YY: two-digit figure for the year
- YYYY: four-digit figure for the year
- WW: two-digit figure for the number of the calendar week: 01-53
- TW: text for "calendar week" (adjustable by using the event **VcTextEntrySupplying**)
- Q: one-digit figure for the quarter: 1-4
- TQ: name of quarter (adjustable by using the event **VcTextEntrySupplying**)
- hh: two-digit figure for the hour in 24 hours format: 00-23
- HH: two-digit figure for the hour in 12 hours format: 01-12
- Th: Text of "o' clock" (adjustable by using the event **VcTextEntrySupplying**)
- TH: "am" or "pm" (adjustable by using the event **VcTextEntrySupplying**)
- mm: two-digit figure for the minute: 00-59
- ss: two-digit figure for the second: 00-59
- TS: short date format, as defined in the regional settings of the windows control panel
- TL: long date format, as defined in the regional settings of the windows control panel
- TT: time format, as defined in the regional settings of the windows control panel
- xC/XC: You can set a maximum ten-place, simple upward counting from a

reference date onward, for example "15:05:07:16:00", which equals 15 months, 5 days, 7 hours, 16 minutes, 0 seconds. The notation is: **xC44:C33:C22:C11:C00**. In written language: Show at least 2 digits for the counters 4...0 and a preceding "-" symbol if the value is negative. The separators are variable and can be replaced by other separators symbols. "x" means: Display a preceding "-" symbol if the value is negative, but no "+" symbol if it is positive. "X" means: Display a preceding "-" symbol if the value is negative and a "+" symbol for positive values. In the dialog **Edit Time Scale Section...** the check boxes **Use reference date** and **Adjust major ticks to reference date** need to be ticked, also, the parameter **Serial annotation** has to be set to **No**. In the application the reference date is set at run time by the call **VcRibbon.set ReferenceDate**, overriding any settings in the dialog.

Note: Characters which are not to be interpreted as part of the date should be preceded by a backslash '\'. '\\' for instance results in '\'. The special characters: ':', '/', '-' and **blank** don't need '\' as prefix.

- **Unit separation:** You can choose between three options for the separating lines in the ribbon: **straight lines**, **ticks** and **no lines**.
- **Tick position:** Decide whether the ticks and their annotations should be displayed at the top or at the bottom of the ribbon.
- **Tick color:** You can select the color of ticks.
- **Alignment:** You can choose between **centered**, **right**, **left** and **at ticks** for the alignment of the ribbon annotation.
- **Serial annotation:** Lets you specify whether serial numbers are to be displayed in the ribbon instead of dates, and if so, whether null should be the origin at the reference date possibly set.
- **Use reference date:** Activate this check box if the start value of the serial annotation (or of the fiscal year or quarter) should coincide with the reference date selected. Otherwise it will be placed onto the beginning of the section.
- **Reference date:** Select the reference date from the date picker.
- **Adjust to Reference date:** Tick this check box to position the line grid on a different value of the time unit, i.e. on the one defined by the reference date, for example on 13:17 of a day. If this option is not selected, the lines of a line grid will be positioned on the beginning of a time unit, for example on 00:00 h of a day.

- **Calendar:** If you want to display a shift ribbon, select one of the shift calendars created in the **Specify Calendars** dialog box.
- **Observe DST:** Tick this check box if daylight saving time is to be considered for this ribbon.

Line grids

In the diagram area and in the histogram, one or more line grids can be displayed below the selected section of the time scale.



By these buttons you can create, copy and delete line grids and move them in the table.

The table lets you modify the settings of the line grids in the selected section:

- **Index:** Displays the serial number of a line grid (cannot be edited).
- **Visible:** Activate this check box for the line grids to be displayed.
- **Type:** Lets you set the basic unit of the line grid, e.g. days, weeks, etc.
- **Interval:** Lets you set the size of the interval between the grid lines as an integer multiple of the basic unit of the grid.
- **Line type:** When clicking on the button in this field, the **Line attributes of line grid** dialog box will appear, where you can set shape and color of the borderlines of the line grid.
- **Priority:** Lets you set the priority of a line grid. It refers to other line grids and to layers (> 0 : in front of the layers, < 0 : behind the layers).
- **Reference Date:** The reference date shifts the beginning of the line grid away from the default start on Monday 0:00 h by the offset specified.
- **Observe DST:** Tick this check box if daylight saving time is to be considered for this line grid
- **Snap target:** The line grid defines its relevant positions as "snap targets" for nodes/layers to be moved.



Calendar grids

Calendar grids can be displayed in the diagram area and in the histogram of this section. If you choose to display a calendar grid, workfree periods will be highlighted by vertical areas.

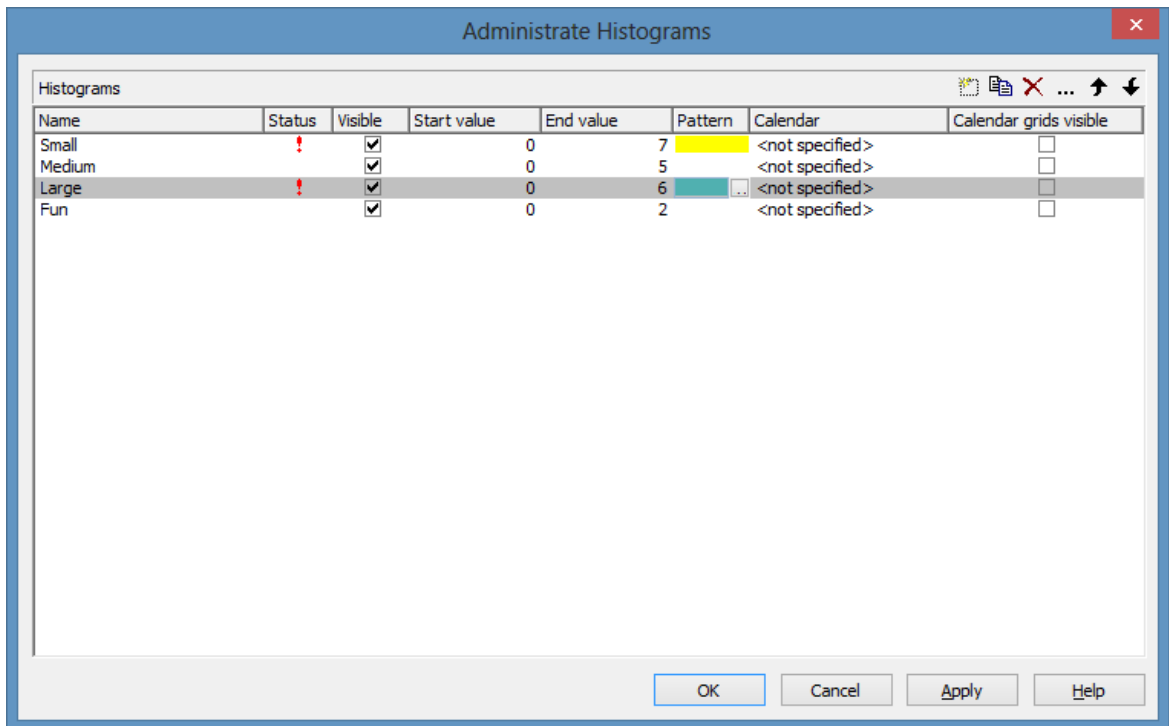


By these buttons you can create, copy and delete calendar grids and move them in the table.

The table lets you modify the settings of the calendar grids:

- **Index:** Displays the serial number of a calendar grid (cannot be edited).
- **Visible:** Activate this check box for the calendar grids to be displayed.
- **Calendar:** Select the calendar that specifies the workfree periods displayed by the calendar grid. If you select the entry <not specified>, the calendar selected in the **Specify Calendars** dialog box will be used.
- **Pattern:** When clicking on this button (), the **Pattern attributes** dialog box will appear, where you can set the type, the foreground and the background color of the pattern for the calendar grid. There are also transparent colors available.
- **Line type:** When clicking on this button (), the **Line attributes of calendar grid** dialog box will appear, where you can enter the settings of the border lines of the calendar grid.
- **Priority:** Lets you set the priority of a calendar grid. It refers to other calendar grids and to layers (> 0: in front of the layers, < 0: behind the layers)
- **Calendar grid:** The calendar grid defines its relevant positions as "snap targets" for nodes/layers to be moved.

4.45 The "Administrate Histograms" Dialog Box





You can get to this dialog box via the **Layout** property page.

You can establish and modify histograms and select which ones are to be displayed.

Name

Lists the names of all histograms that are defined. The names can be edited.

Status

In the **Status** column each histogram that has been added () and/or modified () since the dialog box was opened is marked by a symbol.

Visible

Tick this box if you want the selected histogram to be displayed.

Start value

Specify the smallest value of the numeric scale of the histogram. If necessary, this value will be adapted to the curve values.

End value

Specify the greatest value of the numeric scale of the histogram. If necessary, this value will be adapted to the curve values.

Pattern

Specify pattern und color for the histogram.

Add histogram



A new histogram is created.

Copy histogram



Copies the selected histogram.

Delete histogram



The marked histogram is deleted.

Edit histogram



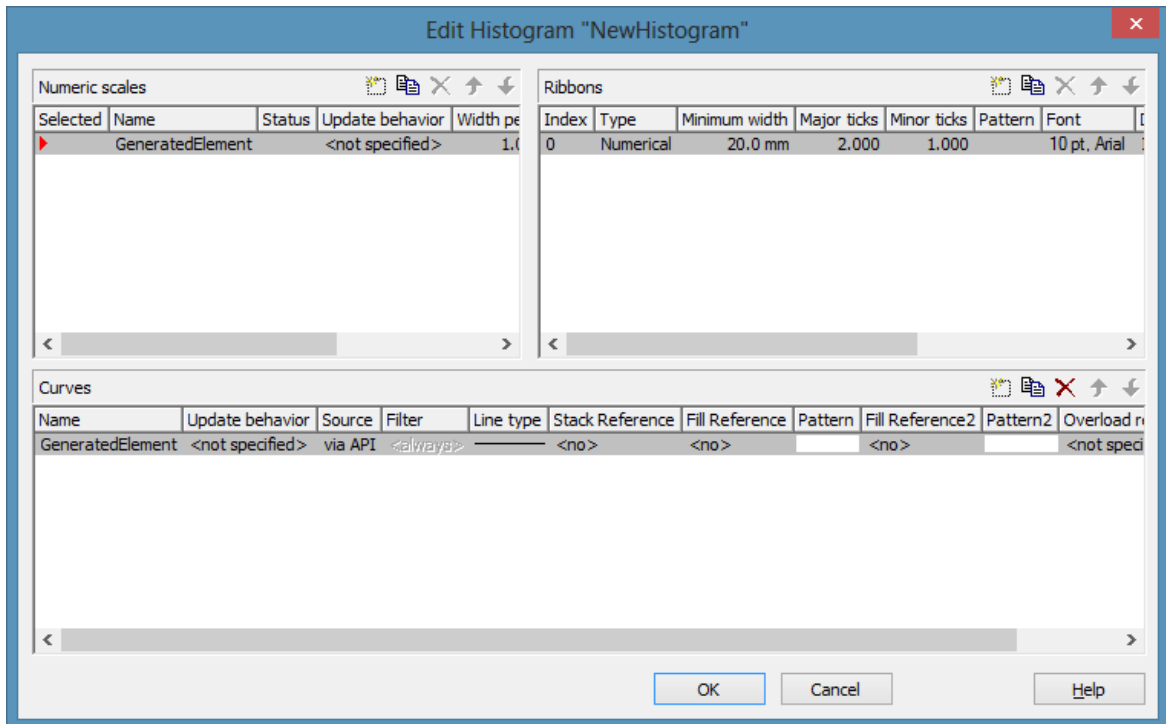
The **Edit Histogram** dialog box will appear.

Promote / demote histogram



By these buttons you can create, copy or delete the histogram or move it by one position up or down in the list, respectively. The order of the histograms in the list equals their order of output.

4.46 The "Edit Histogram" Dialog Box



This dialog box will appear if in the **Administrate Histograms** dialog box the **Edit histogram** button () is clicked.

For the histogram being edited you can establish several numeric scales that contain one or more ribbon(s), and select the numeric scale to be displayed.

The histogram may contain several curves.

For each curve you can individually define the source by which its data are to be supplied. Via filters you can select specific activities to compose the curve. Beside, you can define the appearance of the curves.

Numeric Scales

- **Selected:** The red arrow indicates which one of the numeric scales is displayed.
- **Name:** of the numeric scale
- **Status:** In this column each numeric scale that was added () and/or modified () after opening the dialog box is marked by a symbol.
- **Width per Unit** in mm, specifies the space between the major ticks
- **Update behavior** Select an update behavior for this numeric scale. Leaving the setting to <not selected> means that the numeric scale setting made in the **Edit Update behavior** dialog will apply

- **Unit** specifies the increment of the major ticks
- **Line color** Specify the tick color for all numeric ribbons
- **Line Grids:** Specify whether a line grid is to be displayed.
- **Line type:** The line type of the line grid is displayed here. To change it, click on the button (...). Then the **Line Attributes** dialog box will open.

Ribbons

For each ribbon of the marked numeric scale you can set the below properties:


- **Index:** consecutive number of the ribbon (cannot be edited)
- **Type** of the ribbon (numerical or textual). By the button you open a dialog to specify the type.
- **Minimal width** minimum width in mm
- **Major ticks:** Enter the number of units after which a major tick including an annotation is to occur.
- **Minor ticks:** Enter the number of units after which a minor tick (smaller tick without annotation) is to occur.
- **Pattern:** By clicking on (...) you open the **Edit pattern attributes** dialog where you can specify a pattern, a pattern color and background color or, if needed, a second pattern color. You can define your own colors in addition to the ones suggested. Transparent colors are also available.
- **Font:** The font style and color of the ribbon are indicated. Click on the button (...) to get to the Windows **Font** dialog box.
- **Double format:** Here you can choose from a list of possible double output formats. **I** represents the figures before the decimal separator and **D** represents the figures after the decimal separator.
- **Tick color:** Specify the tick color for all numeric ribbons.
- **Object draw events:** Tick this option if you want to enable the events **VcObjectDrawing** and **VcObjectDrawn**. The event **VcObjectDrawing** lets you replace the default annotation ribbon by a customer-defined one, and with the event **VcObjectDrawn** you can add something to the annotation ribbon that was drawn by VARCHART XGantt,
- **Unit label:** annotation of the label units of the numeric scale.


Curves


- **Name:** In this column, the names of the curves available are listed.
- **Update behavior** Select an update behavior for this curve. Leaving the setting to <not selected> means that the setting for curves made in the **Edit Update behavior** dialog will apply
- **Source:** By defining the source, you can specify where the data for calculating a curve are to be taken from. You can choose between two basic alternatives:

1. by Layer: The curves are generated from the data of layers of those activities, that fulfill the filter criteria. With the help of a filter these activities can be specified further.

2. by API: By this option, the values are set by the API. In the API, the values for a histogram curve can be freely defined using the VcCurve method **SetValues**. A curve defined this way is independent of user interactions and therefore can be used, say, as a reference curve, to display the availability, for example.

By the **Edit** button () you can open the **Select curve data source** dialog box.

- **Filter:** If desired, specify a filter for each curve to specify the activities that contribute to the curve. By the **Edit** button () you can open the **Administrative Filters** dialog box.
- **Line type:** Click on the **Linetype** entry to open the **Line attributes** dialog box.
- **Stack Reference:** To stack histogram curves, for each curve, in the **Stack Reference** field specify the curve on which you want the current curve to be stacked. If you do not want to stack a particular curve, select the entry <No> for that curve blank. If you select the entry <No> in the **Stack Reference** field for all curves, they will not be stacked, but will overlap each other instead. In order to still be able to differentiate between the curves, assign them different patterns.
- **Fill Reference:** This field allows you to specify how far down the fill pattern below a curve should reach. If you select <No> in the **Fill Reference** field for a particular curve, there will be no fill pattern beneath this curve. If you enter <Flatline>, the fill pattern will reach down to the flatline. By specifying a curve in the **Fill Reference** field, the fill pattern will fill the area down to the curve.

- **Pattern:** Specify the pattern below a curve. By the **Edit** button () you can open the **Pattern** dialog box to set the pattern.
- **Fill Reference 2:** Select the second reference curve. The filling below the second reference curve is displayed only if the y values of the current curve (the curve defined in this row) exceed the y values of the second reference curve.
- **Overload results calendar:** Select a calendar created by you for this purpose to store the intervals that have been calculated by the overload dates. You could this calendar, for instance, for a calendar grid in a group.

Pattern 2: Set the pattern and the color of the filling above the second reference curve.

In the tutorial you can find examples for the usage of histograms in the chapters "Using histograms" and "Displaying Capacity Bottlenecks".

Overload results calendar: Select a calendar you have created before for this purpose to store the intervals having been calculated by the overload dates to. This calendar could be used, for instance, to display a calendar grid in the group.

Add numeric scale/ribbon/curve



A new object is created.

Copy numeric scale/ribbon/curve



Copies the selected object.

Delete numeric scale/ribbon/curve



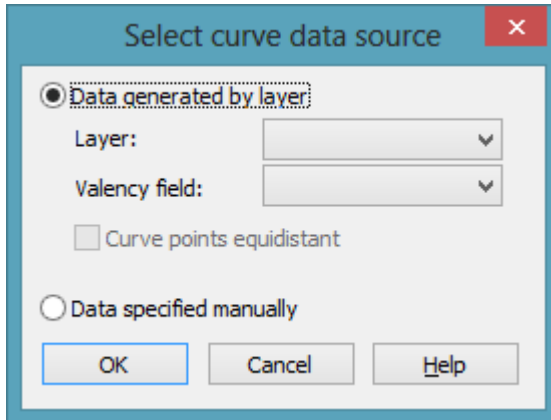
The selected object is deleted.

Promote/demote numeric scale/ribbon/curve



By these buttons you can move the selected object by one position up or down in the list, respectively.

4.47 The "Select Curve Data Source" Dialog Box



You can get to this dialog via the **Edit Histogram** dialog.

Data generated by layer

Select this option, if you want the data to be generated by layer. When the activities are summarised to a curve, the start and end dates of the selected layer type (e.g. the "Start-End" layer) of each activity are adopted.

Then specify the following:

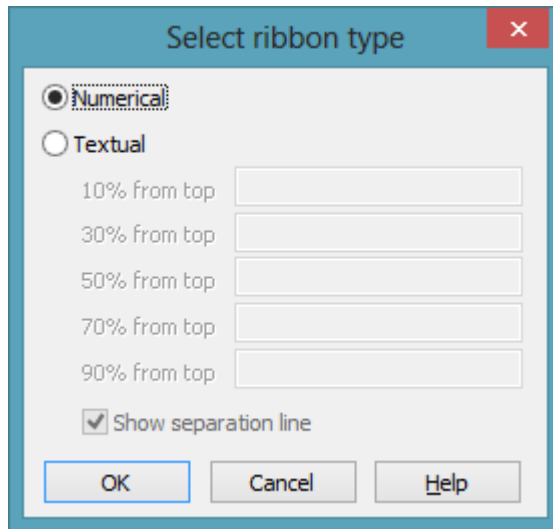
- **Layer**
- **Valency field:** data field from which for each activity the valency for the capacity sum is to be taken.

Data specified manually

Select this option, if the data are to be specified manually. For this option you may choose the option **Curve points equidistant**. Otherwise the curve points will be created only in those points where the y values are changing.

For further information please see the chapter "Important Concepts: Histograms".

4.48 The "Select Ribbon Type" Dialog Box



You can get to this dialog via the **Edit Histogram** dialog.

Numerical

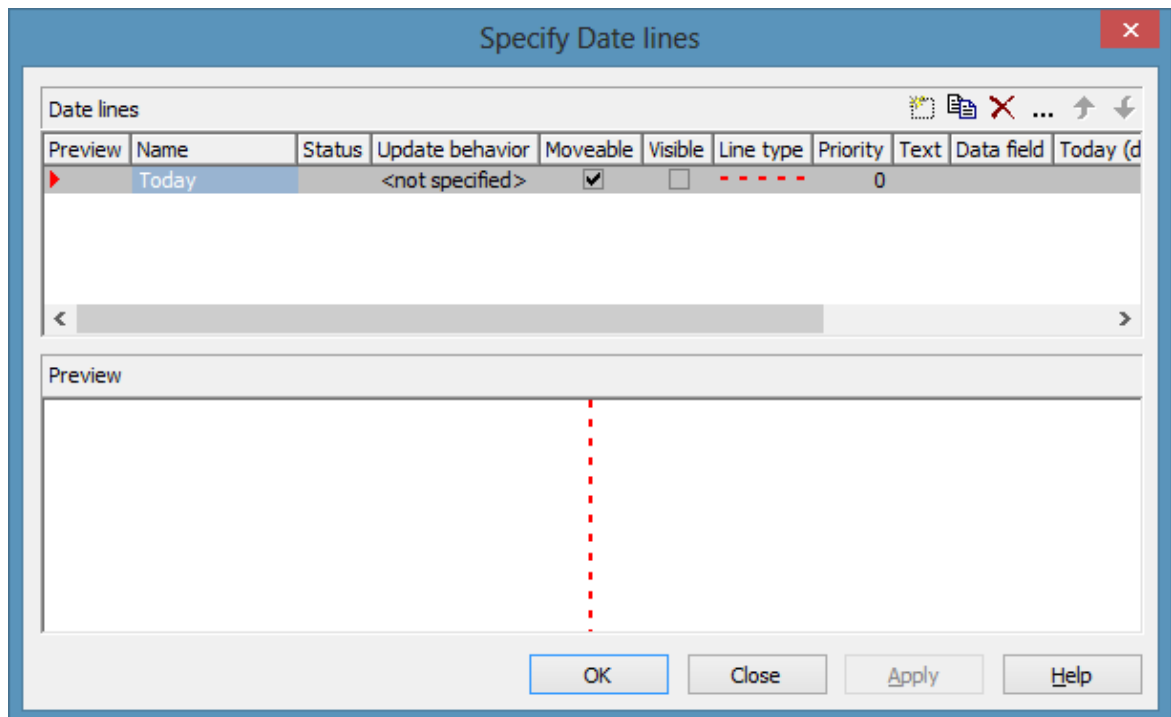
Select this option if the current ribbon of the numeric scale is to be annotated with numbers.

Textual

Select this option if the current ribbon of the numeric scale is to be annotated with texts which you can specify for five predefined positions (10%, 30%, 50%, 70 %, 90 % from top).

After having defined more than one ribbons in the dialog **Edit histograms** you can specify whether to draw a vertical separation line on the right of the corresponding ribbon by clicking **Separation line**.

4.49 The "Specify Date Lines" Dialog Box



Date lines (vertical lines in the diagram) let you highlight specific dates (the actual date or any other date) in your diagram. This dialog box allows to create or delete date lines in your chart and to set options to them. You can invoke this dialog on the **Objects** property page.



Preview

The date line marked by a small red arrowhead is displayed in the preview window.

Name

Lists the names of all date lines that are displayed in the chart. The names can be edited.

Status

In this column date lines that were added () or modified () after the dialog box was was invoked are marked by a symbol.

Update behavior

Select an update behavior for this date line. Leaving the setting to <not selected> means that the setting for date lines made in the **Edit Update behavior** dialog will apply

Moveable

Activate this check box, if you want the date line to be interactively moveable at run time.

Visible

Activate this check box, if the date line should be visible at runtime.

Priority

Specify the priority of the date line (> 0: on top of of layers, < 0: behind layers).

Text


You can enter a text to be displayed at the date line.

Today (dynamically)

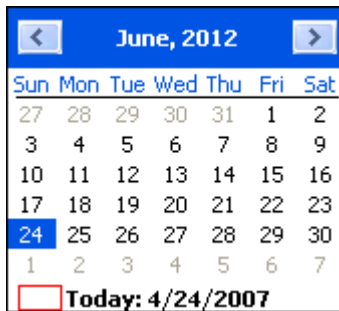
Tick this check box, if on the start of the program the date line should indicate the system date and time. In this case, the **Date** field will be deactivated.

Date

You can modify the date of the date line by marking a section of the date and then selecting a new value by the arrow keys.

Alternatively, you can set the date by the date control. For this, please click on the arrow button (). The **date** dialog box will appear where the selected date is highlighted. If no date was selected, the current date is highlighted. Select a day from the month displayed. You can flip through the months by clicking on the arrow buttons at the top of the calendar. If you click on the name of a month, a select box will appear which lists the names of all months. If you click on the year, a set of arrow buttons will appear by which

you can move to the next or to the previous year. If you click on **Today**, the current date will be selected.



Date

Tick this check box if you want the date line to be identified by the VcGantt method **IdentifyObjectAt**.

This option can also be set by the **VcDateLine.Identifiable** property.

Snap target

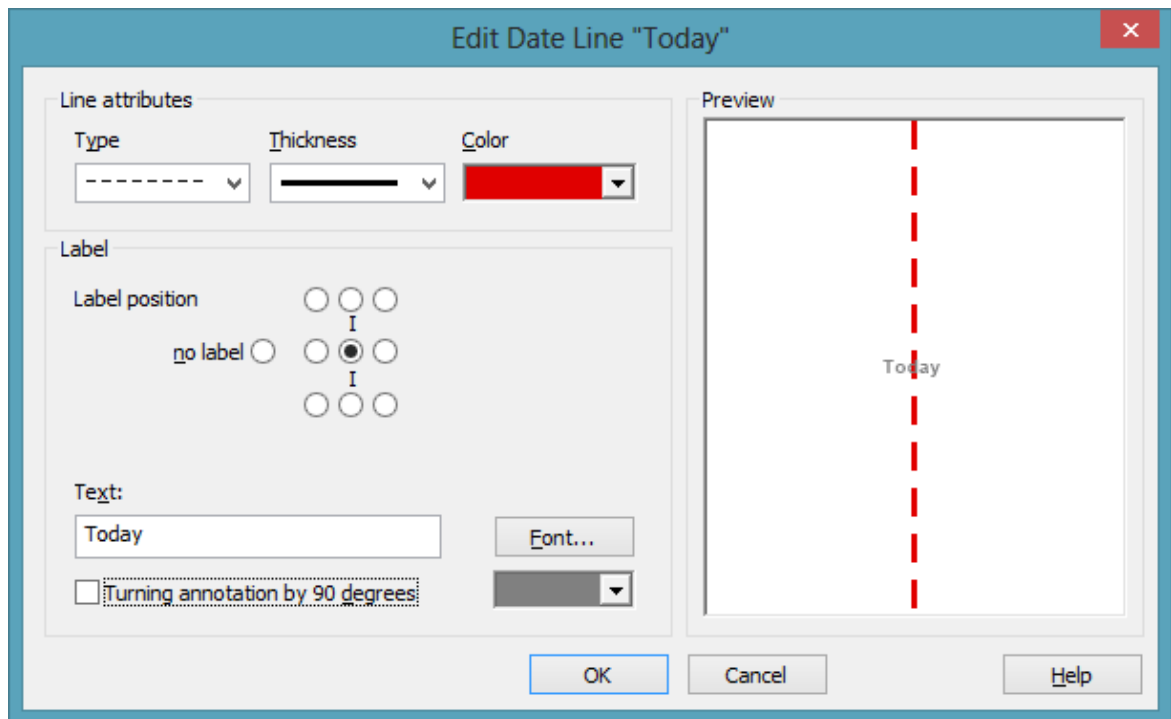
Specify whether the date defines its position (hence its date) as "snap target" for nodes/layers to be moved.

Add / copy / delete / edit / promote / demote date line



By these buttons you can create, copy or delete a date line or move it by one position upward or downward in the list.

4.50 The "Edit Date Line" Dialog Box



Line attributes

Specify the **Type**, **Thickness** and **Color** of the date line.

Label position

Select the position at which a text should be displayed at the date line. If you do not want to display a text, tick the **no label** radio button. It is ticked by default, if no text is specified for the date line. If you specify a text for the date line and then leave the **Text** field, by default the text is displayed at the top right of the line. You can choose a different position for the text, if you want.

Text

Specify the text you want to display at the date line. By default the **Text** field is empty. When you select a text position at the date line the name of the line is transferred to the **Text** field. You can modify the text, if you wish.

Font

This button lets you get to the Windows dialog box **Font** where you can specify the font for the text at the date line. By the button below, you can get to the Windows color picker, that lets you select a color for the text font of the date line or create a new color.

Rotating an annotation by 90 degrees

Activate this check box, if the annotation should be displayed in vertical direction.

4.51 The "Specification of Texts, Graphics and Legend" Dialog Box

You can get to this dialog box if you click in the **Border Area** property page on one of the nine buttons above/below the drawing.

Type of contents

Specify the type of information you want to display at the chosen position:

Empty: If you do not want to output anything at the chosen location, click on this flag.

Text: The text of the six text lines will be displayed at the chosen location.

Graphics: The graphics file (selected by the **Browse** button) will be displayed at the chosen location. Graphics are always positioned in the center.

Legend: A legend will be displayed at the chosen location. It describes the layers used in the diagram.

Following your selection, the sections of the dialog box that are not required are deactivated (all entries are maintained).

Legend attributes

*Only activated when the check box **Legend** has been ticked.* You will open the **Legend attributes** dialog box where you can specify more attributes for the legend.

Graphics file

*Only activated if the check box **Graphics** was ticked.* Select the graphics file to be displayed by clicking on the **Browse** button or enter the file name in the field manually. If the selected graphics file is not stored in the installation directory of the VARCHART web server, please also specify the drive and the directory.

Browse

*Only activated if the check box **Graphics** was ticked.* Click on this button to reach the **Choose Graphics File** dialog box and select the drive, the directory and the name of the appropriate graphics file.

Lines of text

*Only activated if the check box **Text** was ticked.* Specify the text (max. 6 lines) you want to display at the chosen diagram position and/or specify substitutes (e.g. &[System date]) to represent project info. If all six lines are empty, the area will not be displayed in the diagram.

Project details

*Only activated if the check box **Text** was ticked.*

Here you can add several project details (number of pages, page number, system date) to your chart by selecting the appropriate place holder from the list and by clicking on the **Add** button.

The place holders will be replaced by the required data and will continuously be kept up-to-date in the print preview and the printout.

Add

*Only activated if the check box **Text** was ticked.* When you have selected a project detail from the list, click on **Add** to confirm your choice. The project detail will be inserted in the line where the cursor is currently positioned.

Alignment of text

*Only activated if the check box **Text** was ticked.* Specify whether the text lines should be output left-aligned, centred or right-aligned.

Font for all lines

*Only activated if the check box **Text** was ticked.* You will reach the **Font** dialog box where you can specify the font attributes for all six lines. If you use this option to specify the font for all lines, the settings for the font for line 1...6 will be overwritten.

Font for line 1...6

*Only activated if the check box **Text** was ticked.* To assign a different font to each of the six lines, click on this button. Depending on the line in which the cursor is currently positioned, the notation of this button will change to 1, 2, 3, 4, 5 or 6. You will reach the **Font** dialog box where you can specify the font attributes for each separate line.

Clear all texts

*Only activated if the check box **Text** was ticked.* Click on this button to delete the contents of all six lines of text.

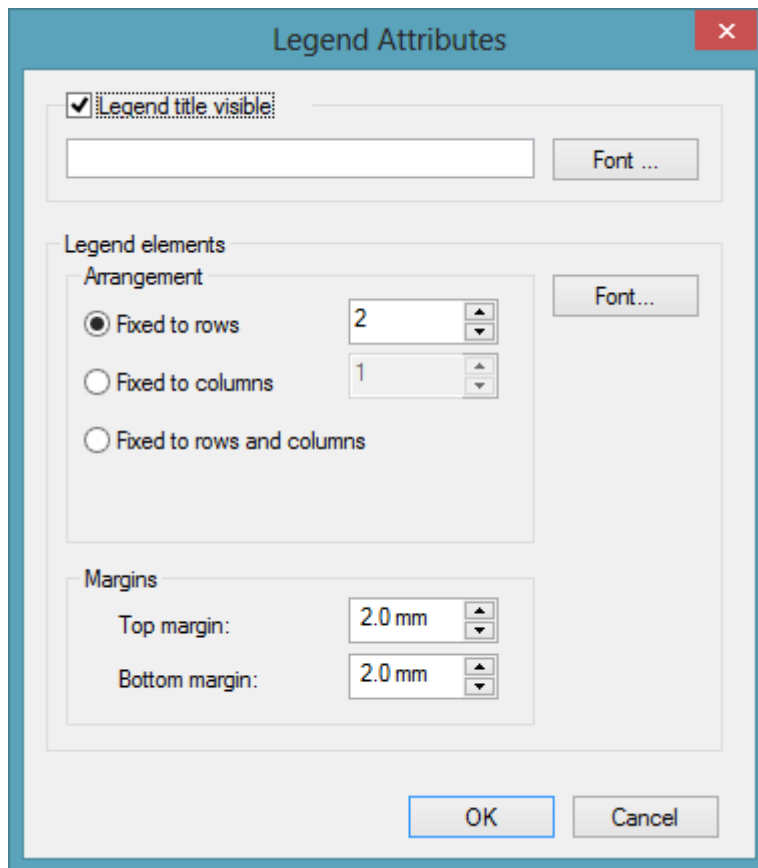
Max. Height (mm)

*Only activated if the check box **Graphics** was ticked.* If you have specified several fields for text, graphics or legend, you can specify the max. height for the current field to prevent field contexts to be cropped.

Max. Width (mm)

*Only activated if the check box **Text** or **Graphics** was ticked.* If you have specified several fields for text, graphics or legend, you can specify the max. width for the current field to prevent field contexts to be cropped.

4.52 The "Legend Attributes Dialog Box"



You can reach this dialog at runtime by clicking the corresponding item of the legend's contextmenu or at designtime clicking the corresponding button in the dialog **Specification of Texts, Graphics and Legend**. The button can only be clicked after having selected **Legend** as **Type of contents**.

Legend title visible

Tick this check box if the legend title shall be displayed and enter a text. By clicking on **Font** you open the corresponding Windows dialog box which lets you specify the font attributes of the legend title.

Arrangement

- Fixed to Rows: Specify the number of rows to be displayed in the legend.
- Fixed to Columns: Specify the number of columns to be displayed in the legend.
- Fixed to Rows andColumns: Specify the number of rows and columns to be displayed in the legend. If the number entered here is lower than the existing layers, the surplus layers are not displayed.

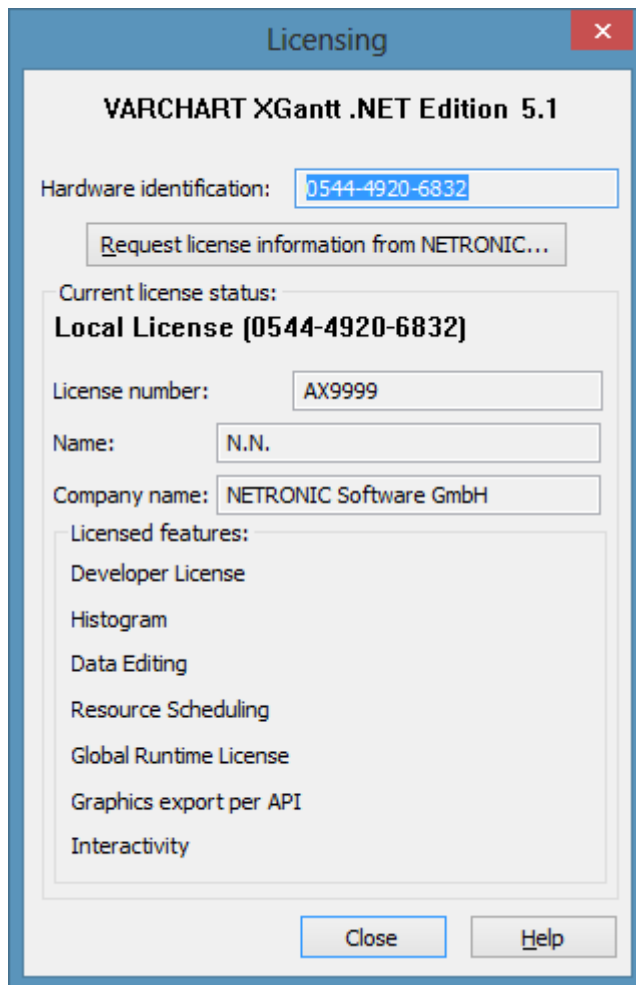
Margins

- Top margin: enter a value for the top margin of the element
- Bottom margin: enter a value for the bottom margin of the element..

Font

By clicking this button you open the Windows **Font** dialog box where you can specify the font attributes for the legend.

4.53 The "Licensing" Dialog Box



You can get to this dialog by the **General** property page.

Before licensing, the program is automatically licensed as a trial version. Compared to the full version, the trial version is subject to restrictions: The trial period for testing the product is limited to 30 days. After this period, all diagrams will show a "Demo" water mark.

Hardware identification

(cannot be edited) The number indicated in this field is calculated from your hardware configuration. It is required by NETRONIC Software GmbH for the licensing procedure. When changing your hardware, you need to renew your license. Please do not hesitate to contact the support team of NETRONIC.

Request license information from NETRONIC

For licensing, click on this button, which will get you to the **Request License Information** dialog.

License number/Name/Company name

(cannot be edited) Indicates your license number, your name and the name of your company.

Licensed features

The modules that have been licensed are indicated. If the licensing procedure was successful, the licensed modules are activated.

- **Developer license**
- **Histogram**
- **Data editing** (provides all functions of editing application data)
- **Resource scheduling** (requires all other modules and provides all functions for resource scheduling)
- **Global runtime license** (In the run time mode, VARCHART XGantt can be run on any computer.)
- **Single-place runtime licenses** (VARCHART XGantt has to be licensed individually for the computer to run on.)
- **Graphics export per API**
- **Interactivity**

Close

Quits the dialog box.

4.54 The "Request License Information" Dialog Box

Request License Information

VARCHART XGantt .NET Edition 5.0

Hardware identification: 0544-4920-6832

First step: Enter your user information below:

License number:

Name:

Company name:

Second step: Request your license information:

If you cannot send emails from your computer,
contact NETRONIC Software GmbH by stating the
four entries above:
email: license@netronic.com
phone: +49/2408/141-0
fax: +49/2408/141-33

Third step: After receiving the license information file, copy
it into the directory of the DLL file.

Enter your license number, your name and the name of your company and click on **Send email to NETRONIC**. An email to NETRONIC will be generated automatically. As soon as we have received it, we will generate your license information file (**NETRONIC.XGantt.VcGantt.lic**) and mail it back to you.

After having received the file, please copy it to the directory in which the file **NETRONIC.XGantt.dll** is stored.

After licensing, you need to activate the new license in each of your projects. So please open a property page in each of your projects, make some change and store it. Then the new license will be activated.

5 User Interface

5.1 Overview

The following list gives an overview of possible user interactions.

- Navigation in the diagram and in the table
- Zooming
- Marking nodes or layers
- Creating nodes
- Moving nodes
- Moving layers
- Change start/end date
- Delete, cut, copy and paste nodes
- Editing node data
- Editing links
- Anchor boxes to nodes
- Editing group data
- Expanding/collapsing groups
- Moving groups
- Modifying table/diagram ratio
- Modifying table column width
- Editing fields in the table
- Inserting table rows
- Editing the timescale
- Modifying the scaling and the frontiers of sections
- Moving the date line
- Editing the legend
- Setting up pages
- Use the print preview

Context menus (right mouse key):

- for the diagram
- for nodes
- for links
- for groups
- for the timescale
- for the histogram
- for the legend
- for boxes

For further information on user interactions in grouped diagrams or in hierarchically sorted diagrams please read the chapters "Important Concepts: Grouping" or "Hierarchy" respectively.

All these interactions trigger an event so that you will be informed about it and will be able to react to it.

5.2 Navigation in the Diagram and in the Table

Use the following keys and shortcuts for navigating in diagram and table:

- The arrow keys move the marking from one node to the other in the selected direction (for further information, in particular concerning the marking in groups, please see chapter 5.4 "Marking Nodes and Layers").
- **Pos1**: scrolling to the left diagram border
- **Ctrl + Pos1**: scrolling to the left upper diagram corner
- **End**: scrolling to the right diagram border
- **Ctrl + End**: scrolling to the right lower diagram corner
- **Page up/down**: scrolling one screen page up/down
- **Ctrl + Shift J**: scroll to the next date line
- **Ctrl + *** (NUM key): the screen section is shifted so that the start of the node is visible

The mouse can also be used for navigating:

- Turn the mouse wheel for scrolling vertically in the diagram or in the histogram (depending on the cursor position)
- By holding down the mouse wheel (or the middle mouse key) and moving the mouse you can scroll in any direction wanted.

5.3 Zooming

The following shortcuts can be used for zooming:

- **Ctrl + Num -**: zoom out
- **Ctrl + Num +**: zoom in

You can also use the mouse for zooming:

- Turn the mouse wheel while holding down the Ctrl key. For that purpose the usage of the mouse wheel for zooming has to be permitted. This can be done by ticking the **AllowZoomingByMouseWheel** box on the **General** property page or by setting the property **VcGantt1.Zooming-PerMouseWheelAllowed** to **True**. This property is set to **False** by default.

For further information about zoom settings for the print output please see chapter 5.21 "Setting up pages".

5.4 Marking Nodes or Layers

To mark a node, click the left mouse key on the node. The first field of the corresponding table line will also be marked.

You can also click on a certain field in the table and with that mark the corresponding activity in the diagram area at the same time.

To mark several nodes which are situated above or below one another in the diagram area, keep the Shift key pressed while clicking on the nodes or on the corresponding table lines in the table area.

Alternatively, you can drag a rectangle around the nodes to be marked, using the left mouse key.

Several nodes which are not situated above or below one another in the diagram area can be marked by keeping the Ctrl key pressed and clicking on the nodes or on the corresponding table lines in the table area.

For groups of the mode **All nodes in one row** and **optimized**: If you navigate downwards, the first node of a group will be marked at first.

If you navigate upwards, the last node of a group will be marked at first.

Within these groups you can use the arrow buttons left/right to navigate to the left/right.

Note: The markings of nodes or table fields/lines are undone by clicking on them a second time or by pressing the ESC-key.

5.5 Creating Nodes

This mode is available only if the **Node creation allowed** option on the **Nodes** property page was activated.

In this mode, the cursor shape turns to a small cross. In this mode you can create a node by drawing a frame by the mouse while keeping the left mouse button pressed. An information window will appear at the current position of the mouse which shows the current start and end date and the duration of the new node.

Create Activity	
Start:	07.09.2007
End:	09.09.2007
Duration:	2 days

If you create a node in a collapsed group in a diagram that has several levels of groups, in addition to the small cross an arrow will appear: It indicates whether the new node will be the first node in the group (arrow up) or the last one (arrow down).

If the cursor is placed in a group title row of an expanded group, the new node will be inserted as the first node,

In hierarchically grouped diagrams you always can insert the new node above or below the reference node (depending on the arrow direction).

If the **Edit new node** option on the **Nodes** property page was activated, the **Edit Data** dialog will appear as soon as you release the mouse button. This dialog lets you edit the data of the new node.

If you have not defined anything else in your settings, a node just created will appear at the position of the mouse.

The **Mode: Create Node** can also be activated by setting the property **InteractionMode** to the value **VcCreateNode**.

The event **VcNodeCreating** is triggered when the user creates a node. The node object is captured, so that a validation can be made. For the validation, the **Edit Data** dialog box needs to be activated. You can delete a node by setting the **returnStatus** to **vcRetStatFalse**.

5.6 Moving Nodes by Mouse

Moving nodes in the diagram

The possibilities of moving a node vary in dependence on the settings on the **Nodes** property page. Find below the description of how to move nodes when the following default settings on the **Nodes** property page are valid (for information about further possible settings please see chapter 4.4 "The Nodes Property Page"):

- All layers moving together
- Move layers as node when shift key pressed allowed

When you position the mouse on a node, the mouse pointer takes the shape of a small square with an arrow pointing left and right (or with four arrows, when the node consists of one layer only). Now you can move the layer by dragging it with the mouse.



If you want to move the complete node (with all layers) press the Shift key while pointing on the node. Now the cursor takes the shape of a small square with four arrows . Hold the Shift key down while dragging the node to a different position. An info box will display the current start and end dates of the node. As soon as you release the mouse key, the node will be dropped at the current position and the box will be closed.

Move Activity	
Start:	01.09.2007
End:	11.09.2007

Note: The Shift key has to be pressed only if you want to move a node that consists of more than one layer.

If the **Moving a node vertically via diagram allowed** box is ticked on the **Nodes** property page, nodes can also be moved in vertical direction.

When a node is being moved vertically in the diagram, a cursor with corresponding arrows indicates in which way the node will be positioned relatively to the other nodes: .

Moving nodes in the table

If the check box **Moving a node vertically via table allowed** has been ticked, you can also move nodes in the table. Up to now, however, it is only possible to move complete nodes only vertically. When a node is being moved vertically in the table, a cursor with corresponding arrows indicates in

which way the node will be positioned relatively to the other nodes:



The event **VcNodeModifying** occurs when the user has modified the length or the position of a node or a value in the **Edit Data** dialog. By the **modificationType** parameter you get further information of the kind of modification. If you set the returnStatus to **vcRetStatFalse**, the modification will be revoked.

5.7 Moving Nodes and Modify Duration by Keys

Usually, the arrow keys <left> and <right> are reserved for various navigating interactions, such as scrolling the diagram, moving a marked field within a node or within the table. These functions you can be changed into modifying functions by the **VcGantt.ArrowKeyMode** property so that the user can move, enlarge or reduce the size of a node by them.

- Move nodes

By simply striking the arrow keys, a node will move; the smallest step size being the same as when moving the node by mouse. The step size can be enlarged by the property **VcGantt.ArowKeyStepMultiplier** and activated by holding the <Ctrl> key down in addition.

Key functions:

Arrow key <left/right>: move node

<Ctrl> + <Arrow key left/right>: modify step size

- Modify Duration

The duration can only be modified for all **visible** layers and only, if only one node ist marked. The above mentioned multiplier for the step size can be used as well.

Key functions:

<Shift> + arrow key <left/right>: change size of the node and thus modify its duration

<Shift> +<Ctrl> + arrow key <left/right>: modify step size

A window displaying information on the position will remain on the screen for a few more seconds after the interaction finished to let the user read its content.

For further information about the corresponding API properties please see the API reference guide.

The event **VcNodeModifying** occurs when the user has modified the length or the position of a node or a value in the **Edit Data** dialog. By the **modificationType** parameter you get further information of the kind of modification. If you set the returnStatus to **vcRetStatFalse**, the modification will be revoked

5.8 Moving Layers

Press the left mouse key to mark a layer and then move the mouse to shift the layer until releasing the mouse button again. When moving the layer horizontally, the **Move Layer** box continuously displays the current start and end dates of the layer, while the duration remains constant. Layers can only be moved within a row; there is no way to move a layer to a different row.

Move Layer	
Start:	09.09.2007
End:	23.09.2007

If you move a symbol layer, the **Move Layer** box will look like this:

Move Layer	
Date:	09.09.2007

5.9 Change Start/End Date

In a similar way you can modify just the start or the end date of a layer if you position the cursor on the outer left or right edge of the layer. The **Change Start Date** or **Change End Date** box (as appropriate) will appear that continuously displays the current start or end date. The duration will change.

Change Start Date		Change End Date	
Start:	01.09.2007	End:	12.09.2007
Duration:	13 days	Duration:	7 days

You can control shifting a layer in design mode by the corresponding **Moveable/Sizeable** buttons in the **EditLayer** dialog, to which you can get by the **Layer** property page. The event **VcNodeModifying** is triggered when a user modifies the length or the position of a node or a value in the **Edit Data** dialog. By the **modificationType** parameter you obtain more information on the kind of modification. If you set the returnStatus to **vcRetStatFalse**, the modification will be revoked.

5.10 Delete, Cut, Copy and Paste Nodes

Via the Del button you can delete marked nodes.

Via Ctrl-X you can cut marked nodes, via Ctrl-C you can copy nodes.

With the Shift key pressed, you can use the arrow up/down buttons to mark several nodes.

If the area of marked nodes contains a group in the mode **All nodes in one row** and **optimized**, all nodes of this group will be marked.

If the first node of such a group is marked and if you move the cursor with pressed Shift key into another row, all nodes of the target and of the start row will be marked.

You can insert copied or cut nodes via Ctrl-V above the target row (the row in which a node is marked).

You can insert copied or cut nodes via Ctrl-Shift-V below the target row.

The insertion position relative to the reference node will be indicated by appropriate arrows at the cursor symbol.

When you insert nodes, the order of grouped nodes will not be changed.

Nodes cannot be inserted in empty groups in the mode **All nodes in one row** and **optimized**.

5.11 Edit Node Data

In the dialog "Edit data" you can edit all node data. You open this dialog by either clicking on the **Edit** item of the corresponding context menu or by double-clicking on the node.

To edit several nodes, you mark the desired nodes and then click the **Edit** item of the context menu of one of the marked nodes to pop up the **Edit Data** dialog. Now you can edit the data of the marked nodes one after another

Fields	Values
ID	1
Name	
Start	1/2/2014
End	1/9/2014
Duration	5
Completion	
Group Level 1	
Group Level 2	
Release Date	
Due Date	

By double-clicking on a node, the event **VcNodeLeftDoubleClicking** is triggered.

Modifying a node interactively, e.g. by the **Edit Data** dialog, triggers the event **VcNodeModifying**. By the **modificationType** parameter you get further information of the kind of modification. If you set the **returnStatus** to **vcRetStatFalse**, the modification will be revoked.

The "Edit data dialog"

The name of the node as well as the number of the current node out of the total number of nodes marked is indicated.

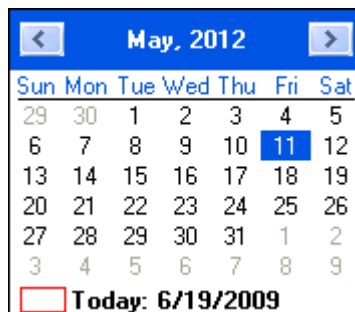
The table displays the data and values of the current node and lets you edit them. With the help of the arrow buttons above the table, you can navigate between the nodes. To store the current node data, click the **Apply** button.

Fields

This column displays the data fields that define the marked node. The data fields available are the ones defined by the data definition in the **Administrative data tables** dialog. Only data fields that are **not** defined as **hidden** are displayed.

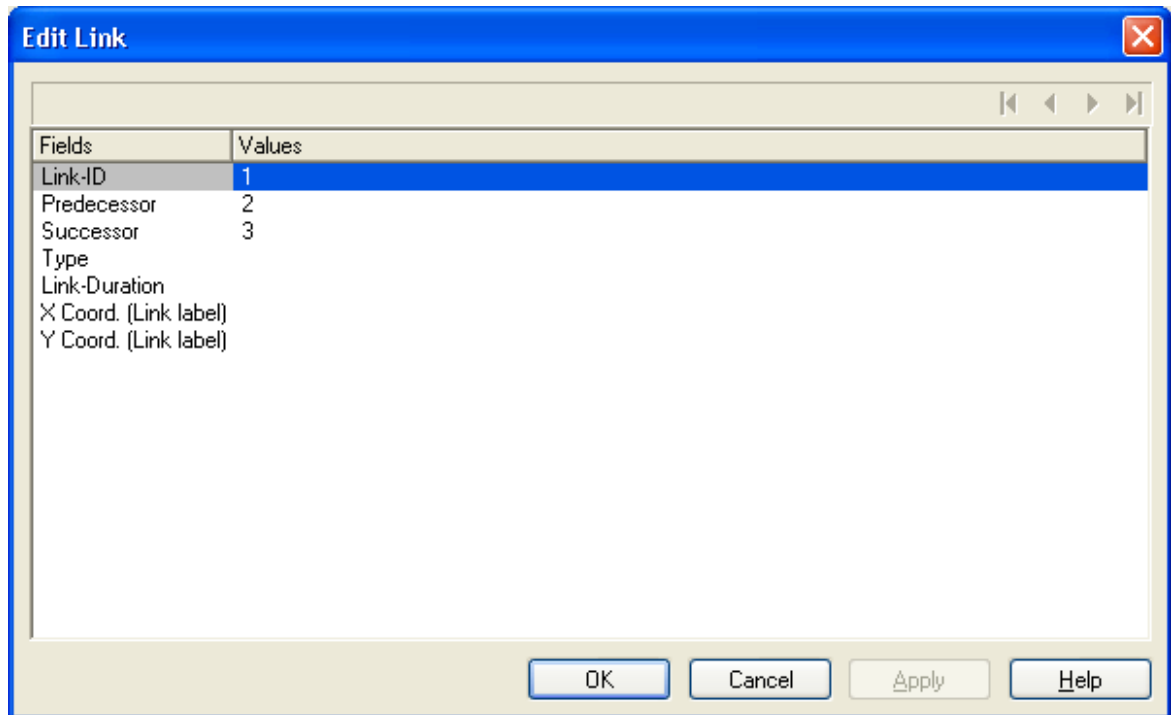
Values

This column lets you edit the values of the nodes marked, but only if they were defined to be **Editable** in the **Administrative Data Tables** dialog. If you edit a data field of the **Date/Time** type, a **Date** dialog will appear that you can select a date from.



The **Date Output Format** is defined on the **General** property page. When editing a field of the type **Integer** you can modify the value by a spin control that offers the desired values by up and down arrows.

5.12 Edit Links

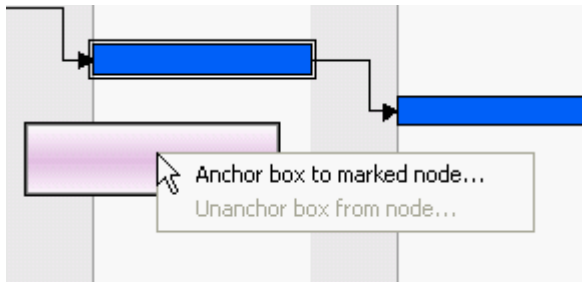


This dialog can be invoked by the method **VcGantt.EditLink**. Here you can view and edit the data of the marked link. The ID of the link is indicated at the first position of the list.

5.13 Anchor Box to Node

Boxes can be anchored to nodes either interactively (mouse + Shift key or context menu) or by using the corresponding API properties and methods.

- **Anchoring by mouse:** Point with the mouse to the box you want to tie to a node and press the Shift key. A little anchor appears. Keep the Shift key pressed and draw a line between the box and the desired node. The box is now anchored to the node. If you have ticked the check box **Anchoring line visible** in the **Administrative boxes** dialog, a line is displayed. Follow the same steps to untie the box again.
- **Anchoring over context menu:** Mark the node to which you want to anchor the box and select **Anchor box to marked node** from the context menu of the box. If the context menu does not pop up, you have to tick **Show context menu for the box** on the **General** property page.



Select **Unanchor box from node** to untie the box again.

If you want to tie the box to another node, carry out the same steps as described above, either by mouse or over context menu.

- **Anchoring via API:** Please see the API Reference Guide for a detailed description of the property **AnchoringInteractionsAllowed** and the method **AnchorToNode** of the object **VcBox**

A box which was anchored can be still moved interactively (provided that you have ticked the check box **Moveable** in the **Administrative boxes** dialog).

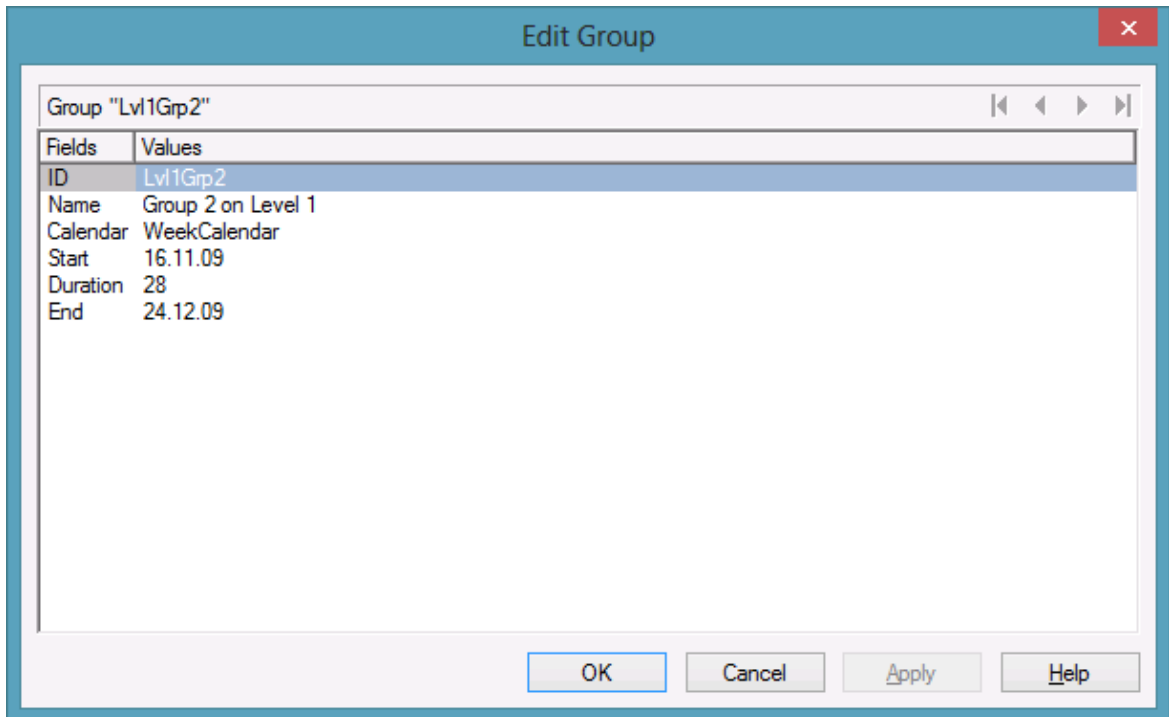
If you move a node which is anchored to a box, the box is moved as well. If the node is collapsed, the box is collapsed as well, thus becoming invisible. When the node is expanded the box is visible again.

If a box is tied interactively to a node, its position on the screen will be maintained. The offset values which are used as basis are converted according to the reference points (Origin, ReferencePoint). If, for example, a box with a certain offset refers to a chart at the top left (origin) and then is anchored to a node, an offset to the the top left node is calculated automatically. This makes sure that the position on the screen will not be

altered. If the box is untied from the node the calculation is carried out backwards.

This method is applied as well when using the API property **AnchorToNode** but not when setting the property **NodeID**.

5.14 Edit Group Data



You can get to this dialog by the context menu of the group or by double-clicking a group layer (which will only be displayed if in the **Grouping dialog** the box **Group node visible** has been ticked).

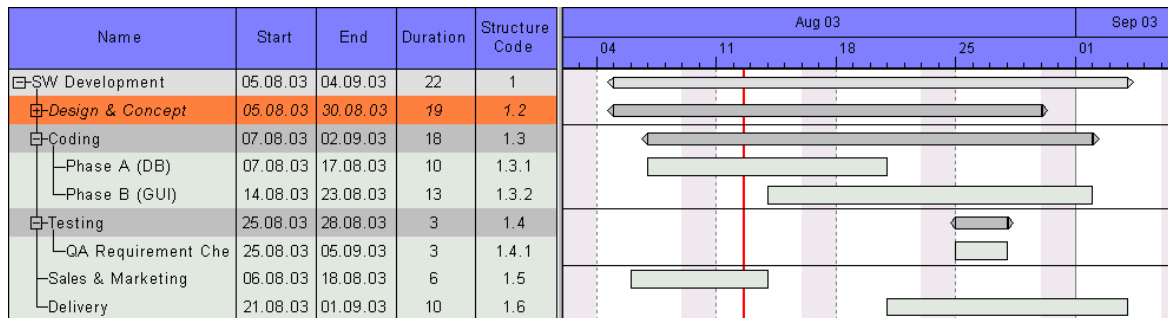
The dialog lets you edit the data of one group or, if more than one group has been marked, the data of every marked group one after the other.

The number of the current group out of the total number of marked groups is indicated above the list.

The arrow buttons above the list allow to navigate to the previous or next (or first or last) marked node.


5.15 Collapsing/Expanding Groups

If a grouping is specified and the **Modifications allowed** box in the **Grouping** dialog is ticked, you can expand a collapsed group/collapse an expanded group by double-clicking on the group heading or by clicking on the **plus** or **minus** symbol of the group heading.



The event **VcGroupModifying** occurs when a user interactively modifies a group. The group object, the type of modification and the return status are returned. If you set the returnStatus to **vcRetStatFalse**, the modification will be revoked.

5.16 Moving Groups

Groups can be moved vertically in the table as well as in the diagram (by dragging the summary bar) when the checkboxes **Moving groups vertically via table** and/or **Moving groups vertically via diagram** in the dialog **Grouping** have been ticked. While dragging, a corresponding cursor indicates where the group will be positioned .

Tip: Groups can only be moved within a parentgroup.

5.17 Editing Fields in the Table

To edit a table field please click on the field and either enter new contents or modify the current one. There are further ways of editing the field contents in the table which are only available after having ticked the **Extended editing allowed** box on the **General** property page.

You can then modify date and time fields by clicking on the arrow button. For further information about the usage of the date dialog box see chapter 4.40 The "Specify Date Lines" Dialog.

The value of numeric data fields may be increased or decreased by clicking on the up or down arrow buttons.

For further information about extended editing see chapter 4.2 "The General Property Page".

Note: By pressing the ESC-key you can leave the field without saving the changes.

5.18 Modifying Table Column Width

You can change the width of a column in the table interactively by moving the separation line between the columns in the table caption.

ID	Name	↔	Start	End
----	------	---	-------	-----

You can change the width of a table column in the table caption only.

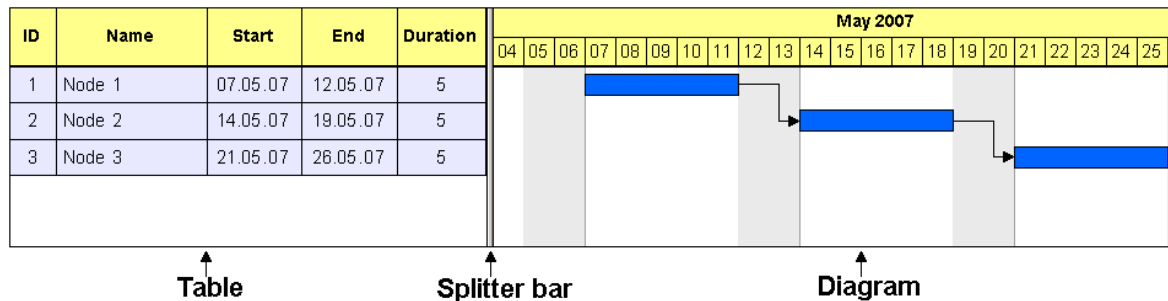
The event **VcTableWidthChanging** occurs when the user modifies the width of the table. The table and the modified diagram aspect ratio are returned. If you set the returnStatus to **vcRetStatFalse**, the modification will be revoked.

The event **VcTableColumnWidthChanging** occurs when the user modifies the width of a table column. The table, the index and the current width (as 1/100 mm) of the modified column are returned. If you set the returnStatus to **vcRetStatFalse**, the modification will be revoked.

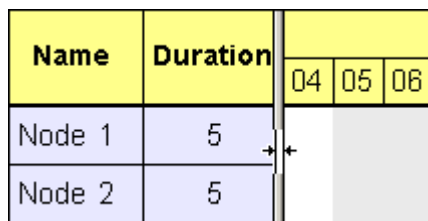
The column width can be calculated automatically as well. For that, on the **General** property page the **Table column width optimization allowed** box has to be checked. After that, a double-click on a column separation line at run time will cause the automatical adjustment of the column width on the left to the length of the text contained in the column. This will trigger the **VcTableColumnWidthOptimizing** event. If the optimization has been carried out, the event **VcTableColumnWidthChanging** will be triggered.

5.19 Modifying the Table/Diagram Ratio

The table and the diagram are separated from one another by a splitter bar.



If you move the mouse over the splitter bar, the pointer shape changes to a double vertical line with an arrow to the left and right.



The pointer must be positioned in the area between the table rows and diagram, not in the area between the table caption and the timescale.

By dragging the mouse, you can now change the width ratio of the table to the diagram. (The maximum table width is limited by the total of column widths specified in the **Edit Table** dialog.)

5.20 Inserting table rows

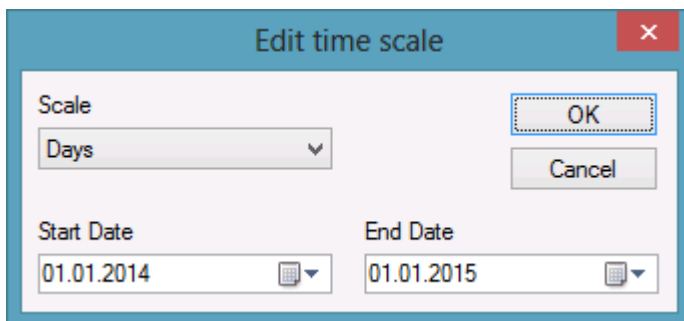
If the check box **Extended editing allowed** on the **General** property page was ticked, the Ins-key can be used for inserting a table row above the current one. If no row was marked, the new line is inserted at the end of the table.

5.21 Modifying the Timescale

In the **Edit Time scale** dialog box you can set the time scale type (minutes, hours, days, weeks, months) and the start and end of the time scale.

When shifting the beginning of the time scale, the beginning must not be shifted beyond the end of the first section, if more than a single section was defined.

You can open this dialog by double-clicking on the time scale or selecting the corresponding context menu item. When shifting the beginning of the timescale, the beginning must not be shifted beyond the end of the first section, if more than a single section has been defined.



By double-clicking on the time scale, the event **VcTimeScaleLeftDoubleClicking** is triggered. The `TimeScale` object and the mouse position (x,y-coordinates) are returned. If you set the `returnStatus` to `vcRetStatFalse`, the integrated **Edit Timescale** dialog box will be revoked.

The "Edit time scale" dialog

Scale

Select the timescale. Choose between minutes, hours, days, weeks and months.

Start Date

Specify the start date of the timescale. If you click on the arrow button, a Date dialog will appear that you can select a date from.



The date output format is defined on the **General** property page.

End Date

Specify the end date of the timescale. If you click on the arrow button, a Date dialog will appear that you can select a date from.

The date output format is defined on the **General** property page.

5.22 Modifying the Scaling and the Borders of Sections

Scaling Timescale Sections



You can rescale a time scale section interactively by positioning the mouse cursor onto the section, pressing the left mouse key and dragging the mouse towards the left or right. The shape of the cursor will change to a vertical line with an arrow to the left and right. Dragging the cursor towards the left will downsize the width of the timescale units, dragging it to the right will blow them up. While dragging an info box will pop up to inform you about the percentage by which the time scale section is altered.

Note: The closer you place the cursor to the beginning of a section, the stronger the enlargement/downsizing will be. If you intend to enlarge/downsize a lot, you are suggested to place the cursor close to the beginning on the left, while for smaller adjustments placing the cursor towards the end on the right is suggested.

The event **VcTimeScaleSectionRescaling** occurs when the user rescales a section of the time scale. The TimeScale object, the section index and the current **BasicUnitWidth** are returned. If you set the return status to **vcRetStatFalse**, the modification will be revoked.

Moving the Limits of a Timescale Section



You can move the limits between two timescale sections by shifting the separating line between them. The shape of the cursor will change to a vertical double-line with an arrow to the left and right.

The event **VcTimeScaleSectionStartModifying** occurs when the user modifies the start date of a section interactively. The TimeScale object, the section index and the current start date are returned. If you set the returnStatus to **vcRetStatFalse**, the modification will be revoked.

5.23 Moving the Date Line

You can modify the date of a date line by moving it via the mouse.

Before, on the **Specify Date Line** dialog the **Moveable** check box of the corresponding date line has to be activated for the relevant date line.

Beside, you can generate date lines via the API.

The event **VcDateLineModifying** occurs when the user has moved a date line. The modified date line object is captured and returned so that you receive the new values. If you set the return status to **vcRetStatFalse**, the modification will be revoked.

5.24 Setting up pages

All settings concerning the page layout can be made in the corresponding dialog which can be opened either by clicking the **Page setup** item of the diagram contextmenu or by clicking the corresponding button in the **Print preview**.

Page Setup

Scaling

Mode: **Fit to page counts**

Zoom factor: **100.0** %

Maximum width: **1** page(s)

Maximum height: **1** page(s)

Current	
	9.97
	1
	1

☐ Repeat title/table/time scale/legend

☒ Show table

☐ Adopt appearance from view on screen

Table columns (e.g. 1-5;7):

☒ Show diagram

☐ Time scale start: **01.01.2014**

☐ Time scale end: **01.01.2015**

☐ Adjust time scale to width of pages

Options

☒ Pad pages with space

☒ Show frame outside

Alignment: **Centered**

☐ Show crop marks

☐ Show folding marks (DIN 824): **Form A**

Footer line

☐ Page numbering: **Row.Column**

☐ Text:

☐ Additionally print current date

Sheet margins

Left: **1.5** cm

Top: **1.0** cm

Right: **1.0** cm

Bottom: **1.0** cm

OK **Cancel**

Mode

By selecting a scaling mode from the drop down list and setting the corresponding values **Zoom factor** and **Maximum width/height** you specify a zoom factor for your output. After having clicked the **Apply** button, the values which result from your settings are shown under **Current**.

Zoom factor

100% is equivalent to the original size; a smaller value correspondingly reduces the size of the diagram, a greater value increases it.

Fit to page counts

By selecting this option you can specify the maximum number of pages, both heightwise and widthwise, into which the diagram may be split for the output. If necessary, one of the two values may be ignored in order to print the diagram as large as possible while preventing it from being distorted.

Zoom with horizontal fit

This option lets you regulate the pagination by selecting a zoom factor as well as a fixed number of pages in width. This number of pages is reached by downsizing or expanding the time scale.

Repeat title/table/timescale

By ticking this check box title, table and timescale of a diagram that was partitioned into pages will be added to each page.

Show table

Specify whether the table is to be printed or not. If you don't tick the check box, the table will not be printed.

Adopt appearance from view on screen

This option lets you specify whether the table width that is currently shown on the screen is to be adopted for the print preview and for the output.

This feature can also be set by the property **VcPrinter.TableWidth-AdoptionFromViewOnScreen**.

Show table columns

Here you can set the number of table columns to be printed. Specify single columns or ranges of columns, that are to be separated by commas or semicolons. Example: "1;5-7;3" specifies the columns 1 and 3 and the range from 5 to 7.

Show diagram

Specify whether the diagram (timescale and layers) shall be also printed or not.

Time scale start

This option lets you specify the start date of the time range to be used for the output. The time range can only be restricted in comparison to the time range displayed on the screen. Hence only a later start date than that having been set by the VcGantt property **TimeScaleStart** leads to a modified output.

This feature can also be set by the property **VcPrinter.TimeColumnStartDate**.

Time scale end

This option lets you specify the start date of the time range to be used for the output. The time range can only be restricted in comparison to the time range displayed on the screen. Hence only an end date prior to that having been set by the VcGantt property **TimeScaleEnd** leads to a modified output.

This feature can also be set by the property **VcPrinter.TimeColumnEndDate**.

Adjust time scale to width of pages

This option leads to a better utilization of the printing pages:

- If scaling fit to page is selected: The zoom factor is calculated in such a way that the space of the selected number of pages is fully used for printing into the height while the time scale gets downsized or enlarged so that the selected number of pages is used to full capacity into the width.
- If a scaling via zoom factor is selected: The time scale gets downsized or enlarged so that the selected number of pages is being used to full capacity into the width.

Pad pages with space

This option lets you specify whether enough space is to be left between the diagram and the boxes of the title and legend area so that the boxes are always printed in full width and are fixed to the margin. If the option is not selected, there will be no space left between the diagram and the boxes and their width may vary on the different pages depending on the diagram.

Frame outside

*Only activated if the **Repeat title/table/timescale** check box has been ticked.* If you tick this box, each page will be given a frame, otherwise a frame will be drawn around the whole of the diagram.

Alignment

Select one of the possible alignments for the diagram from the list.

Show crop marks

If you tick this check box, crop marks will be printed on the edges of the diagram that help gluing together the single pages to get a complete chart.

Show folding marks (DIN 824)

Specify folding marks to fold your drawing according to DIN standard 824 (current version from 1981) for the folding of constructional drawings. The following formats are available:

- **Form A:** includes a filing margin on the left side so that the drawing can be punched and filed away
- **Form B:** slightly smaller so that a flexi filing fastener can be applied and together with the fastener the drawing corresponds to the width of DIN A4.
- **Form C:** the folded drawing is not to be punched but to be put in a sheet protector

The available folding marks can be displayed for every format, whereas the DIN 824 only mentions the formats DIN A0 to A3 explicitly.

Page numbers

If you tick this check box, a page number will be displayed in the bottom left-hand corner of each page. The following possibilities are available:

- **Row.Column:** Useful for charts stretching across more than one pages both heighthwise and widthwise. The vertical position of the page is displayed before the dot, the horizontal position after it.
- **Column.Row:** Useful for charts stretching across more than one pages both heighthwise and widthwise. The horizontal position of the page is displayed before the dot, the vertical position after it.
- **Page/Count:** The current page number is displayed before the slash and after it the total number of pages: 1/6, 2/6 etc.

Text

Please tick this check box to set a text into the bottom left-hand corner of each page. If there is a page number, the additional text will be placed right of it.

For numbering the pages you may enter in **Additional text** the following place holders which will be replaced with the appropriate contents on the printout:

{PAGE}	= consecutive numbering of pages
{NUMPAGES}	= total number of pages
{ROW}	= line position of the section in the complete chart
{COLUMN}	= column position of the section in the complete chart

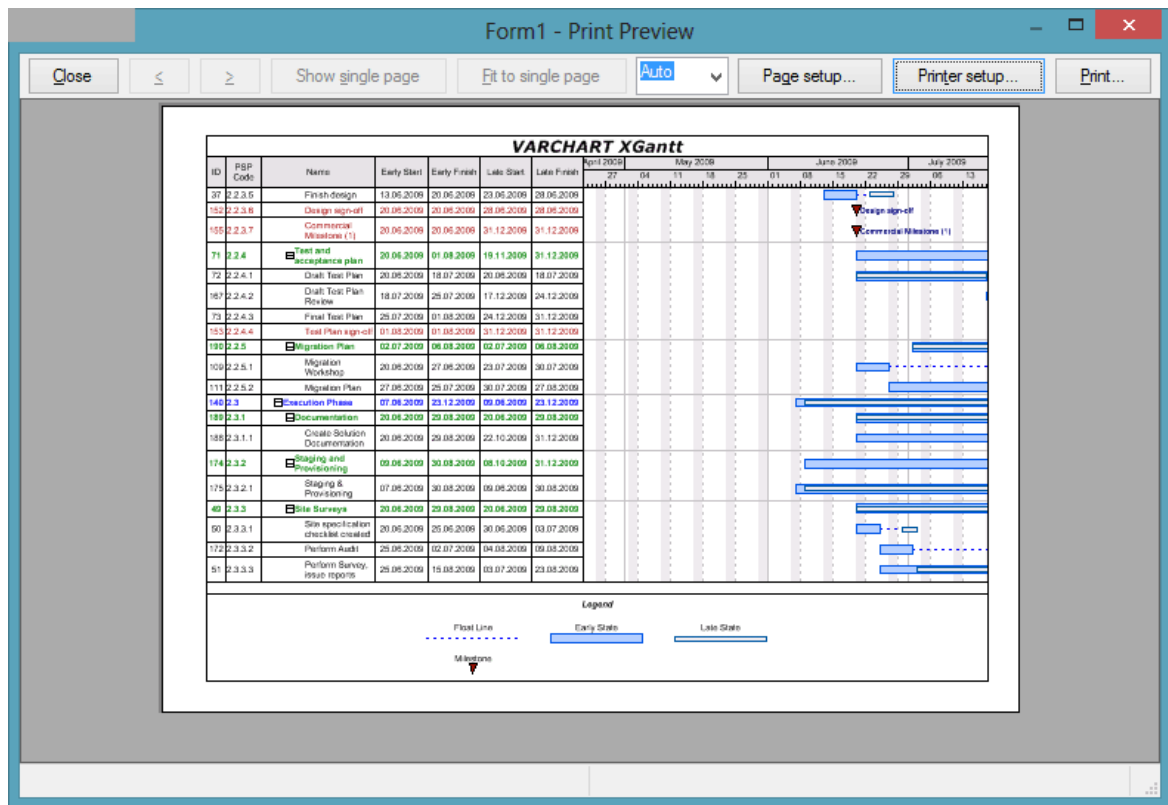
Additionally print current date

If you tick this check box, the date of printing will be printed in the bottom left corner. If there is a page number or an additional text, the print date will be placed right of them.

Sheet margins

The fields **Top**, **Bottom**, **Left** and **Right** let you set the margin between the diagram and the edge of the paper sheet (unit: cm). Minimum margins existing for technical reasons cannot be overridden by the values entered here. Printers that by default print minimum margins will add the values entered here to the default minimum margins, thus resulting in broader margins than visible in these settings.

5.25 Print Preview



Before printing, you can view the diagram in the print preview where it will be displayed as defined by the settings of the **Page Setup** dialog and as it will be printed.

You can view single pages or an overview of all pages or you can zoom and print a certain section of your diagram interactively.

The status bar shows the total number of pages and their horizontal and vertical spreading. In the **Single Page** mode, also the number of the current page is shown.

Close

By clicking on this button, you will leave the page preview and return to your diagram.






*Only activated when the **Single** button has been pressed.* If the diagram consists of more than one page, you can click this button to view the previous page. You traverse the pages horizontally starting from the bottom right and finishing at the top left page.

>

*Only activated when the **Single** button has been pressed.* If the diagram consists of more than one page, you can press this button to view the next page. You traverse the pages horizontally starting from the top left and finishing at the bottom right page.

Show Single Page/Overview

If the diagram consists of more than one page you can either view the pages one by one or in the overview. The overview shows all pages, their size depending on the total number of pages. The **Single Page** mode initially shows the first page in full size, the buttons  and  allowing to browse through the pages. By double-clicking a page you can easily switch between the two modes **Single Page** and **Overview**.

If you want to zoom a certain section of your diagram, switch to the **Single Page** mode and with the mouse draw a rectangle around the desired section while holding down the left mouse button. As soon as you release the button, the selected section will be enlarged and can be printed by clicking the  button that appears in place of the **Print** button. Please note that the zooming factor will not influence the scaling factor set in the **Page Setup** dialog.

Fit To Single Page

This button lets you scale down a multiple-page diagram to one page. The **Fit To Single Page** mode also allows to zoom a certain section as described under **Show Single Page/Overview**

Zoom factor

You can modify the size of the diagram by selecting a zoom factor from the list or by defining an individual one. This is only possible in the "Show Single Page" mode. To modify the zoom factor you can also use the scroll-wheel while holding down the <CTRL> key. The zoom factor it will not modify the size of the output. Depending on the selected zoom factor, vertical and/or horizontal scroll bars will be displayed. Alternatively, you can use the mouse wheel to scroll vertically, holding down <Shift> to scroll horizontally.

The zoom factor **Auto** is the pre-set default and will always enlarge or downsize the sheet to the full size of the screen.

Page Setup

When clicking on this button, you will get to the dialog **Page Setup** to modify page settings.

Printer Setup

*Only visible if the check box **Use PrintDlgEx dialog** on the **General** property page has not been ticked.*

When clicking on this button, you will get to the Windows dialog **Printer Setup**, where you can modify printer settings.

Print/Print Area

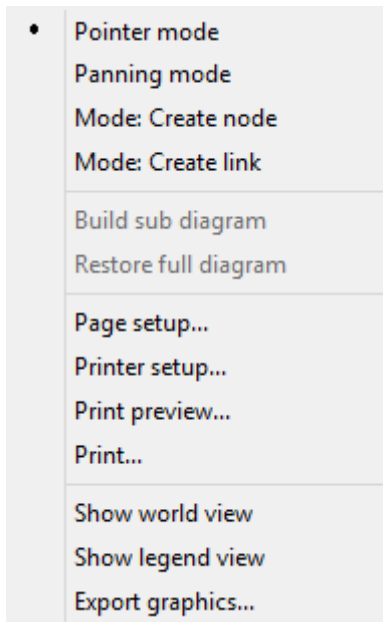
Click on this button to reach the Windows **Print** dialog box to start the print procedure.

If you have zoomed a section in the page preview, the button's label will change to **Print Area** and when you click it, the **Selection** radio button in the Windows **Print** dialog box will already be selected. If you click on **OK** the section displayed on the screen will be printed.

Please note that the zooming factor will not influence the scaling factor set in the **Page Setup** dialog.

5.26 Context Menu of the Diagram

If you press the right mouse key when the cursor is positioned in the diagram area (but not on a node), the following context menu will appear:



The event **VcDiagramRightClicking** occurs when the user clicks the right mouse key on the diagram, not hitting a node. The position of the mouse (x,y-coordinates) is captured, so that you can for example display your own context menu at the appropriate location. If you set the returnStatus to **vcRetStatNoPopup**, the integrated context menu will be revoked.

Pointer mode

The pointer mode is the default mode. It allows all types of interactions except for generating nodes and links.

Panning mode

In the panning mode you can move certain screen sections by way of a cursor shaped like a hand.

The panning mode has to be activated on the **General** property page.

Mode: Create node

This mode is available only if the **Allow new nodes** option on the **Nodes** property page is activated.

In this mode, the pointer shape changes to a small cross. While in this mode, you can create a node by dragging the mouse and pressing the left mouse button. A little box will appear at the current position of the mouse which shows the current start and end date and the duration of the new node.

Create Activity	
Start:	07.09.2007
End:	09.09.2007
Duration:	2 days

If you are creating a node in a collapsed group, additionally to the small cross an arrow appears: It shows whether the new node will be the first node in the group (arrow up) or the last one (arrow down).

If the **Edit new node** option on the **Nodes** property page is activated, the **Edit Data** dialog box will appear, as soon as you release the mouse button. In the **Edit Data** dialog box you can edit all data of the new node.

If you have not defined anything else in your settings, the node just created will appear at the current position of the mouse.

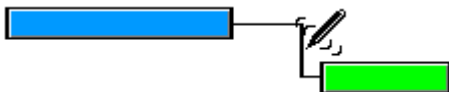
The **Mode: Create Node** can also be activated by setting the property **InteractionMode** to the value **VcCreateNode**.

The event **VcNodeCreating** occurs when the user creates a node. The node object is captured, so that a validation can be made. For the validation, the **Edit Data** dialog box has to be activated. If you set the returnStatus to **vcRetStatFalse**, the node will be deleted.

Mode: Create link

The pointer shape changes into a pencil. Use the mouse to draw a link between two nodes and create a finish-start link.

This mode is available only if the **Show Links** option on the **Links** property page is activated.



The event **VcLinkCreating** occurs when the user creates a link between two nodes. The generated link object is returned, so that a validation and if necessary a data base entry can be made. If you set the returnStatus to **vcRetStatFalse**, the link will be deleted again.

Mode: Create box

This mode is available only if the **Allow new boxes** option on the **General** property page is activated.

While in this mode, you can create a box by dragging the mouse and pressing the left mouse button.

The **Mode: Create box** can also be activated by setting the property **InteractionMode** to the value **VcCreateBox**.

Also see the events **VcBoxCreating** and **VcBoxCreated**.

Build sub diagram

(only active if nodes are marked) Select this item to display a subdiagram of the marked nodes.

Restore full diagram

*(only active if the option **Build sub diagram** has been selected before)* Select this item to restore the full diagram.

Page setup

The **Page Setup** dialog box appears.

The **Page Setup** dialog box also can be invoked by the VcGantt method **ShowPageSetupDialog**.

Print setup

*Only selectable if the check box **Use PrintDlgEx dialog** on the **General** property page has not been ticked.*

The Windows **Print Setup** dialog box appears. This dialog box also can be invoked by the VcGantt method **PrinterSetup**.

Print preview

The **Page Preview** dialog box appears. This dialog box also can be invoked by the VcGantt method **ShowPrintPreviewDialog**.

Print

Select the **Print** option to reach the Windows **Print** dialog box. This dialog box also can be invoked by the VcGantt method **ShowPrintDialog**.

Show world view

This menu item lets you switch on/off the world view. The world view is an additional window that shows the complete diagram. A frame marks the diagram section currently displayed in the main window. If you move this frame with the mouse, the according diagram section is displayed in the main window.

The world view also can be displayed oder hidden by the property **VcWorldView.Visible**.

Show legend view

This menu item lets you switch on or off the legend view. The legend will appear in a separate window.

The legend view also can be displayed oder hidden by the property **VcLegendView.Visible**.

Export graphics

When you choose this option, the Windows **Save As** dialog box appears where you can save the current graphics as a graphics file.

This dialog box also can be invoked by the VcGantt method **ShowExportGraphicsDialog**.

When exporting, the size of the exported diagram will be calculated this way:

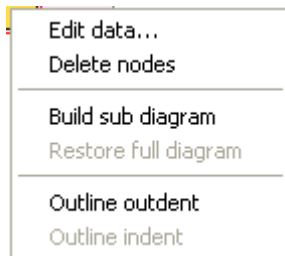
- PNG: a resolution of 100 dpi and a zoom factor of 100% are assumed. If alternatively a value of ≤ -50 is specified in the parameter **SizeX**, the absolute number will be used as DPI input.
- GIF, TIFF, BMP, JPEG: a resolution of 100 dpi and a zoom factor of 100% are assumed. If alternatively a value of ≤ -50 is specified in the parameter **SizeX**, the absolute number will be used as DPI input. In addition, an internal limit of 50 MBs of memory size is required for the uncompressed source bit map in the memory; so larger diagrams may have a smaller resolution than expected.

- WMF: A fixed resolution is assumed where the longer side uses coordinates between 0 and 10,000 while the shorter side uses correspondingly smaller values to keep the aspect ratio.
- EMF/EMF+: The total resolution is adopted, using coordinates scaled by 1/100 mm.

For further details on the different formats please read the chapter "Important Concepts: Graphics Formats".

5.27 Context Menu of Nodes

If you click the right mouse key on one or several marked nodes, the following menu will appear:



The event **VcNodeRightClicking** occurs when the user clicks the right mouse key on a node (location = vcInDiagram) or on a table entry related to an activity (location = vcInTable). The node object hit and the mouse position (x,y-coordinates) are returned, so that you can display a context menu at the appropriate position. If you set the returnStatus to **vcRetStatNoPopup**, the integrated context menu will be revoked.

Edit Data

Opens the **Edit Data** dialog box. If you marked more than a single node, you can edit them right away.

Delete Nodes

Select this option to delete the marked node(s).

Build sub diagram

Select this item to display a subdiagram of the marked nodes.

Restore full diagram

*(only active if the option **Build sub diagram** has been selected before)* Select this item to restore the full diagram.

Outline outdent

(only for hierarchy) The position of the marked node in the hierarchy will be increased.

Outline indent

(only for hierarchy) The position of the marked node in the hierarchy will be decreased.

5.28 Context Menu of Links

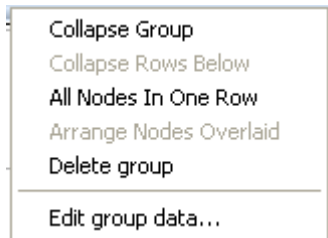
If you click the right mouse key on a link, the **Delete Link** context menu will appear. To delete the marked link, please click the left mouse key to confirm.

Delete link

The event **VcLinksRightClicking** occurs when the user clicks the right mouse key on a link or on several overlapping links. The `LinkCollection` object and the mouse position (x,y-coordinates) are captured and passed, so that you can display your own context menu at the appropriate position. If you set the `returnStatus` to **vcRetStatNoPopup**, the integrated context menu will be revoked.

5.29 Context Menu of Groups

If you right-click on a group title in the table or a group layer in the diagram (which will only be displayed if in the **Grouping** dialog the checkbox **Group node visible** has been ticked), a context menu will appear that offers basic options on groups:



The event **VcGroupRightClicking** occurs when the user clicks the right mouse key on a group heading in the table. The group object and the mouse position (x,y-coordinates) are returned, so that you can display a context menu at the appropriate position. If you set the `returnStatus` to **vcRetStatNoPopup**, the integrated context menu will be revoked.

Collapse/Expand Group

This menu item lets you expand a collapsed group or collapse an expanded one.

Expand/Collapse Rows Below

This menu item lets you expand the rows of a collapsed group or collapse the rows below an expanded group respectively.

If you have chosen for the group **All Nodes In One Row**, this option will collapse only the subgroups of the selected group.

All Nodes In One Row/Nodes In Separate Rows

If you choose the option **All Nodes In One Row** all activities in a group will be displayed in one line. If the activities in the group coincide, they will be automatically displayed underneath one another in expanded mode to prevent overlapping. If the group is collapsed, the activities may overlap.

With this type of arrangement, the table section for the activities is suppressed, so you will need to utilise the layer annotation or tooltip to identify the activities for the user.

The option **Nodes In Separate Rows** lets you display each node in its own row.

Arrange Nodes Optimized/Arange Nodes Overlaid

*(Selectable only, if **All Nodes In One Row** was selected.)*

If you select **Arange Nodes Overlaid**, the nodes are displayed in one row, even if they are overlapping each other.

If you select **Arrange Nodes Optimized**, the layout of the nodes will be optimized to avoid overlapping, even if they require more space than a single row.

Delete group

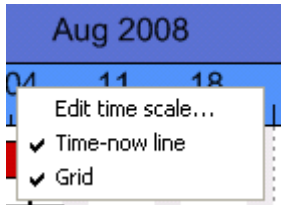
This menu item lets you delete an empty marked group.

Edit group data

The corresponding dialog will appear.

5.30 Context Menu of the Time Scale

A right mouse click on the time scale will open the below menu:



The event **VcTimeScaleRightClicking** occurs when the user clicks the right mouse key on the timescale. The **TimeScale** object and the mouse position (x,y-coordinates) are returned. At this position you can show your customized context menu. If you set the **returnStatus** to **vcRetStatNoPopup**, the integrated context menu will be revoked.

Edit Timescale

Select this option to reach the **Edit Timescale** dialog box.

Time-now Line

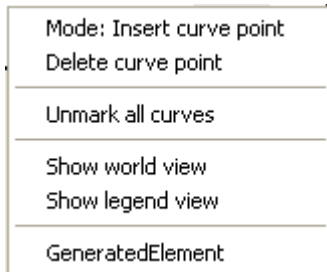
Specify whether your diagram should contain a time-now line (date line).

Grid

Specify whether your diagram should contain grid lines.

5.31 Context Menu of the Curve

If you press the right mouse button in an empty section of the histogram or on a curve, the below context menu will occur:



If the user presses the right mouse button on an empty section of the histogram or on a curve, the event **VcHistogramRightClicking** or **VcCurveRightClicking** is triggered, respectively. The histogram or curve object and the mouse position (x,y-coordinates) are returned by the parameters of the event. You can suppress the integrated context menu at the given position by setting the returnStatus to **vcRetStatNoPopup** and pop up your own context menu.

Mode: Insert curve point

In this mode you can add a curve point by pressing the left mouse button.

Delete curve point

To delete a curve point, click on it with the right mouse button and select the option **Delete curve point** in the context menu.

Unmark all curves

All curves will be unmarked.

Show world view

This menu item lets you switch on or off the world view. The world view is an additional window that shows the complete diagram including the histogram. A frame points out the section currently displayed in the main window.

Show legend view

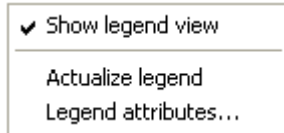
This menu item lets you switch on or off the legend view which is an additional window for showing the legend.

Curves

If available, the API curves are indicated in this context menu, where they can be marked.

5.32 Context Menu of the Legend

A right mouse click on the legend will open the below menu:



Show legend view

This menu item lets you switch on or off the legend view.

Actualize legend

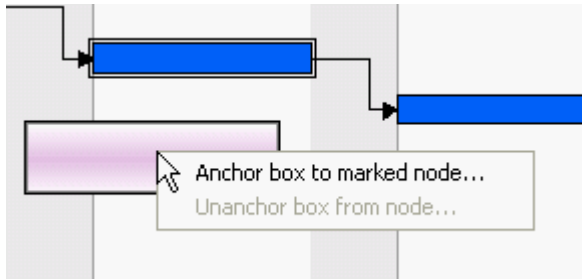
This menu item lets you refreshing the legend which is needed after modifications in the chart, such as adding or deleting nodes, because they are not displayed automatically in the legend. The refreshing can also be carried out by switching off and on the legend view. This concerns the loading of nodes as well. If on the property page **Additional views** the attribute **Initially visible** was selected for the legend view and no nodes have been loaded when running the program, the legend stays empty until it was refreshed.

Legend attributes

With this item you open the corresponding dialog where you can specify the settings concerning legend title, legend elements and margins. For further information about this dialog please see chapter 4.44 "The Legend Attributes Dialog Box".

5.33 Context Menu of Boxes

A right mouse click on a box will open the below menu:



If the context menu does not pop up, you have to activate the option **Show context menu for the box** on the **General** property page.

Anchor box to marked node

This item lets you anchor a box to the marked node. This is only possible if you have selected the option **Anchoring interactions allowed** in the **Administrate boxes** dialog.

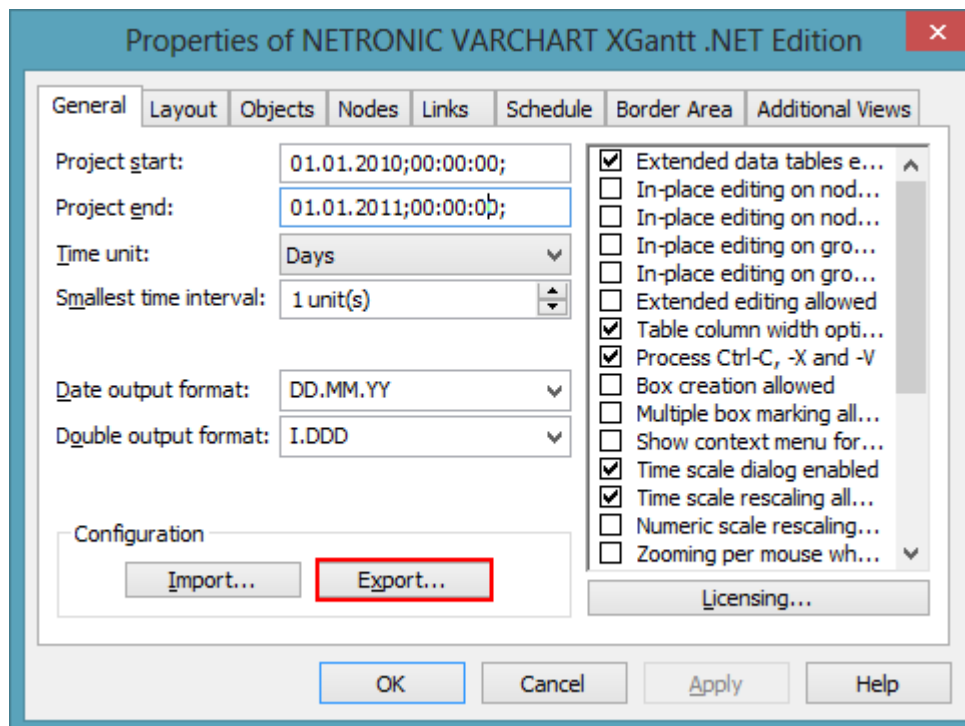
Unanchor box from node

This item lets you anchor a box to the marked node.

6 Frequently Asked Questions

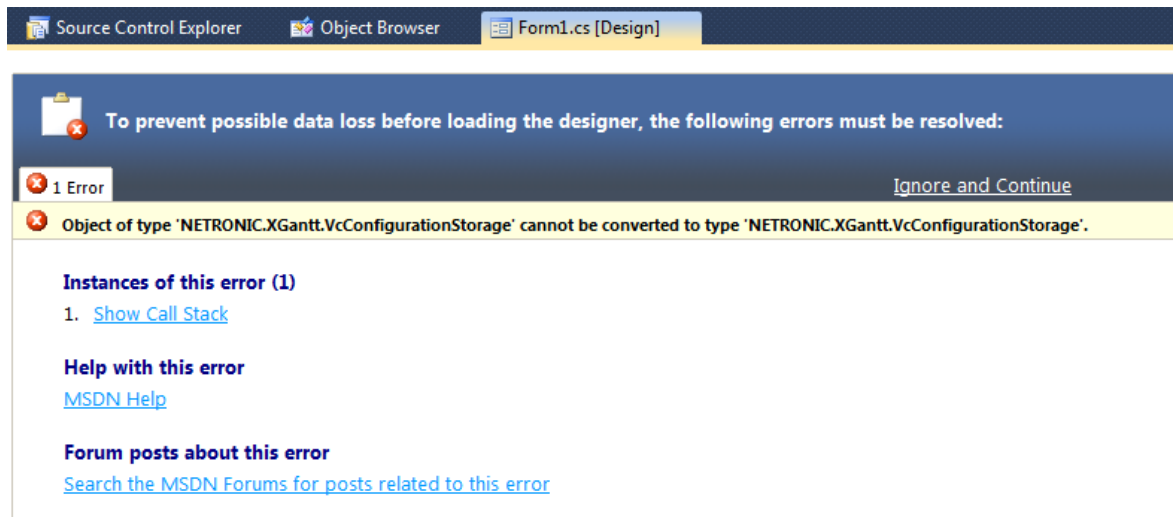
6.1 How to Upgrade from VARCHART XGantt .NET 4.4 to VARCHART XGantt .NET 5.0?

1. Before installing VARCHART XGantt.NET 5.0, please open the form designer of Visual Studio with the form using XGantt 4.4 and save the current configuration of XGantt by clicking the **Export** button on the **General** property page:

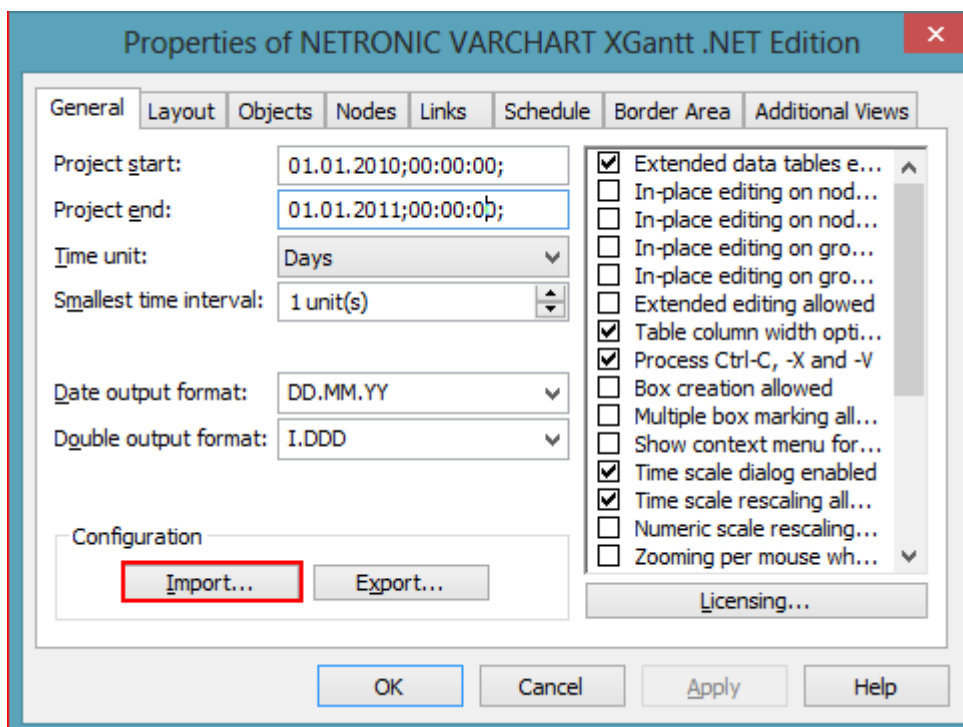


2. First, close the form and then end Visual Studio.
3. Install VARCHART XGantt .NET 5.0
4. Open your project in Visual Studio again and in the Solution Explorer delete the reference NETRONIC XGantt (still referring to XGantt 4.4) and insert a new reference NETRONIC XGantt, having to refer to VARCHART XGantt 5.0.
5. Open the form designer with the form containing XGantt. The following error message will appear:

432 Frequently Asked Questions



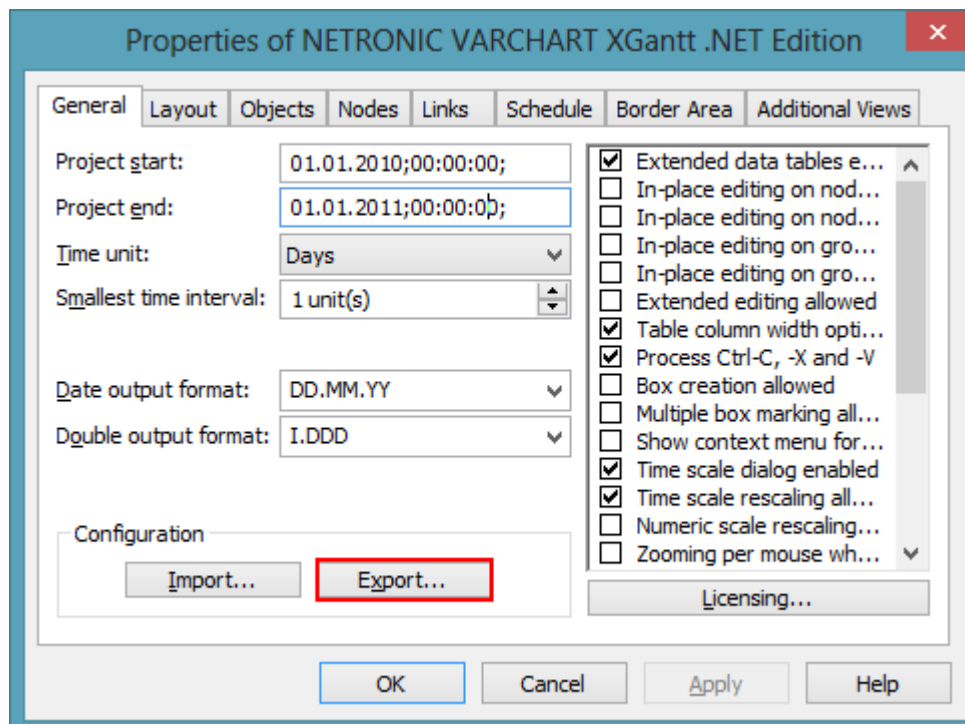
6. Click **Ignore and Continue**. The form in the form designer will be displayed correctly again but the XGantt will be set back to its default configuration.
7. Now import the configuration you have saved before by clicking the **Import** button on the **General** property page.



8. VARCHART XGantt now uses your individual configuration again.

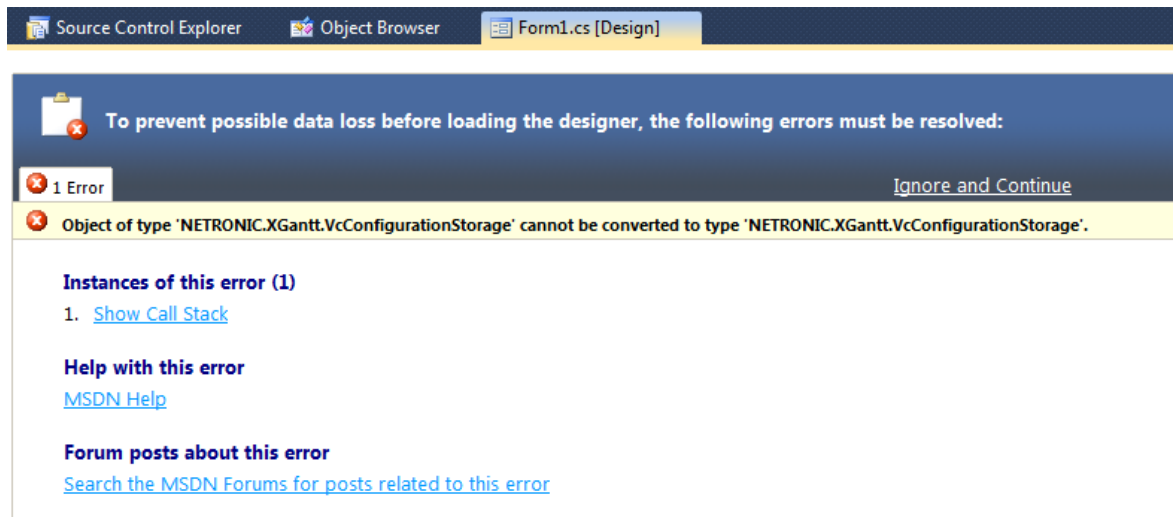
6.2 How to Upgrade from one Build of VARCHART XGantt .NET to a new one (within the same version)?

1. Before installing VARCHART XGantt.NET 5.0, please open the form designer of Visual Studio with the form using XGantt 4.4 and save the current configuration of XGantt by clicking the **Export** button on the **General** property page:

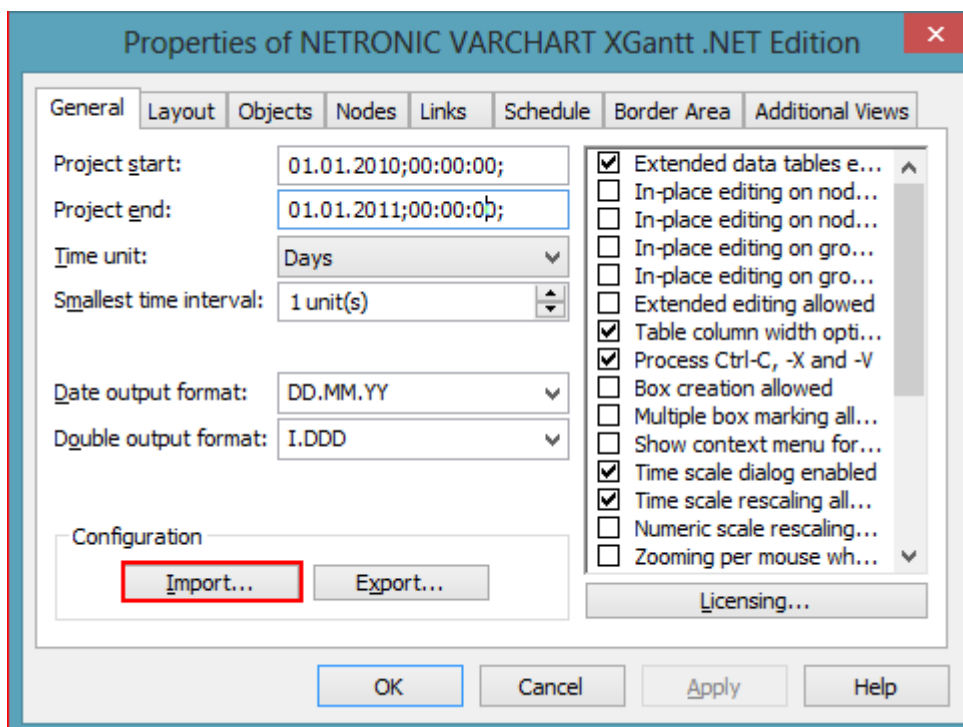


2. First, close the form and then end Visual Studio.
3. Install the new build of VARCHART XGantt .NET in the same folder as the old build.
4. Open the form designer with the form containing XGantt. The following error message will appear:

434 Frequently Asked Questions

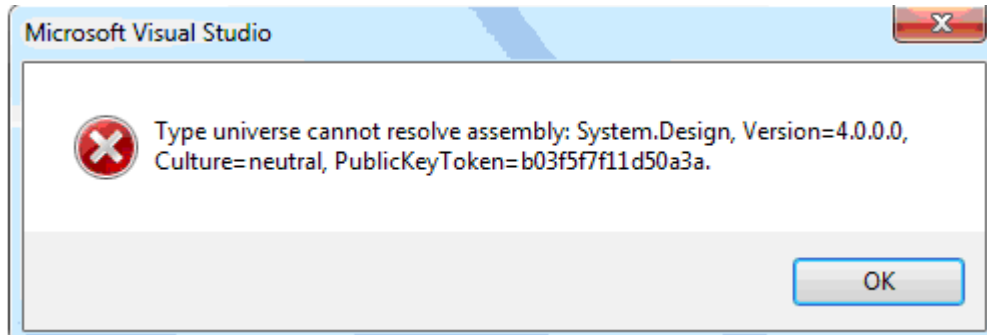


5. Click **Ignore and Continue**. The form in the form designer will be displayed correctly again but the XGantt will be set back to its default configuration.
6. Now import the configuration you have saved before by clicking the **Import** button on the **General** property page.



7. VARCHART XGantt now uses your individual configuration again.

6.3 Why does an error message occur, when I create a new project in Visual Studio 2010 and try to drag the control onto the form?



This error message occurs because in Visual Studio 2010 the **.NET Framework 4 Client Profile** is set as default but the NETRONIC VARCHART requires the target framework **.NET Framework 4** since the former lacks the System.Design.dll, which is required by the property pages at design-time. Hence you have to change the target framework from **.NET Framework Client Profile** to **.NET Framework 4** in the **Application Settings (C#)** or **Advanced Compiler Settings (VB)** **before** you drag the control onto the form.

6.4 How can I Activate the License File?

1. Please close your programming environment.
2. Copy the license file `NETRONIC.XGantt.VcGantt.lic` to the installation directory of `VARCHART XGantt.NET`.
3. Please re-start your programming environment and re-build your project again.

6.5 How can I Limit the Timescale Width?

If you touch the timescale on the extreme left side of the visible area keeping the left mouse button pressed to widen the timescale, you can easily reach a factor far in excess of 1000%. To control this, use the **VcTimeScaleSectionRescaling** event. The below example shows how to allow for a twofold enlargement at maximum.

Example Code VB.NET

```
Private Sub VcGantt1_VcTimeScaleSectionRescaling(ByVal sender As Object,
ByVal e As NETRONIC.XGantt.VcTimeScaleSectionRescalingEventArgs) Handles
VcGantt1.VcTimeScaleSectionRescaling

    Dim nOldUnitWidth As Long
    Dim returnStatus As VariantType

    nOldUnitWidth = e.TimeScale.Section(0).UnitWidth

    If (e.NewBasicUnitWidth > (2 * nOldUnitWidth)) Then
        nOldUnitWidth = 2 * nOldUnitWidth
        returnStatus = e.ReturnStatus.vcRetStatFalse
    End If

End Sub
```

Example Code C#

```
private void VcGantt1_VcTimeScaleSectionRescaling(object sender,
NETRONIC.XGantt.VcTimeScaleSectionRescalingEventArgs e)
{
    long nOldUnitWidth = e.TimeScale.get_Section(0).UnitWidth;
    object returnStatus = e.ReturnStatus;
    if (e.NewBasicUnitWidth > (2 * nOldUnitWidth))
    {
        nOldUnitWidth = 2 * nOldUnitWidth;
    }
}
```

6.6 How can I Move a Bar into the Visible Area by Clicking on the Table?

The event **VcNodeLeftClicking** captures both the node and the information **InTable** or **InDiagram**. If the table was clicked on (**InTable**), the relevant date of the node is retrieved and transferred to the VcGantt object using the **ScrollToDate** method.

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeLeftClicking(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcNodeClickingEventArgs) Handles
VcGantt1.VcNodeLeftClicking

    Dim myDataDef As VcDataDefinition
    Dim myDataDefTable As VcDataDefinitionTable
    Dim myDataField As VcDataDefinitionField
    Dim myIndex As Integer
    Dim location As VcLocation

    If (location = VcLocation.vcInTable) Then
        ' in case the Index of the "Start" field is not known
        myDataDef = VcGantt1.DataDefinition
        myDataDefTable =
myDataDef.DefinitionTable(VcDataTableType.vcMaindata)
        myDataField = myDataDefTable.DataDefinitionFieldByName("Start")
        myIndex = myDataField.ID
        VcGantt1.ScrollToDate(e.Node.DataField(myIndex),
VcHorizontalAlignment.vcLeftAligned, 2)
    End If

End Sub
```

Example Code C#

```
private void VcGantt1_VcNodeLeftClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    VcDataDefinition myDataDef = VcGantt1.DataDefinition;
    VcDataDefinitionTable myDataDefTable =
myDataDef.get_DefinitionTable(VcDataTableType.vcMaindata);
    VcDataDefinitionField myDataField =
myDataDefTable.DataDefinitionFieldByName("Start");
    int myIndex = myDataField.ID;
    VcLocation location = VcLocation.vcInTable;

    if (Location.ToString().Length > 0)
        VcGantt1.ScrollToDate(Convert.ToDateTime(e.Node.get_DataField(2)),
VcHorizontalAlignment.vcLeftAligned, 2);
}
```

6.7 How can I Make Overlapping Activities in a Group Visible?

To avoid bottlenecks in holiday rosters or for machine allocations, overlapping activities in a group can be made visible.

Activities can overlap if the activities were grouped and in the **Grouping** dialog the **Nodes in separate rows** option is **not** selected. By using the **Nodes in one line** option, the activity groups can be collapsed and expanded. If a group is collapsed, overlapping activities are invisible while in expanded groups, they are displayed as staggered piles to indicate the overlapping.

To make overlapping activities in a group visible, select the **Nodes in one line** option on the **Sorting** property page to display the activities of a group in one line. If the activities of a group overlap, they will be displayed in different lines even when the option is not activated, allowing to detect any collisions at a glance.

If the activities are collapsed, overlapping activities cannot be detected. Therefore you should deactivate the **Modifications allowed** option to prevent a user from switching between these two types of display. When the **Initially collapsed** option is *not* activated, the groups will be displayed in their expanded states, i.e. overlapping activities can be instantly recognised as they are displayed beneath each other in separate lines.

6.8 How can I Save and Reload the Order of Activities?

On condition that the activities are loaded from a file, you can save and reload the activities.

In order to save and reload the order of activities, please open the **Nodes** property page and select a data field from **Row number field**. XGantt will store the identification to this data field. If the order of the nodes was modified interactively, you can update it by using the method **UpdateRowNumberField**. It requires grouping and the hierarchy to be deactivated.

Finally, please add the below code:

Example Code VB.NET

```
Private Sub Form_Unload ()
    VcGantt1.UpdateRowNumberField
    VcGantt1.SaveAs ("file name.csv")
End Sub
```

Example Code C#

```
private void Form_Unload()
{
    VcGantt1.UpdateRowNumberFields;
    VcGantt1.SaveAs("file name.csv");
}
```

6.9 Why can I not Create Nodes Interactively at Times?

If during runtime you cannot create nodes with the mouse, please activate the check box **Allow new nodes** on the **Nodes** property page.

If in addition you tick **New nodes via double-click** you can generate nodes by double-clicking on the mouse.

Beside, if a calendar is activated, nodes cannot be generated in workfree periods.

Check if the property **NodeCreationAllowed** has not been set to **False**.

6.10 How can I Disable the Default Context Menus?

You can disable a predefined context menu to occur by setting the `returnStatus` to **`vcRetStatNoPopup`**.

Example Code VB.NET

```
'switching off the context menu of diagram
Private Sub VcGantt1_VcDiagramRightClicking(ByVal sender As Object,
ByVal e As NETRONIC.XGantt.VcDiagramClickingEventArgs) Handles
VcGantt1.VcDiagramRightClicking
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup
End Sub

'switching off the context menu of links
Private Sub VcGantt1_VcLinksRightClicking(ByVal sender As Object, ByVal
e As NETRONIC.XGantt.VcLinksClickingEventArgs) Handles
VcGantt1.VcLinksRightClicking
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup
End Sub

'switching off the context menu of nodes
Private Sub VcGantt1_VcNodeRightClicking(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcNodeClickingEventArgs) Handles
VcGantt1.VcNodeRightClicking
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup
End Sub
```

Example Code C#

```
'switching off the context menu of diagram
private void VcGantt1_VcDiagramRightClicking(object sender,
NETRONIC.XGantt.VcDiagramClickingEventArgs e)
{
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}

'switching off the context menu of linksprivate void
VcGantt1_VcLinksRightClicking(object sender,
NETRONIC.XGantt.VcLinksClickingEventArgs e)
{
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}

'switching off the context menu of nodes
private void VcGantt1_VcNodeRightClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

6.11 How can I Improve the Performance?

> Suspend update

Projects that include a large number of nodes may take too long if updating actions are repeated for each node. Not every automatic update procedure is necessary; in those cases you can suspend single updates, work off a sequence of code and then do a final update. Suspending and re-activating updates both can be done by the method **SuspendUpdate**, which is set to **True** at the beginning of the code sequence and to **False** at its end. Using this method can improve the overall performance considerably.

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection

VcGantt1.SuspendUpdate(True)

For Each dataRecord In dataRecordCltn
    dataRecord.DataField(4) = "10"
    dataRecord.Update()
Next

VcGantt1.SuspendUpdate(False)
```

Example Code C#

```
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;

vcGantt1.SuspendUpdate(true);

foreach (VcDataRecord dataRecord in dataRecordCltn)
{
    dataRecord.set_DataField(4, "10");
    dataRecord.Update();
}

vcGantt1.SuspendUpdate(false);
```

You can also accelerate the updating procedure of links via the **SuspendUpdate** method.

If you modify table formats in large projects, you also should use the **SuspendUpdate** method.

444 Frequently Asked Questions

Example Code VB.NET

```
Private Sub ModifyTable_Click()

    Dim formatCltn As VcTableFormatCollection
    Dim aFormat As VcTableFormat
    Dim index As Integer

    VcGantt1.SuspendUpdate(True)

    formatCltn = VcGantt1.LeftTable.TableFormatCollection
    For Each aFormat In formatCltn
        For index = 1 To aFormat.FormatFieldCount
            aFormat.FormatField(index).BackColor = Color.Green
            aFormat.FormatField(index).TextFontColor = Color.Red
            aFormat.FormatField(index).Alignment =
VcFormatFieldAlignment.vcFFACenter
        Next
    Next

    VcGantt1.SuspendUpdate(False)

End Sub
```

Example Code C#

```
private void ModifyTable_Click()
{
    VcTableFormatCollection formatCltn =
vcGantt1.LeftTable.TableFormatCollection;

    vcGantt1.SuspendUpdate(true);

    foreach (VcTableFormat aFormat in formatCltn)
    {
        for (int index=1; index <= aFormat.FormatFieldCount; index++)
        {
            aFormat.get_FormatField(index).BackColor = Color.Green;
            aFormat.get_FormatField(index).TextFontColor = Color.Red;
            aFormat.get_FormatField(index).Alignment =
VcFormatFieldAlignment.vcFFACenter;
        }
    }

    vcGantt1.SuspendUpdate(false);
}
```

This method also accelerates the updating procedure when you use not equidistant histogram curves.

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve
Dim index As Integer
Dim aDate As Date

histogram = VcGantt1.HistogramCollection.FirstHistogram
curve = histogram.CurveCollection.CurveByName("Curve1")
```

```

' current date
aDate = Date.Today()

VcGantt1.SuspendUpdate(True)

For index = 1 To 3000
    ' shifting by 2 hrs
    aDate = aDate.AddHours(2)
    curve.SetValues(aDate, index)
Next
VcGantt1.SuspendUpdate(False)

```

Example Code C#

```

VcHistogram histogram = vcGantt1.HistogramCollection.FirstHistogram();
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

// current date
DateTime aDate = DateTime.Today;

vcGantt1.SuspendUpdate(true);

for (int index=1; index < 3000; index++)
{
    // shifting by 2 hrs
    aDate = aDate.AddHours(2);
    curve.SetValues(aDate, Convert.ToString(index));
}

vcGantt1.SuspendUpdate(false);

```

The method also can accelerate the updating procedure when you use calendars because modifications of the calendars need a lot of time when the nodes have been loaded since then for all nodes the program has to check if they depend on a calendar.

> Graphics

Another reason for a low performance may be graphics in table, node or box fields that are too large or that have too many pixels.

6.12 What to do if the Control Does Not Work With a User Account of a Computer

If you find that the control does not react when two users invoke the same application that uses the control, the reason for this may be that the control was not installed for both users. When generating the setup program by which the control is installed on the computer of your customer, the option "install for all users" needs to be selected.

An installation for several users can be activated at a later time by extending the safety settings of the files that belong to the control, allowing different accounts to access the files. The safety settings you can modify by the menu item "properties" of the context menu of the affected file or by the command line using the command 'cacs'. You can find a list of the files that belong to the control in the chapter "Delivery" at the beginning of this book.

6.13 Can All Fonts be Used?

Due to the support of GDI+ there are some cutbacks in terms of font display. GDI+ is unable to display postscript and bitmap fonts. The first group includes fonts that may be of the type **OpenType**, but being "classical fonts" they have some sort of internal postscript structure, such as "Warnock Pro". The second group includes the early Windows fonts "Courier", "Times", "System" and "MS Sans Serif".

For this reason, the above fonts are not offered by the font selection dialogs of the VARCHART control. If you set them via the API, an alternative font will be displayed. In terms of the early fonts, NETRONIC has put up a replacement rule that selects a similar "late" font; external fonts are replaced by "Arial" to ensure a display at all.

Probably or probably not future versions of GDI+ will support the fonts presently not supported. Unfortunately, more information on this subject can only be obtained in blogs and news groups, but not at MSDN.

7 API Reference

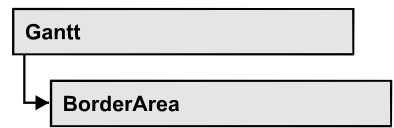
7.1 Object Types

- VcBorderArea
- VcBorderBox
- VcBox
- VcBoxCollection
- VcBoxFormat
- VcBoxFormatCollection
- VcBoxFormatField
- VcCalendar
- VcCalendarCollection
- VcCalendarGrid
- VcCalendarGridCollection
- VcCalendarProfile
- VcCalendarProfileCollection
- VcCurve
- VcCurveCollection
- VcDataDefinitionField
- VcDataDefinitionTable
- VcDataRecord
- VcDataRecordCollection
- VcDataTable
- VcDataTableCollection
- VcDataTableField
- VcDataTableFieldCollection
- VcDateLine
- VcDateLineCollection
- VcDateLineGrid
- VcDateLineGridCollection
- VcFilter
- VcFilterCollection
- VcFilterSubCondition
- VcGantt
- VcGroup

- VcGroupCollection
- VcGroupLevelLayout
- VcGroupLevelLayoutCollection
- VcHierarchyLevelLayout
- VcHistogram
- VcHistogramCollection
- VcInfoWindow
- VcInterval
- VcIntervalCollection
- VcLayer
- VcLayerCollection
- VcLayerFormat
- VcLayerFormatField
- VcLegendView
- VcLineFormat
- VcLineFormatCollection
- VcLineFormatField
- VcLink
- VcLinkAppearance
- VcLinkAppearanceCollection
- VcLinkCollection
- VcMap
- VcMapCollection
- VcMapEntry
- VcNode
- VcNodeCollection
- VcNodeLevelLayout
- VcNumericScale
- VcNumericScaleCollection
- VcPrinter
- VcRect
- VcResourceScheduler2
- VcRibbon
- VcScheduler
- VcSection
- VcTable
- VcTableCollection
- VcTableFormat
- VcTableFormatCollection
- VcTableFormatField

- VcTimeScale
- VcTimeScaleCollection
- VcUpdateBehavior
- VcUpdateBehaviorCollection
- VcUpdateBehaviorContext
- VcWorldView

7.2 VcBorderArea



An object of the type **VcBorderArea** designates the title or legend area of the graphics.

Methods

- **BorderBox**

Methods

BorderBox

Method of VcBorderArea

This method gives access to a **BorderBox** object.

	Data Type	Explanation
Parameter: boxPosition	VcBorderBoxPosition Possible Values: .vcBBXPBottomBottomCentered 8 .vcBBXPBottomBottomLeft 7 .vcBBXPBottomBottomRight 9 .vcBBXPBottomTopCentered 5 .vcBBXPBottomTopLeft 4 .vcBBXPBottomTopRight 6 .vcBBXPLegend 51 .vcBBXPTopCentered 2 .vcBBXPTopLeft 1 .vcBBXPTopRight 3	Box position second line in the bottom area, centered second line in the bottom area, left second line in the bottom area, right first line in the bottom area, centered first line in the bottom area, left first line in the bottom area, right legend top centered top left top right
Return value	VcBorderBox	Box of the title and legend area

Example Code VB.NET

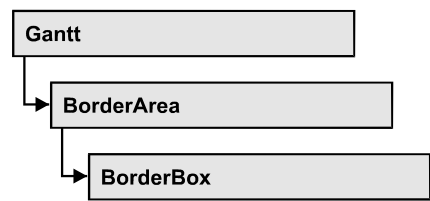
```
Dim boardArea As VcBorderArea
Dim bBoxBBL As VcBorderBox

boardArea = VcGantt1.BorderArea
bBoxBBL = boardArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft)
bBoxBBL.LegendTitle = "Explanation"
```

Example Code C#

```
VcBorderArea boardArea = vcGantt1.BorderArea;  
  
VcBorderBox bBoxBBL =  
boardArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);  
bBoxBBL.LegendTitle = "Explanation";
```

7.3 VcBorderBox



An object of the type **VcBorderBox** designates one of the boxes in the title or legend area of the graphics.

Properties

- Alignment
- GraphicsFileName
- LegendElementsArrangement
- LegendElementsBottomMargin
- LegendElementsMaximumColumnCount
- LegendElementsMaximumRowCount
- LegendElementsTopMargin
- LegendFont
- LegendTitle
- LegendTitleFont
- LegendTitleVisible
- Text
- TextFont
- Type

Properties

Alignment

Property of VcBorderBox

This property lets you set or retrieve the alignment of this BorderBox object.

	Data Type	Explanation
Property value	VcBorderBoxAlignment Possible Values: .vcBBXACentered -1 .vcBBXALeft -3	Alignment of the border box Center Left

.vcBBXARight -2	Right
-----------------	-------

GraphicsFileName

Property of VcBorderBox

This property lets you set or retrieve the name of the graphics file used in the VcBorderBox object. *Available formats:*

- *.BMP (Microsoft Windows Bitmap)
- *.EMF (Enhanced Metafile or Enhanced Metafile Plus)
- *.GIF (Graphics Interchange Format)
- *.JPG (Joint Photographic Experts Group)
- *.PNG (Portable Network Graphics)
- *.TIF (Tagged Image File Format)
- *.VMF (Viewer Metafile)
- *.WMF (Microsoft Windows Metafile, probably with EMF included)

EMF, EMF+, VMF and WMF are vector formats that allow to store a file independent of pixel resolution. All other formats are pixel-oriented and confined to a limited resolution.

The VMF format basically has been deprecated, but it will still be supported for some time to maintain compatibility with existing applications.

	Data Type	Explanation
Property value	System.String	Name of the graphics file

Example Code VB.NET

```
Dim borderArea As VcBorderArea
Dim borderBox As VcBorderBox

borderArea = VcGantt1.BorderArea
borderBox = borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomTopRight)
borderBox.Type = VcBorderBoxType.vcBBXTGraphics
borderBox.GraphicsFileName = "C:\Asterix.jpg"
```

Example Code C#

```
VcBorderArea borderArea = vcGantt1.BorderArea;
VcBorderBox borderBox =
borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomTopRight);
borderBox.Type = VcBorderBoxType.vcBBXTGraphics;
borderBox.GraphicsFileName = @"C:\Asterix.jpg";
```

LegendElementsArrangement**Property of VcBorderBox**

This property lets you set or retrieve the arrangement of the elements in the legend.

	Data Type	Explanation
Property value	VcLegendElementsArrangement	Type of arrangement of the legend elements
	Possible Values: .vcLEAFixedToColumns 0 .vcLEAFixedToRows 1 .vcLEAFixedToRowsAndColumns 2	The legend elements are merely aligned along columns. The legend elements are merely aligned along rows. The legend elements are aligned along rows and columns.

LegendElementsBottomMargin**Property of VcBorderBox**

This property lets you set or retrieve the width between the legend elements and the bottom of the border box (unit: mm).

	Data Type	Explanation
Property value	System.Int16	Width of bottom margin

LegendElementsMaximumColumnCount**Property of VcBorderBox**

This property lets you set or retrieve the number of columns to which the elements in the legend should disperse.

	Data Type	Explanation
Property value	System.Int16	Number of columns

LegendElementsMaximumRowCount

Property of VcBorderBox

This property lets you set or retrieve the number of rows to which the elements in the legend should disperse.

	Data Type	Explanation
Property value	System.Int16	Number of rows

LegendElementsTopMargin

Property of VcBorderBox

This property lets you set or retrieve the width between the legend elements and the top of the border box (unit: mm).

	Data Type	Explanation
Property value	System.Int16	Width of top margin

LegendFont

Property of VcBorderBox

This property lets you set or retrieve the font attributes of the legend.

	Data Type	Explanation
Property value	System.DrawingFont	Font attributes of the legend

Example Code VB.NET

```
Dim borderArea As VcBorderArea
Dim borderBox As VcBorderBox

borderArea = VcGantt1.BorderArea
borderBox = borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft)
borderBox.Type = VcBorderBoxType.vcBBXTLegend
MsgBox(borderBox.LegendFont.Name)
```

Example Code C#

```
VcBorderArea borderArea = vcGantt1.BorderArea;
VcBorderBox borderBox =
borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
borderBox.Type = VcBorderBoxType.vcBBXTLegend;
MessageBox.Show(borderBox.LegendFont.Name);
```

LegendTitle

Property of VcBorderBox

This property lets you set or retrieve the legend title.

	Data Type	Explanation
Property value	System.String	Legend title

Example Code VB.NET

```
Dim borderArea As VcBorderArea
Dim borderBox As VcBorderBox

borderArea = VcGantt1.BorderArea
borderBox = borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft)
borderBox.LegendTitle = "Explanation"
```

Example Code C#

```
VcBorderArea borderArea = vcGantt1.BorderArea;

VcBorderBox borderBox =
borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
borderBox.LegendTitle = "Explanation";
```

LegendTitleFont

Property of VcBorderBox

This property lets you set or retrieve the font attributes of the legend title.

	Data Type	Explanation
Property value	System.DrawingFont	Font attributes of the legend title

Example Code VB.NET

```
Dim borderArea As VcBorderArea
Dim borderBox As VcBorderBox

borderArea = VcGantt1.BorderArea
borderBox = borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft)
borderBox.Type = VcBorderBoxType.vcBBXTLegend
MsgBox(borderBox.LegendTitleFont.Name)
```

Example Code C#

```
VcBorderArea borderArea = vcGantt1.BorderArea;
VcBorderBox borderBox =
borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
borderBox.Type = VcBorderBoxType.vcBBXTLegend;
MessageBox.Show(borderBox.LegendTitleFont.Name);
```

LegendTitleVisible

Property of VcBorderBox

This property lets you set or retrieve whether the legend title is visible.

	Data Type	Explanation
Property value	System.Boolean	Legend title visible (True)/ not visible (False)

Example Code VB.NET

```
Dim borderArea As VcBorderArea
Dim borderBox As VcBorderBox

borderArea = VcGantt1.BorderArea
borderBox = borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft)
borderBox.LegendTitleVisible = False
```

Example Code C#

```
VcBorderArea borderArea = vcGantt1.BorderArea;

VcBorderBox borderBox =
borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
borderBox.LegendTitleVisible = false;
```

Text

Property of VcBorderBox

This property lets you set or retrieve the text of a head line (above or below the diagram). For numbering the pages or displaying the system date you may enter the below wild cards which will be replaced by the appropriate contents on the printout:

{COLUMN} = page number wide (of a two-dimensional page layout)

{NUMPAGES} = total number of pages

{PAGE} = consecutive numbering of pages

{ROW} = page number high (of a two-dimensional page layout)

{SYSTEMDATE} = system date

The property Text is an Indexed Property, which in C# is addressed by the methods set_Text (rowIndex, pvn) and get_Text (rowIndex).

	Data Type	Explanation
Parameter: rowIndex	System.Int16	Row index {0...6}
Property value	System.String	Text in text boxes

Example Code VB.NET

```

Dim borderArea As VcBorderArea
Dim borderBox As VcBorderBox

borderArea = VcGantt1.BorderArea
borderBox = borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft)
borderBox.Type = VcBorderBoxType.vcBBXTText
borderBox.Text(index) = "Department A"

```

Example Code C#

```

VcBorderArea borderArea = vcGantt1.BorderArea;

VcBorderBox borderBox =
borderArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
borderBox.Type = VcBorderBoxType.vcBBXTText;
borderBox.set_Text(index, "DepartmentA");

```

TextFont**Property of VcBorderBox**

This property lets you set or retrieve the font attributes of a title line (above or below the diagram).

This property is an indexed property, which in C# is referred to by one of the methods **set_TextFont (rowIndex, pvn)** and **get_TextFont (row-Index)**.

The property TextFont is an Indexed Property, which in C# is addressed by the methods **set_TextFont (rowIndex, pvn)** and **get_TextFont (rowIndex)**.

	Data Type	Explanation
Parameter: rowIndex	System.Int16	Row index {0...6}
Property value	System.DrawingFont	Font attributes of the text

Example Code VB.NET

```
Dim borderArea As VcBorderArea
Dim bBoxTL As VcBorderBox

Set borderArea = VcGantt1.BorderArea
Set bBoxBBL = borderArea.BorderBox(vcBBXPBottomBottomLeft)

bBoxTL.TextFont(i).Bold = False
bBoxTL.TextFont(i).Italic = False
bBoxTL.TextFont(i).Name = "Symbol"
```

Example Code C#

```
// Text for Title
VcBorderBox borderBox =
VcGantt1.BorderArea.BorderBox(VcBorderBoxPosition.vcBBXPTopCentered);
borderBox.Type = VcBorderBoxType.vcBBXTText;

Font titleFont1 = new Font("Arial", 20, FontStyle.Bold);

borderBox.set_Text(1, "Time Scheduler");
borderBox.set_TextFont(1, titleFont1);
```

Type

Property of VcBorderBox

This property lets you set or retrieve the type of the BorderBox object.

	Data Type	Explanation
Property value	VcBorderBoxType	Box type
	Possible Values: .vcBBXTGraphics 3 .vcBBXTLegend 4 .vcBBXTNothing 0 .vcBBXTText 1 .vcBBXTTextWithGraphics 2	graphics legend nothing text text and graphics

Example Code VB.NET

```
Dim bBoxBBL As VcBorderBox

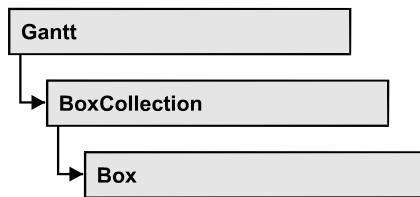
bBoxBBL = boardArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomLeft)
bBoxBBL.Type = VcBorderBoxType.vcBBXTGraphics
```

Example Code C#

```
VcBorderArea boardArea = vcGantt1.BorderArea;

VcBorderBox bBoxBBL =
boardArea.BorderBox(VcBorderBoxPosition.vcBBXPBottomBottomLeft);
bBoxBBL.Type = VcBorderBoxType.vcBBXTGraphics;
```

7.4 VcBox



An object of the type **VcBox** designates a box to display texts or graphics.

Properties

- AnchoringInteractionsAllowed
- AnchoringLineVisible
- FieldText
- FormatName
- LineColor
- LineThickness
- LineType
- Marked
- Moveable
- Name
- NodeID
- Origin
- Priority
- ReferencePoint
- Resizing
- Specification
- UpdateBehaviorName
- Visible

Methods

- AnchorToNode
- GetActualExtent
- GetTopLeftPixel
- GetXYOffset
- IdentifyFormatField
- SetXYOffset
- SetXYOffsetByTopLeftPixel

Properties

AnchoringInteractionsAllowed

Property of VcBox

By this property you can set or retrieve whether a box can be tied to a node interactively .

	Data Type	Explanation
Property value	System.Boolean	Box can/cannot be tied to a node interactively Default value: False

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.AnchoringInteractionsAllowed = False
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.AnchoringInteractionsAllowed = false;
```

AnchoringLineVisible

Property of VcBox

By this property you can set or retrieve whether a line is shown between the specified reference points if the box is tied to a node.

	Data Type	Explanation
Property value	System.Boolean	Anchoring line between node and box is/is not shown Default value: False

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.AnchoringLineVisible = False
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.AnchoringLineVisible = false;
```

FieldText**Property of VcBox**

This property lets you set or retrieve the contents of a box field. You also can specify the offset in the **Edit Box** dialog box.

If a text field contains more than one line, you can use "\n" in the text string to separate two lines of the text field (Example: "Line1\nLine2"). Otherwise the lines will be separated at blanks.

The property FieldText is an Indexed Property, which in C# is addressed by the methods set_FieldText (fieldIndex, pvn) and get_FieldText (fieldIndex).

	Data Type	Explanation
Parameter: ⇒ fieldIndex	System.Int16	Field index
Property value	System.String	Field content

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.FieldText(0) = "User: "
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.set_FieldText(0, "User: ");
```

FormatName**Property of VcBox**

This property lets you set or retrieve the name of the box format.

	Data Type	Explanation
Property value	VcBoxFormat	BoxFormat object or Nothing

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.FormatName = "Standard"
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.FormatName = "Standard";
```

LineColor

Property of VcBox

This property lets you set or retrieve the color of the border line of the box.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255})

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.LineColor = System.Drawing.Color.Blue
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.LineColor = System.Drawing.Color.Blue;
```

LineThickness

Property of VcBox

This property lets you set or retrieve the line thickness of the border line of the box.

If you set this property to values between 1 and 4, an absolute line thickness is defined in pixels. Irrespective of the zoom factor a line will always show the same line thickness in pixels. When printing though, the line thickness is adapted for the sake of legibility and becomes dependent of the zoom factor:

Value	Points	mm
1	1/2 point	0.09 mm
2	1 point	0.18 mm
3	3/2 points	0.26 mm
4	2 points	0.35 mm

A point equals 1/72 inch and represents the unit of the font size.

If you set this property to values between 5 and 1,000, the line thickness is defined in 1/100 mm, so the lines will be displayed in a true thickness in pixels that depends on the zoom factor.

	Data Type	Explanation
Property value	System.Int16	Line thickness LineType {1...4}: line thickness in pixels LineType {5...1000}: line thickness in 1/100 mm Default value: As defined in the dialog

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.LineThickness = 2
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.LineThickness = 2;
```

LineType

Property of VcBox

This property lets you set or retrieve the type of the border line of the box.

	Data Type	Explanation
Property value	VcLineType	Line type Default value: vcSolid
	Possible Values: .vcDashed 4 .vcDashed 4 .vcDashedDotted 5	Line dashed Line dashed Line dashed-dotted

.vcDashedDotted 5	Line dashed-dotted
.vcDotted 3	Line dotted
.vcDotted 3	Line dotted
.vcLineType0 100	Line Type 0

.vcLineType1 101	Line Type 1

.vcLineType10 110	Line Type 10

.vcLineType11 111	Line Type 11

.vcLineType12 112	Line Type 12

.vcLineType13 113	Line Type 13

.vcLineType14 114	Line Type 14

.vcLineType15 115	Line Type 15

.vcLineType16 116	Line Type 16

.vcLineType17 117	Line Type 17

.vcLineType18 118	Line Type 18

.vcLineType2 102	Line Type 2

.vcLineType3 103	Line Type 3

.vcLineType4 104	Line Type 4

.vcLineType5 105	Line Type 5

.vcLineType6 106	Line Type 6

.vcLineType7 107	Line Type 7

.vcLineType8 108	Line Type 8

.vcLineType9 109	Line Type 9

.vcNone 1	No line type assigned
.vcNone 1	No line type
.vcSolid 2	Line solid
.vcSolid 2	Line solid

Example Code VB.NET

```

Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.LineType = VcLineType.vcDotted

```

Example Code C#

```

VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.LineType = VcLineType.vcDotted;

```


Marked

Property of VcBox

This property lets you set or retrieve whether a text box is marked.

	Data Type	Explanation
Property value	System.Boolean	True: box marked; false: box unmarked

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.Marked = True
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.Marked = true;
```

Moveable

Property of VcBox

This property lets you set or retrieve whether the box can be moved interactively.

	Data Type	Explanation
Property value	System.Boolean	Movable (True)/ not Movable (False) Default value: True

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.Moveable = False
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.Moveable = false;
```

Name

Property of VcBox

This property lets you set or retrieve the name of a box. You can also specify the name in the **Administrate Boxes** dialog box.

	Data Type	Explanation
Property value	System.String	Box name

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox
Dim boxName As String

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
MsgBox(box.Name)
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();

MessageBox.Show(box.Name);
```

NodeID

Property of VcBox

By this property you can set or retrieve the node ID of the node which the box is tied to. You can also specify the Node-ID in the **Administrate Boxes** dialog box.

	Data Type	Explanation
Property value	System.String	ID of the node the box is tied to

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox
Dim box.NodeID As String

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
boxNodeID = 3
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();

boxNodeID = 3;
```

Origin

Property of VcBox

This property lets you set or retrieve the origin of the box, i. e. the point of the diagram from which the offset to the reference point of the box will be measured.

With the help of the properties **Origin**, **ReferencePoint** and the method **GetXYOffset** you can position each box in the diagram area. The relative position of the boxes is independent of the current diagram size.

	Data Type	Explanation
Property value	VcBoxOrigin	Origin of the box
	Possible Values: .vcBOBottomCenter 28 .vcBOBottomLeft 27 .vcBOBottomRight 29 .vcBOCenterCenter 25 .vcBOCenterLeft 24 .vcBOCenterRight 26 .vcBOTopCenter 22 .vcBOTopLeft 21 .vcBOTopRight 23	bottom center bottom left bottom right center center center left center right top center top left top right

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.Origin = VcBoxOrigin.vcBOTopCenter
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.Origin = VcBoxOrigin.vcBOTopCenter;
```

Priority

Property of VcBox

This property lets you set or retrieve the priority of the box.

	Data Type	Explanation
Property value	System.Int16	Priority value

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.Priority = 3
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.Priority = 3;
```

ReferencePoint

Property of VcBox

This property lets you set or retrieve the reference point of the box, i. e. the point of the box from which the offset to the origin will be measured.

	Data Type	Explanation
Property value	VcBoxReferencePoint	Reference point of the box
	Possible Values:	
	.vcBRPBottomCenter 28	bottom center
	.vcBRPBottomLeft 27	bottom left
	.vcBRPBottomRight 29	bottom right
	.vcBRPCenterCenter 25	center center
	.vcBRPCenterLeft 24	center left
	.vcBRPCenterRight 26	center right
	.vcBRPTopCenter 22	top center
	.vcBRPTopLeft 21	top left
	.vcBRPTopRight 23	top right

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.ReferencePoint = VcBoxReferencePoint.vcBRPCenterRight
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.ReferencePoint = VcBoxReferencePoint.vcBRPCenterRight;
```

Resizing

Property of VcBox

This property lets you set or retrieve whether and how the size of a box can be modified.

	Data Type	Explanation
Property value	VcBoxResizing Possible Values: .vcBRHeight 23 .vcBRNo 0 .vcBRWidth 24 .vcBRWidth/Height 1050	Interactive modification of the size of the box The height of the box can be modified interactively. The size of the box cannot be modified interactively. The width of the box can be modified interactively. Width and height of the box can be modified interactively.

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.Resizing = VcBoxRResizing.vcBRWidth
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.Resizing = VcBoxRResizing.vcBRWidth;
```

Specification**Read Only Property of VcBox**

This property lets you retrieve the specification of a box. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored without problems to text files or data bases. This allows for persistency. A specification can be used to create a box by the method **VcBoxCollection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the box

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
MsgBox(box.Specification)
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
MessageBox.Show(box.Specification);
```

UpdateBehaviorName

Property of VcBox

This property lets you set or retrieve the name of the UpdateBehavior.

	Data Type	Explanation
Property value	System.String	Name of the UpdateBehavior

Visible

Property of VcBox

This property lets you set or retrieve whether a box is visible. You also can specify this property in the **Administrate Boxes** dialog box.

	Data Type	Explanation
Property value	System.Boolean	Box visible/invisible Default value: True

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
box.Visible = False
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
box.Visible = false;
```

Methods

AnchorToNode

Method of VcBox

This method lets you tie boxes to nodes or untie them again. An anchored box can be still moved (provided that you have set the property **Moveable**). To untie a box from the node, you have to pass "NULL" as parameter.

If you move a node which is anchored to a box, the box is moved as well. If the node is collapsed, the box is collapsed as well, thus becoming invisible. When the node is expanded the box is visible again.

If a box is tied to a node, its position on the screen will be maintained. The offset values which are used as basis are converted according to the reference points (Origin, ReferencePoint). If, for example, a box with a certain offset refers to a chart at the top left (origin) and then is anchored to a node, an offset to the the top left node is calculated automatically. This makes sure that the position on the screen will not be altered. If the box is untied from the node the calculation is carried out backwards.

	Data Type	Explanation
Parameter: node	VcNode	Node object to which the box is tied
Return value	System.Boolean	Box is anchored to node/untied from node

Example Code VB.NET

```
VcNode node = VcGantt1.NodeCollection.FirstNode()
VcGantt1.BoxCollection.FirstBox().AnchorToNode(node)
```

Example Code C#

```
VcNode node = vcGantt1.NodeCollection.FirstNode();
vcGantt1.BoxCollection.FirstBox().AnchorToNode(node);
```

GetActualExtent

Method of VcBox

This method lets you retrieve the actual extent of the box (unit: 1/100 mm).

By regarding these values when setting the XY offset, you can modify the reference point of the anchoring line without changing the position of the box.

	Data Type	Explanation
Parameter: ↔ width	System.Int32	width of the box
↔ height	System.Int32	height of the box
Return value	System.Boolean	Extent of the box is returned/not returned

GetTopLeftPixel

Method of VcBox

This method lets you convert to pixel and display the saved XY offset for the top left corner.

The x value can be further used with the method **VcGantt.GetDate** for instance to get a date.

	Data Type	Explanation
Parameter:		
⇐ x	System.Int32	X value of the offset
⇐ y	System.Int32	Y value of the offset
Return value	System.Boolean	Offset is returned/not returned

GetXyOffset

Method of VcBox

This method lets you retrieve the distance between origin and reference point in x and y direction (unit: 1/100 mm).

	Data Type	Explanation
Parameter:		
⇐ xOffset	System.Int32	X value of the offset
⇐ yOffset	System.Int32	Y value of the offset
Return value	System.Boolean	Offset is returned/not returned

IdentifyFormatField

Method of VcBox

This method lets you retrieve the index of the format field at the specified position. If there is a field at the position specified, **True** will be returned, if there isn't, the method will deliver **False**.

	Data Type	Explanation
Parameter:		
⇒ x	System.Int32	X coordinate of the position
⇒ y	System.Int32	Y coordinate of the position

⇐ format	VcBoxFormat	Identified format
⇐ formatFieldIndex	System.Int16	Index of the format field
Return value	System.Boolean	A format field exists/does not exist at the position specified

SetXYOffset

Method of VcBox

This method lets you specify the distance between origin and reference point in x and y direction (unit: 1/100 mm).

You also can specify the offset in the **Administrate Boxes** dialog box.

	Data Type	Explanation
Parameter:		
⇒ xOffset	System.Int32	X value of the offset
⇒ yOffset	System.Int32	Y value of the offset
Return value	System.Boolean	Offset is set (True)/not set (False)

Example Code VB.NET

```
Dim offSet As Boolean
offSet = VcGantt1.BoxCollection.FirstBox.SetXYOffset(100, 100)
```

Example Code C#

```
bool offSet = vcGantt1.BoxCollection.FirstBox().SetXYOffset(100, 100);
```

SetXYOffsetByTopLeftPixel

Method of VcBox

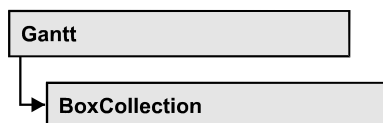
This method lets you internally convert the specified pixel value of the top left corner to an XY offset and then save the offset.

This enables you for instance to place a box at an XY coordinate from an event.

	Data Type	Explanation
Parameter:		
⇒ x	System.Int32	X value of the offset
⇒ y	System.Int32	Y value of the offset

Return value	System.Boolean	Offset is set (True) / not set (False)
--------------	----------------	--

7.5 VcBoxCollection



The VcBoxCollection object contains all boxes available. You can access all objects in an iterative loop by **For Each box In VcBoxCollection** or by the methods **First...** and **Next...**. You can access a single box by the method **BoxByName**. The number of boxes in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the boxes in the corresponding way.

Properties

- Count

Methods

- Add
- AddBySpecification
- BoxByIndex
- BoxByName
- Copy
- FirstBox
- GetEnumerator
- NextBox
- Remove
- Update

Properties

Count

Read Only Property of VcBoxCollection

This property lets you retrieve the number of boxes in the box collection.

	Data Type	Explanation
Property value	System.Int32	Number of boxes

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim numberOfBoxes As Integer

boxCltn = VcGantt1.BoxCollection
numberOfBoxes = boxCltn.Count
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
int numberOfBoxes = boxCltn.Count;
```

Methods

Add

Method of VcBoxCollection

By this method you can create a box as a member of the BoxCollection. If the name has not been used before, the new box object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned. To make the new box visible in the diagram, the box collection needs to be updated by the **Update** call.

	Data Type	Explanation
Parameter: ⇒ boxName	System.String	Box name
Return value	VcBox	New box object

Example Code VB.NET

```
newBox = VcGantt1.BoxCollection.Add("box1")
```

Example Code C#

```
newBox = vcGantt1.BoxCollection.Add("box1");
```

AddBySpecification

Method of VcBoxCollection

This method lets you create a box by using by a box specification. This way you can keep a box persistent. This way of creating allows box objects to become persistent. The specification of a box can be saved and re-loaded (see VcBox property **Specification**). In a subsequent the box can be created can be created again from the specification and is identified by its name. To make

the new box visible in the diagram, the box collection needs to be updated by the **Update** call.

	Data Type	Explanation
Parameter: ⇒ specification	System.String	Box specification
Return value	VcBox	New box object

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection

boxCltn = VcGantt1.BoxCollection
boxCltn.AddBySpecification(textSpecification)
boxCltn.Update()
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
boxCltn.AddBySpecification(textSpecification);
boxCltn.Update();
```

BoxByIndex

Method of VcBoxCollection

This method lets you access a box by its index. If a box does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the box
Return value	VcBox	Box object returned

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection

boxCltn = VcGantt1.BoxCollection
box = boxCltn.BoxByIndex(0)
box.LineThickness = 2
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.BoxByIndex(0);
box.LineThickness = 2;
```

BoxByName

Method of VcBoxCollection

By this method you can retrieve a box by its name. If a box of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ boxName	System.String	Box name
Return value	VcBox	Box

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection

boxCltn = VcGantt1.BoxCollection
box = boxCltn.BoxByName("BoxOne")
box.LineThickness = 3
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.BoxByName("BoxOne");
box.LineThickness = 3;
```

Copy

Method of VcBoxCollection

By this method you can copy a box. If the box that is to be copied exists, and if the name for the new box does not yet exist, the new box object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned. To make the copied box visible in the diagram, the box collection needs to be updated by the **Update** call.

	Data Type	Explanation
Parameter: ⇒ boxName	System.String	Name of the box to be copied
⇒ newBoxName	System.String	Name of the new box
Return value	VcBox	Box object

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection

boxCltn = VcGantt1.BoxCollection
boxCltn.Copy("BoxOne", "NewBox")
boxCltn.Update()
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
boxCltn.Copy("BoxOne", "NewBox");
boxCltn.Update();
```

FirstBox**Method of VcBoxCollection**

This method can be used to access the initial value, i.e. the first box of a box collection, and then to continue in a forward iteration loop by the method **NextBox** for the boxes following. If there is no box in the BoxCollection object, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcBox	First box

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
```

GetEnumerator**Method of VcBoxCollection**

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the box objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

Example Code VB.NET

```
Dim box As VcBox

For Each box In VcGantt1.BoxCollection
    ListBox1.Items.Add(box.FormatName)
Next
```

Example Code C#

```
foreach (VcBox box in vcGantt1.BoxCollection)
    listBox1.Items.Add(box.FormatName);
```

NextBox

Method of VcBoxCollection

This method can be used in a forward iteration loop to retrieve subsequent boxes from a box collection after initializing the loop by the method **FirstBox**. If there is no box left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcBox	Succeeding box

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox

While Not box Is Nothing
    ListBox1.Items.Add(box.Name)
    box = boxCltn.NextBox
End While
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();

while (box != null)
{
    ListBox.Items.Add(box.Name);
    box = boxCltn.NextBox();
}
```

Remove

Method of VcBoxCollection

This method lets you delete a box. To make the deletion visible in the diagram, the box collection needs to be updated by the **Update** call.

484 API Reference: VcBoxCollection

	Data Type	Explanation
Parameter: ⇒ boxName	System.String	Box name
Return value	System.Boolean	Box deleted (True)/not deleted (False)

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.BoxByIndex(0)
boxCltn.Remove(box.Name)
boxCltn.Update()
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.BoxByIndex(0);
boxCltn.Remove(box.Name);
boxCltn.Update();
```

Update

Method of VcBoxCollection

This method lets you update a box collection after having modified it.

	Data Type	Explanation
Return value	System.Boolean	Update successful (True)/ not successful (False)

Example Code VB.NET

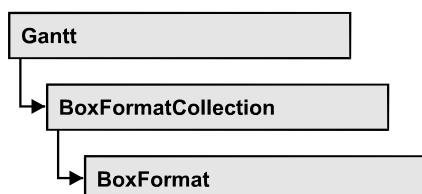
```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.BoxByIndex(0)
boxCltn.Remove(box.Name)
boxCltn.Update()
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.BoxByIndex(0);
boxCltn.Remove(box.Name);
boxCltn.Update();
```

7.6 VcBoxFormat



An object of the type **VcBoxFormat** defines the formats of boxes. With **For Each formatField In BoxFormat** you can retrieve all box formats

Properties

- FieldsSeparatedByLines
- FormatField
- FormatFieldCount
- Name
- Specification

Methods

- CopyFormatField
- GetEnumerator
- RemoveFormatField

Properties

FieldsSeparatedByLines

Property of VcBoxFormat

This property lets you set or retrieve whether fields are to be separated by lines.

	Data Type	Explanation
Property value	System.Boolean	Box fields separated by lines (True)/ not separated by lines (False).

Example Code VB.NET

```

Dim boxFormat As VcBoxFormat

boxFormat = VcGantt1.BoxFormatCollection.FormatByIndex(0)
boxFormat.FieldsSeparatedByLines = True
  
```

Example Code C#

```
VcBoxFormat boxFormat = vcGantt1.BoxFormatCollection.FormatByIndex(0);
boxFormat.FieldsSeparatedByLines = true;
```

FormatField**Read Only Property of VcBoxFormat**

This property gives access to a VcBoxFormatField object by its index. The index has to be in the range from 0 to FormatFieldCount-1.

The property FormatField is an Indexed Property, which in C# is addressed by the method `get_FormatField(index)`.

	Data Type	Explanation
Parameter: index	System.Int16	Index of the box format field
Property value	VcBoxFormatField	Nox format field

Example Code VB.NET

```
Dim boxFormat As VcBoxFormat
Dim formatField As VcBoxFormatField

boxFormat = VcGantt1.BoxFormatCollection.FirstFormat
formatField = boxFormat.FormatField(0)
MsgBox(formatField.FormatName)
```

Example Code C#

```
VcBoxFormat boxFormat = vcGantt1.BoxFormatCollection.FirstFormat();
VcBoxFormatField formatField = boxFormat.get_FormatField(0);
MessageBox.Show(formatField.FormatName);
```

FormatFieldCount**Read Only Property of VcBoxFormat**

This property allows to determine the number of fields in a box format.

	Data Type	Explanation
Property value	System.Int16	Number of fields of the box format

Example Code VB.NET

```
Dim boxFormat As VcBoxFormat
Dim formatField As VcBoxFormatField

boxFormat = VcGantt1.BoxFormatCollection.FirstFormat
MsgBox(boxFormat.FormatFieldCount)
```

Example Code C#

```
VcBoxFormat boxFormat = vcGantt1.BoxFormatCollection.FirstFormat();
MessageBox.Show(boxFormat.FormatFieldCount.ToString());
```

Name**Property of VcBoxFormat**

This property lets you retrieve/set the name of a box format. You can also specify the name in the **Administrative Box Formats** dialog box.

	Data Type	Explanation
Property value	System.String	Box format name

Example Code VB.NET

```
Dim boxFormat As VcBoxFormat

For Each boxFormat In VcGantt1.BoxFormatCollection
    ListBox1.Items.Add(boxFormat.Name)
Next
```

Example Code C#

```
foreach (VcBoxFormat boxFormat in vcGantt1.BoxFormatCollection)
    listBox1.Items.Add(boxFormat.Name);
```

Specification**Read Only Property of VcBoxFormat**

This property lets you retrieve the specification of a box format. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored without problems to text files or data bases. This allows for persistency. A specification can be used to create a box format by the method **VcBoxFormatCollection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the box format

Example Code VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormat As VcBoxFormat

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormat = boxFormatCltn.FirstBoxFormat
MsgBox(boxFormat.Specification)
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormat boxFormat = boxFormatCltn.FirstBoxFormat();
MessageBox.Show(boxFormat.Specification);
```

Methods

CopyFormatField

Method of VcBoxFormat

This method allows to copy a box format field. The new VcBoxFormatField object is returned. It is given automatically the next index not used before.

	Data Type	Explanation
Parameter:		
⇒ position	VcFormatFieldInnerPosition	Position of the new box format field
	Possible Values: .vcInnerAbove 1 .vcInnerBelow 3 .vcInnerLeftOf 0 .vcInnerRightOf 4	above below left of right of
⇒ refIndex	System.Int16	Index of the reference box format field
Return value	VcBoxFormatField	Box format field object

Example Code VB.NET

```
Dim boxFormat As VcBoxFormat
Dim formatField As VcBoxFormatField

boxFormat = VcGantt1.BoxFormatCollection.FormatByIndex(2)
boxFormat.CopyFormatField(VcFormatFieldInnerPosition.vcInnerRightOf, 0)
```

Example Code C#

```
VcBoxFormat boxFormat = vcGantt1.BoxFormatCollection.FormatByIndex(0);
VcBoxFormatField formatField =
boxFormat.CopyFormatField(VcFormatFieldInnerPosition.vcInnerRightOf, 0);
```

GetEnumerator

Method of VcBoxFormat

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the box format fields included.

	Data Type	Explanation
Return value	VcObject	Reference object

Example Code VB.NET

```

Dim boxFormat As VcBoxFormat
Dim formatField As VcBoxFormatField

boxFormat = VcGantt1.BoxFormatCollection.FirstFormat
For Each formatField In boxFormat
    ListBox1.Items.Add(formatField.FormatName)
Next

```

Example Code C#

```

VcBoxFormat boxFormat = vcGantt1.BoxFormatCollection.FirstFormat();
foreach(VcBoxFormatField formatField in boxFormat)
    listBox1.Items.Add(formatField.FormatName);

```

RemoveFormatField**Method of VcBoxFormat**

This method lets you remove a box format field by its index. After that, the program will set all box format field indexes newly in order to number them consecutively.

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the box format field to be deleted

Example Code VB.NET

```

Dim boxFormat As VcBoxFormat
Dim i As Integer

boxFormat = VcGantt1.BoxFormatCollection.FirstFormat

For i = 0 To boxFormat.FormatFieldCount - 1
    boxFormat.RemoveFormatField(i)
Next

```

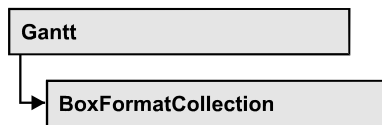
Example Code C#

```

VcBoxFormat boxFormat = vcGantt1.BoxFormatCollection.FirstFormat();
for (short i=0; i<boxFormat.FormatFieldCount-1; i++)
    boxFormat.RemoveFormatField(i);

```

7.7 VcBoxFormatCollection



The VcBoxFormatCollection object contains all box formats available. You can access all objects in an iterative loop by **For Each boxFormat In BoxFormatCollection** or by the methods **First...** and **Next...**. You can access a single box format by the method **BoxFormatByName**. The number of box formats in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the box formats in the corresponding way.

Properties

- Count

Methods

- Add
- AddBySpecification
- Copy
- FirstFormat
- FormatByIndex
- FormatByName
- GetEnumerator
- NextFormat
- Remove

Properties

Count

Read Only Property of VcBoxFormatCollection

This property lets you retrieve the number of box formats in the box format collection.

	Data Type	Explanation
Property value	System.Int32	Number of box formats

Example Code VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim numberOfBoxformats As Integer

boxFormatCltn = VcGantt1.BoxFormatCollection
numberOfBoxformats = boxFormatCltn.Count
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
int numberOfBoxformats = boxFormatCltn.Count;
```

Methods

Add

Method of VcBoxFormatCollection

By this method you can create a box format as a member of the BoxFormatCollection. If the name has not been used before, the new box object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ formatName	System.String	Box format name
Return value	VcBoxFormat	New box format object

Example Code VB.NET

```
Dim newBoxFormat = VcGantt1.BoxFormatCollection.Add("boxFormat1")
```

Example Code C#

```
newBoxFormat = vcGantt1.BoxFormatCollection.Add("boxFormat1");
```

AddBySpecification

Method of VcBoxFormatCollection

This method lets you create a box format by using a box format specification. This way of creating allows box format objects to become persistent. The specification of a box format can be saved and re-loaded (see VcBoxFormat property **Specification**). In a subsequent session the box format can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter: ⇒ formatSpecification	System.String	Box format specification
Return value	VcBoxFormat	New box format object

Copy

Method of VcBoxFormatCollection

By this method you can copy a box format. If the box format that is to be copied exists, and if the name for the new box format does not yet exist, the new box format object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ FormatName	System.String	Name of the box format to be copied
⇒ newFormatName	System.String	Name of the new box format
Return value	VcBoxFormat	Box format object

Example Code VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormat As VcBoxFormat

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormat = boxFormatCltn.Copy("CurrentBoxFormat", "NewBoxFormat")
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormat boxFormat = boxFormatCltn.Copy("CurrentBoxFormat", "NewBoxFormat");
```

FirstFormat

Method of VcBoxFormatCollection

This method can be used to access the initial value, i.e. the first box format of a box format collection and then to continue in a forward iteration loop by the method **NextFormat** for the box formats following. If there is no box format in the box format collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcBoxFormat	First box format

Example Code VB.NET

```
Dim format As VcBoxFormat

format = VcGantt1.BoxFormatCollection.FirstFormat
```

Example Code C#

```
VcBoxFormat format = vcGantt1.BoxFormatCollection.FirstFormat();
```

FormatByIndex

Method of VcBoxFormatCollection

This method lets you access a box format by its index. If a box format does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the box format
Return value	VcBoxFormat	Box format object returned

Example Code VB.NET

```
Dim formatBoxCltn As VcBoxFormatCollection
Dim formatBox As VcBoxFormat

formatBoxCltn = VcGantt1.BoxFormatCollection
formatBox = formatBoxCltn.FormatByIndex(2)
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormat format = boxFormatCltn.FormatByIndex(2);
```

FormatByName

Method of VcBoxFormatCollection

By this method you can retrieve a box format by its name. If a box format of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ formatName	System.String	Name of the box format
Return value	VcBoxFormat	Box format

Example Code VB.NET

```
Dim formatBoxCltn As VcBoxFormatCollection
Dim formatBox As VcBoxFormat

formatBoxCltn = VcGantt1.BoxFormatCollection
formatBox = formatBoxCltn.FormatByName("Standard")
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormat format = boxFormatCltn.FormatByName("Standard");
```

GetEnumerator**Method of VcBoxFormatCollection**

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the box format objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

Example Code VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormat As VcBoxFormat

boxFormatCltn = VcGantt1.BoxFormatCollection
For Each boxFormat In boxFormatCltn
    ListBox1.Items.Add(boxFormat.Name)
Next
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
foreach (VcBoxFormat boxFormat in boxFormatCltn)
    listBox1.Items.Add(boxFormat.Name);
```

NextFormat**Method of VcBoxFormatCollection**

This method can be used in a forward iteration loop to retrieve subsequent box formats from a box format collection after initializing the loop by the

method **FirstFormat**. If there is no format left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcBoxFormat	Subsequent box format

Example Code VB.NET

```
Dim formatBoxCltn As VcBoxFormatCollection
Dim formatBox As VcBoxFormat

formatBoxCltn = VcGantt1.BoxFormatCollection
formatBox = formatBoxCltn.FirstFormat

While Not formatBox Is Nothing
    ListBox1.Items.Add(formatBox.Name)
    formatBox = formatBoxCltn.NextFormat
End While
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormat boxFormat = boxFormatCltn.FirstFormat();

while (boxFormat != null)
{
    ListBox.Items.Add(boxFormat.Name);
    boxFormat = boxFormatCltn.NextFormat();
}
```

Remove

Method of VcBoxFormatCollection

This method lets you delete a box format. If the box format is used in another object, it cannot be deleted. Then False will be returned, otherwise True.

	Data Type	Explanation
Parameter: ⇒ FormatName	System.String	Box format name
Return value	System.Boolean	Box format deleted (True)/not deleted (False)

Example Code VB.NET

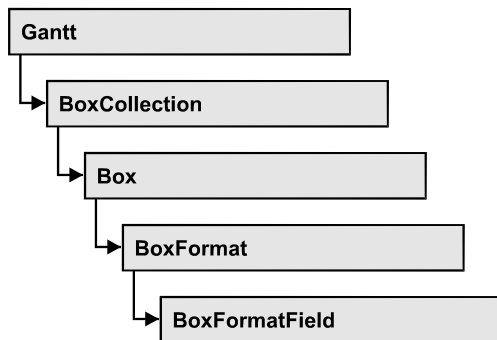
```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormat As VcBoxFormat

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormat = boxFormatCltn.FormatByIndex(1)
boxFormatCltn.Remove(boxFormat.Name)
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;  
VcBoxFormat boxFormat = boxFormatCltn.FormatByIndex(1);  
boxFormatCltn.Remove(boxFormat.Name);
```

7.8 VcBoxFormatField



An object of the type **VcBoxFormat** represents a field of a VcBoxFormat-Object. A box format field does not have a name as many other objects, but it has an index that defines its position in the box format.

Properties

- Alignment
- FormatName
- GraphicsHeight
- Index
- MaximumTextLineCount
- MinimumTextLineCount
- MinimumWidth
- PatternBackgroundColor
- PatternColorAsARGB
- TextFont
- TextFontColor
- Type

Properties

Alignment

Property of VcBoxFormatField

This property lets you set or retrieve the alignment of the content of the box format field.

	Data Type	Explanation
Property value	VcFormatFieldAlignment	Alignment of the field content
	Possible Values: .vcFFABottom 28 .vcFFABottomLeft 27 .vcFFABottomRight 29 .vcFFACenter 25 .vcFFALeft 24 .vcFFARight 26 .vcFFATop 22 .vcFFATopLeft 21 .vcFFATopRight 23	Bottom Bottom left Bottom right Center Left Right Top Top left Top right

Example Code VB.NET

```

Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.Alignment = VcFormatFieldAlignment.vcFFACenter

```

Example Code C#

```

VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.Alignment = VcFormatFieldAlignment.vcFFACenter;

```

FormatName**Read Only Property of VcBoxFormatField**

This property lets you retrieve the name of the box format to which this field belongs.

	Data Type	Explanation
Property value	System.String	Name of the box format

Example Code VB.NET

```

Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
MsgBox(boxFormatField.FormatName)

```

Example Code C#

```

VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
MessageBox.Show(boxFormatField.FormatName);

```

GraphicsHeight

Property of VcBoxFormatField

This property lets you set or retrieve for the type **vcFFTGraphics** the height of the graphics in the box format field.

	Data Type	Explanation
Property value	System.Int16	Height (in mm) of the graphics 0...200

Example Code VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.Type = VcFormatFieldType.vcFFTGraphics
boxFormatField.GraphicsHeight = 150
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.Type = VcFormatFieldType.vcFFTGraphics;
boxFormatField.GraphicsHeight = 150;
```

Index

Read Only Property of VcBoxFormatField

This property lets you retrieve the index of the box format field in the associated box format.

	Data Type	Explanation
Property value	System.Int16	Index of the box format field

Example Code VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
MsgBox(boxFormatField.Index)
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
MessageBox.Show(boxFormatField.Index.ToString());
```


MaximumTextLineCount

Property of VcBoxFormatField

This property lets you set or retrieve the maximum number of lines in the box format field, if the box format field is of the type **vcFFTText**. Also see the property **MinimumTextLineCount**.

	Data Type	Explanation
Property value	System.Int16	Maximum number of lines

Example Code VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.Type = VcFormatFieldType.vcFFTText
boxFormatField.MaximumTextLineCount = 5
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.Type = VcFormatFieldType.vcFFTText;
boxFormatField.MaximumTextLineCount = 5;
```

MinimumTextLineCount

Property of VcBoxFormatField

This property lets you set or retrieve the minimum number of lines in the box format field, if it is of the type **vcFFTText**. If there is more text than can be taken by the lines, the format field will be enlarged dynamically up to the maximum number of lines. When assigning a value by this property, please also remember to set the **MaximumTextLineCount** value anew, since otherwise the minimum value might overwrite the maximum value.

	Data Type	Explanation
Property value	System.Int16	Minimum number of lines 0...20

Example Code VB.NET

```

Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.Type = VcFormatFieldType.vcFFTTText
boxFormatField.MinimumTextLineCount = 3

```

Example Code C#

```

VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.Type = VcFormatFieldType.vcFFTTText;
boxFormatField.MinimumTextLineCount = 3;

```

MinimumWidth

Property of VcBoxFormatField

This property lets you set or retrieve the minimum width of the box field in mm. The field width may be enlarged, if above or below the field fields exist that have greater minimum widths.

	Data Type	Explanation
Property value	System.Int16	Minimum width of the box format field 0...200

Example Code VB.NET

```

Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.MinimumWidth = 100

```

Example Code C#

```

VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.MinimumWidth = 100;

```

PatternBackgroundColor

Property of VcBoxFormatField

This property lets you set or retrieve the background color of the box format field. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between

0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color.

If the box format field shall have the background color of the box format, select the value **-1**.

	Data Type	Explanation

Example Code VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.BackgroundColor = Color.Red
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.BackgroundColor = Color.Red;
```

PatternColorAsARGB

Read Only Property of VcBoxFormatField

This property lets you set or retrieve the pattern color of the box format field. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color. When casting an RGB value on an ARGB value, an alpha value of 255 has to be added.

If the box format field shall have the background color of the box format, select the value **-1**.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255}) Default value: -1

Example Code VB.NET

```
boxFormatField.PatternColor = RGB(0, 255, 0)
```

TextFont

Property of VcBoxFormatField

This property lets you set or retrieve the font of the box format field, if it is of the type **vcFFTText**.

	Data Type	Explanation
Property value	System.Drawing.Font	Font type of the box format

Example Code VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
MsgBox(boxFormatField.TextFont.FontFamily.ToString())
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
MessageBox.Show(boxFormatField.TextFont.Name.ToString());
```

TextFontColor

Property of VcBoxFormatField

This property lets you set or retrieve the font color of the box format field, if it is of the type **vcFFTText**.

	Data Type	Explanation
Property value	System.Drawing.Color	Font color of the box format Default value: Color.Black

Example Code VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.TextFontColor = Color.Red
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.TextFontColor = Color.Red;
```

Type

Property of VcBoxFormatField

This property lets you enquire the type of the box format field.

	Data Type	Explanation
Property value	VcFormatFieldType	Type of the box format field
	Possible Values: .vcFFTGraphics 64 .vcFFTText 36	Graphics Text

Example Code VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormatField As VcBoxFormatField

boxFormatCltn = VcGantt1.BoxFormatCollection
boxFormatField = boxFormatCltn.FirstFormat.FormatField(0)
boxFormatField.Type = VcFormatFieldType.vcFFTGraphics
boxFormatField.GraphicsHeight = 150
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
VcBoxFormatField boxFormatField =
boxFormatCltn.FirstFormat().get_FormatField(0);
boxFormatField.Type = VcFormatFieldType.vcFFTGraphics;
boxFormatField.GraphicsHeight = 150;
```

7.9 VcCalendar



A calendar serves to define work and non work periods. It is composed of a continuous sequence of work and nonwork periods, that commonly are made of Workday and Workweek objects, but may also consist of intervals. A calendar just created by default contains an interval that covers the whole project. Bars and layers adapt to the time pattern provided by the calendar.

A calendar also is useful for scheduling, e.g. to count the work days between two set dates.

You also can use a calendar to interrupt nodes by workfree intervals.

Furthermore, calendars specify calendar grids.

Properties

- CalendarProfileCollection
- IntervalCollection
- Name
- SecondsPerWorkday
- Specification
- Type

Methods

- AddDuration
- CalcDuration
- Clear
- GetEndOfPreviousWorktime
- GetNextIntervalBorder
- GetPreviousIntervalBorder
- GetStartOfInterval
- GetStartOfNextWorktime
- IsWorktime
- Update

Properties

CalendarProfileCollection

Read Only Property of VcCalendar

This property gives access to the CalendarProfileCollection object that contains all calendar profiles available in this VcCalendar object.

	Data Type	Explanation
Property value	VcCalendarProfileCollection	CalendarProfileCollection object

IntervalCollection

Read Only Property of VcCalendar

This property gives access to the IntervalCollection object that contains all intervals available.

	Data Type	Explanation
Property value	VcIntervalCollection	IntervalCollection object

Name

Read Only Property of VcCalendar

This property lets you retrieve the name of a calendar.

	Data Type	Explanation
Property value	System.String	Name of the calendar

Example Code VB.NET

```
Dim calendar As VcCalendar
Dim calendarName As String

calendar = VcGantt1.CalendarCollection.FirstCalendar
calendarName = calendar.Name
```

Example Code C#

```
VcCalendar calendar = vcGantt1.CalendarCollection.FirstCalendar();
string calendarName = calendar.Name;
```

SecondsPerWorkday

Read Only Property of VcCalendar

This property lets you set/retrieve the number of seconds of a workday. This feature can be also set in the **Specify Calendars** dialog.

	Data Type	Explanation
Property value	System.Int32	Seconds of a workday

Specification

Read Only Property of VcCalendar

This property lets you retrieve the specification of a calendar. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored smoothly to text files or data bases. This allows for persistency. A specification can be used to create a calendar by the method **VcCalendarCollection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the calendar

Example Code VB.NET

```
Dim calendarCltn As VcCalendarCollection
Dim calendar As VcCalendar

calendarCltn = VcGantt1.CalendarCollection
calendar = calendarCltn.FirstCalendar
MsgBox(calendar.Specification)
```

Example Code C#

```
VcCalendarCollection calendarCltn = vcGantt1.CalendarCollection;
VcCalendar calendar = calendarCltn.FirstCalendar();
MessageBox.Show(calendar.Specification);
```

Type

Property of VcCalendar

This property lets you set or retrieve the calendar type. If you change the type, all properties of this calendar will be deleted.

	Data Type	Explanation
Property value	VcCalendarType	Calendar type
	Possible Values:	

.vcNormalCalendar	139
.vcShiftCalendar	12

Example Code VB.NET

```
Dim calendarCltn As VcCalendarCollection
Dim calendar As VcCalendar

calendarCltn = VcGantt1.CalendarCollection
calendar = calendarCltn.CalendarByIndex(0)
calendar.Type = VcCalendarType.vcNormalCalendar
```

Example Code C#

```
VcCalendarCollection calendarCltn = vcGantt1.CalendarCollection;
VcCalendar calendar = calendarCltn.CalendarByIndex(0);
calendar.Type = VcCalendarType.vcNormalCalendar;
```

Methods

AddDuration

Method of VcCalendar

This method lets you assign a duration (work time) to a date of the calendar, considering the settings of the calendar. If e.g. you have defined workfree weekends to your calendar, a duration of three days added to a Friday will result in the Wednesday following.

	Data Type	Explanation
Parameter:		
⇒ date	System.DateTime	Date the duration is to be inserted at
⇒ duration	System.Int32	Number of time units (e.g.days)
Return value	System.DateTime	Date the duration was inserted at

Example Code VB.NET

```
Dim calendar As VcCalendar
Dim newDate As Date

calendar = VcGantt1.CalendarCollection.CalendarByName("WeekCalendar")
newDate = calendar.AddDuration("16.06.2017", 3)
```

Example Code C#

```
VcCalendar calendar =
vcGantt1.CalendarCollection.CalendarByName("WeekCalendar");
DateTime newDate = calendar.AddDuration(Convert.ToDateTime("16.06.2017"), 3);
```

CalcDuration

Method of VcCalendar

This method lets you retrieve the number of work time elements (e.g. work days) available between two defined dates. The unit (e.g. days) of the value returned is the one defined in the **Time Unit** field on the **General** property page.

	Data Type	Explanation
Parameter:		
⇒ fromDate	System.DateTime	Start date of the duration that the number of work time elements is to be retrieved of
⇒ toDate	System.DateTime	End date of the duration that the number of work time elements is to be retrieved of
Return value	System.Int32	Number of time units (e.g. days) of the duration

Example Code VB.NET

```
Dim calendar As VcCalendar
Dim duration As Integer

calendar = VcGantt1.CalendarCollection.CalendarByName("WeekCalendar")
duration = calendar.CalcDuration("01.01.2014", "31.12.2014")
```

Example Code C#

```
VcCalendar calendar =
vcGantt1.CalendarCollection.CalendarByName("WeekCalendar");
int duration = calendar.CalcDuration(Convert.ToDateTime("01.01.2014"),
Convert.ToDateTime("31.12.2014"));
```

Clear

Method of VcCalendar

Removes the profiles and intervals formerly defined in this VcCalendar object, thus completely clearing it (=> 100% working time). The changes will only be displayed after an update.

	Data Type	Explanation

GetEndOfPreviousWorktime

Method of VcCalendar

This method lets you retrieve the end of the work time that precedes the reference date. The reference date has to belong to a non-working period.

	Data Type	Explanation
Parameter:		
⇒ date	System.DateTime	Date that the previous work time refers to
Return value	System.DateTime	Final date of the previous work time

Example Code VB.NET

```
Dim calendar As VcCalendar
Dim endOfWork As Date

calendar = VcGantt1.CalendarCollection.CalendarByName("WeekCalendar")
endOfWork = calendar.GetEndOfPreviousWorktime("18.06.2014")
```

Example Code C#

```
VcCalendar calendar =
VcGantt1.CalendarCollection.CalendarByName("WeekCalendar");
DateTime endOfWork =
calendar.GetEndOfPreviousWorktime(Convert.ToDateTime("18.06.2014"));
```

GetNextIntervalBorder

Method of VcCalendar

This method lets you retrieve the beginning of the interval succeeding. If the reference date is in a non work time, the date returned will be the beginning of the succeeding work time, and vice versa.

	Data Type	Explanation
Parameter:		
⇒ date	System.DateTime	Date that the subsequent interval border refers to
Return value	System.DateTime	Start date of the subsequent interval border

Example Code VB.NET

```
Dim calendar As VcCalendar
Dim nextIntervalBorder As Date

calendar = VcGantt1.CalendarCollection.CalendarByName("WeekCalendar")
nextIntervalBorder = calendar.GetNextIntervalBorder("18.06.2014")
```

Example Code C#

```
VcCalendar calendar =
vcGantt1.CalendarCollection.CalendarByName("WeekCalendar");
DateTime nextIntervalBorder =
calendar.GetNextIntervalBorder(Convert.ToDateTime("18.06.2014"));
```

GetPreviousIntervalBorder**Method of VcCalendar**

This method lets you retrieve the end of the preceding interval. If the reference date is in a non work time, the date returned will be the end of the preceding work time, and vice versa.

	Data Type	Explanation
Parameter: ⇒ date	System.DateTime	Date that of the preceding interval border refers to
Return value	System.DateTime	End date of the interval border preceding

Example Code VB.NET

```
Dim calendar As VcCalendar
Dim previousIntervalBorder As Date

calendar = vcGantt1.CalendarCollection.CalendarByName("WeekCalendar")
previousIntervalBorder = calendar.GetPreviousIntervalBorder("18.06.2014")
```

Example Code C#

```
VcCalendar calendar =
vcGantt1.CalendarCollection.CalendarByName("WeekCalendar");
DateTime previousIntervalBorder =
calendar.GetPreviousIntervalBorder(Convert.ToDateTime("18.06.2014"));
```

GetStartOfInterval**Method of VcCalendar**

This method lets you retrieve the beginning of the interval that the reference date is located in.

	Data Type	Explanation
Parameter: ⇒ date	System.DateTime	Reference date of the interval, that the start date is to be retrieved of
Return value	System.DateTime	Start date of the interval

Example Code VB.NET

```
Dim calendar As VcCalendar
Dim startOfInterval As Date

calendar = VcGantt1.CalendarCollection.CalendarByName("WeekCalendar")
startOfInterval = calendar.GetStartOfInterval("18.06.2014")
```

Example Code C#

```
VcCalendar calendar =
vcGantt1.CalendarCollection.CalendarByName("WeekCalendar");
DateTime startOfInterval =
calendar.GetStartOfInterval(Convert.ToDateTime("18.06.2014"));
```

GetStartOfNextWorktime

Method of VcCalendar

This method lets you retrieve the beginning of the work time that succeeds the reference date.

	Data Type	Explanation
Parameter: ⇒ date	System.DateTime	Reference date, of which the start date of the subsequent work time is to be retrieved
Return value	System.DateTime	Start date of the subsequent work time

Example Code VB.NET

```
Dim calendar As VcCalendar
Dim startOfNextWorktime As Date

calendar = VcGantt1.CalendarCollection.CalendarByName("WeekCalendar")
startOfNextWorktime = calendar.GetStartOfNextWorktime("18.06.2017")
```

Example Code C#

```
VcCalendar calendar =
vcGantt1.CalendarCollection.CalendarByName("WeekCalendar");
DateTime startOfNextWorktime =
calendar.GetStartOfNextWorktime(Convert.ToDateTime("18.06.2017"));
```

IsWorktime

Method of VcCalendar

This method lets you retrieve whether or not the date passed is in a work time.

	Data Type	Explanation
Parameter: ⇒ date	System.DateTime	Date to be checked for being a work time

Return value	System.Boolean	Date passed does /does not belong to a work time
---------------------	----------------	--

Example Code VB.NET

```
Dim calendar As VcCalendar
Dim isWorktime As Boolean

calendar = VcGantt1.CalendarCollection.CalendarByName("WeekCalendar")
isWorktime = calendar.IsWorktime ("18.06.2014")
```

Example Code C#

```
VcCalendar calendar =
vcGantt1.CalendarCollection.CalendarByName("WeekCalendar");
bool isWorktime = calendar.IsWorktime(Convert.ToDateTime("18.06.2014"));
```

Update

Method of VcCalendar

This method lets you update a calendar after having modified it. It ensures other objects that use calendar (e.g. a calendarGrid) to be updated as well.

	Data Type	Explanation
Return value	Void	

Example Code VB.NET

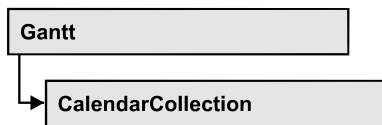
```
Dim calendar As VcCalendar

calendar = VcGantt1.CalendarCollection.CalendarByName("WeekCalendar")
calendar.Update()
```

Example Code C#

```
VcCalendar calendar =
vcGantt1.CalendarCollection.CalendarByName("WeekCalendar");
calendar.Update();
```

7.10 VcCalendarCollection



An object of the type `VcCalendarCollection` automatically contains all available calendars. You can access all objects in an iterative loop by **For Each calendar In CalendarCollection** or by the methods **First...** and **Next...**. You can access a single calendar by the method **CalendarByName**. The number of calendars in the collection object can be retrieved by the property **Count**. By the property **Active** you can set or retrieve the calendar which controls the calendar grid.

Properties

- Active
- Count

Methods

- Add
- AddBySpecification
- CalendarByIndex
- CalendarByName
- Copy
- FirstCalendar
- GetEnumerator
- NextCalendar
- Remove
- Update

Properties

Active

Property of `VcCalendarCollection`

This property lets you retrieve or set the default calendar for nodes, if no other calendar was assigned.

	Data Type	Explanation
Property value	VcCalendar	Currently used calendar

Example Code VB.NET

```

Dim workday As VcWorkday
Dim freeday As VcWorkday
Dim workweek As VcWorkweek
Dim calendarCltn As VcCalendarCollection
Dim calendar As VcCalendar

workday = VcGantt1.WorkdayCollection.CreateWorkday("Work day")
workday.AddNonWorkInterval("00:00:00", "00:00:00")
workday.AddWorkInterval("08:00:00", "16:30:00")
freeday = VcGantt1.WorkdayCollection.CreateWorkday("Workfree day")
freeday.AddNonWorkInterval("00:00:00", "00:00:00")
calendarCltn = VcGantt1.CalendarCollection
calendar = calendarCltn.CreateCalendar("New calendar")
workweek = VcGantt1.WorkweekCollection.CreateWorkweek("Work week")
workweek.AddWorkday(workday, VcWeekday.vcMonday, VcWeekday.vcFriday)
workweek.AddWorkday(freeday, VcWeekday.vcSaturday, VcWeekday.vcSunday)
calendar.AddWorkweek(workweek, "01.01.13", "31.12.14")
calendar.Update()
calendarCltn.Active = calendar

```

Example Code C#

```

VcWorkday workday = VcGantt1.WorkdayCollection.CreateWorkday("Work day");
workday.AddNonWorkInterval(Convert.ToDateTime("00:00:00"),
Convert.ToDateTime("00:00:00"));
workday.AddWorkInterval(Convert.ToDateTime("08:00:00"),
Convert.ToDateTime("16:30:00"));
VcWorkday freeday = VcGantt1.WorkdayCollection.CreateWorkday("Workfree day");
freeday.AddNonWorkInterval(Convert.ToDateTime("00:00:00"),
Convert.ToDateTime("00:00:00"));
VcCalendarCollection calendarCltn = VcGantt1.CalendarCollection;
VcCalendar calendar = calendarCltn.CreateCalendar("New calendar");
VcWorkweek workweek = VcGantt1.WorkweekCollection.CreateWorkweek("Work week");
workweek.AddWorkday(workday, VcWeekday.vcMonday, VcWeekday.vcFriday);
workweek.AddWorkday(freeday, VcWeekday.vcSaturday, VcWeekday.vcSunday);
calendar.AddWorkweek(workweek, Convert.ToDateTime("01.01.13"),
Convert.ToDateTime("31.12.14"));
calendar.Update();
calendarCltn.Active = calendar;

```

Count**Read Only Property of VcCalendarCollection**

This property lets you retrieve the number of calendars in the CalendarCollection object.

	Data Type	Explanation
Property value	System.Int32	Number of calendars

Example Code VB.NET

```
Dim calendarCltn As VcCalendarCollection
Dim numberOfCalendar As Integer

calendarCltn = VcGantt1.CalendarCollection
numberOfCalendar = calendarCltn.Count
```

Example Code C#

```
VcCalendarCollection calendarCltn = vcGantt1.CalendarCollection;
int numberOfCalendar = calendarCltn.Count;
```

Methods

Add

Method of VcCalendarCollection

By this method you can create a calendar as a member of the CalendarCollection. If the name has not been used before, the new calendar object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ calendarName	System.String	Calendar name
Return value	VcCalendar	New calendar object

AddBySpecification

Method of VcCalendarCollection

This method lets you create a calendar by using a calendar specification. This way of creating allows calendar objects to become persistent. The specification of a calendar can be saved and re-loaded (see VcCalendar property **Specification**). In a subsequent the calendar can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter: ⇒ Specification	System.String	Calendar specification
Return value	VcCalendar	New calendar object

CalendarByIndex

Method of VcCalendarCollection

This method lets you access a calendar by its index. If a calendar does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the calendar
Return value	VcCalendar	Calendar object returned

CalendarByName

Method of VcCalendarCollection

By this method you can retrieve a calendar by its name. If a calendar of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ calendarName	System.String	Name of the calendar
Return value	VcCalendar	Calendar

Example Code VB.NET

```
Dim calendarCltn As VcCalendarCollection

calendarCltn = VcGantt1.CalendarCollection
calendarCltn.Active = calendarCltn.CalendarByName("Calendar_1")
```

Example Code C#

```
VcCalendarCollection calendarCltn = vcGantt1.CalendarCollection;
calendarCltn.Active = calendarCltn.CalendarByName("Calendar_1");
```

Copy

Method of VcCalendarCollection

By this method you can copy a calendar. If the calendar that is to be copied exists, and if the name for the new calendar does not yet exist, the new calendar object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter:		
⇒ calendarName	System.String	Name of the calendar to be copied
⇒ newCalendarName	System.String	Name of the calendar
Return value	VcCalendar	Calendar object

FirstCalendar

Method of VcCalendarCollection

This method can be used to access the initial value, i.e. the first calendar of a calendar collection, to continue in a forward iteration loop by the method **NextCalendar** for the calendars following. If there is no calendar in the calendar collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcCalendar	First calendar

Example Code VB.NET

```
Dim calendarCltn As VcCalendarCollection
Dim calendar As VcCalendar

calendarCltn = VcGantt1.CalendarCollection
calendar = calendarCltn.FirstCalendar
```

Example Code C#

```
VcCalendarCollection calendarCltn = vcGantt1.CalendarCollection;
VcCalendar calendar = calendarCltn.FirstCalendar();
```

GetEnumerator

Method of VcCalendarCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the calendar objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

Example Code VB.NET

```
Dim calendar As VcCalendar

For Each calendar In VcGantt1.CalendarCollection
    MsgBox(calendar.Name)
Next
```

Example Code C#

```
foreach (VcCalendar calendar in vcGantt1.CalendarCollection)
    MessageBox.Show(calendar.Name);
```

NextCalendar

Method of VcCalendarCollection

This method can be used in a forward iteration loop to retrieve subsequent calendars from a calendar collection after initializing the loop by the method **FirstCalendar**. If there is no calendar left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcCalendar	Succeeding calendar

Example Code VB.NET

```
Dim calendarCltn As VcCalendarCollection
Dim calendar As VcCalendar
calendarCltn = VcGantt1.CalendarCollection
calendar = calendarCltn.FirstCalendar

While Not calendar Is Nothing
    ListBox1.Items.Add(calendar.Name)
    calendar = calendarCltn.NextCalendar
End While
```

Example Code C#

```
VcCalendarCollection calendarCltn = vcGantt1.CalendarCollection;
VcCalendar calendar = calendarCltn.FirstCalendar();

while (calendar != null)
{
    ListBox.Items.Add(calendar.Name);
    calendar = calendarCltn.NextCalendar();
}
```

Remove

Method of VcCalendarCollection

This method lets you delete a calendar. If the calendar is used in another object, it cannot be deleted. Then False will be returned, otherwise True.

	Data Type	Explanation
Return value	System.Boolean	Calendar deleted (True)/not deleted (False)

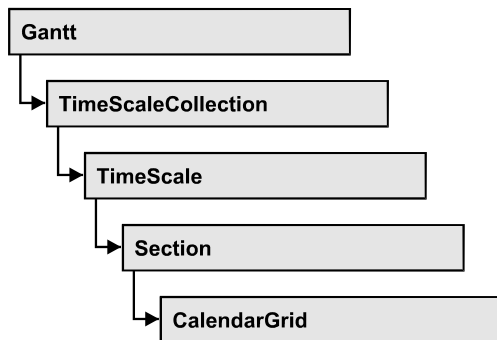
Update

Method of VcCalendarCollection

This method lets you update a calendar collection after having modified it.

	Data Type	Explanation
Return value	System.Boolean	update successful (True)/ not successful (False)

7.11 VcCalendarGrid



An object of the type **VcCalendarGrid** is a grid of vertical lines to highlight workfree periods by colored vertical areas.

Properties

- BackgroundColor
- BackgroundColorDataFieldIndex
- BackgroundColorMapName
- CalendarName
- CalendarNameDataFieldIndex
- CalendarNameMapName
- EndSnapTarget
- Identifiable
- LineColor
- LineColorDataFieldIndex
- LineColorMapName
- LineThickness
- LineType
- Name
- Pattern
- PatternColor
- PatternColorDataFieldIndex
- PatternColorMapName
- PatternDataFieldIndex
- PatternMapName
- Priority
- SnapTarget
- Specification
- StartSnapTarget
- UseGraphicalAttributesOfIntervals

- Visible
- VisibleDataFieldIndex
- VisibleMapName

Methods

- IdentifyInterval

Properties

BackgroundColor

Property of VcCalendarGrid

This property lets you specify or retrieve the color of the vertical areas of the calendar grid. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color.

Also see **set/getPatternColor** and **set/getPattern**.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values Default value: 14 211 288. Visual Basic: RGB (216, 216, 216)

Example Code VB.NET

```
Dim section As VcSection
Dim calendarGrid As VcCalendarGrid

section = VcGantt1.TimeScaleCollection.Active.Section(0)
calendarGrid = section.CalendarGrid(0)
calendarGrid.Color = Color.Blue
```

Example Code C#

```
VcSection section = vcGantt1.TimeScaleCollection.Active.get_Section(0);
VcCalendarGrid calendarGrid = section.get_CalendarGrid(0);
calendarGrid.Color = Color.LightSteelBlue;
```

BackgroundColorDataFieldIndex

Property of VcCalendarGrid

This property lets you set or retrieve the data field index to be used with a color map specified by the property **BackColorMapName**. If you set this property to -1, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

BackColorMapName

Property of VcCalendarGrid

This property lets you set or retrieve the name of a color map (type `vcColorMap`). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **BackColorDataFieldIndex**, then the background color is controlled by the map. If no data field entry applies, the background color that is specified in the property **BackColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

CalendarName

Property of VcCalendarGrid

This property lets you assign a calendar to the calendar grid to highlight the calendar's workfree periods.

	Data Type	Explanation
Property value	System.String	Character string that passes the calendar name

CalendarNameDataFieldIndex

Property of VcCalendarGrid

This property lets you set or retrieve the index of the data field that holds the name of the calendar for the calendar grid of the grouping level. This property also can be set on the **Calendar Grid** property page.

	Data Type	Explanation
Property value	System.Int16	Index of the data field which contains the name of the calendar

CalendarNameMapName

Property of VcCalendarGrid

This property lets you set or retrieve the name of a calendar map (type vcTextMap). If set to "", no map will be used. If a map name and additionally a data field index is specified by the property **CalendarNameDataFieldIndex**, the calendar is selected by the map. If no data field entry applies, the calendar that was assigned to the calendar grid of the grouping level will be used.

	Data Type	Explanation
Property value	System.String	Name of the calendar map

EndSnapTarget

Read Only Property of VcCalendarGrid

This property lets you set or retrieve whether the end date of this calendar is to define as snap target.

	Data Type	Explanation
Property value	System.Boolean	End date of this calendar grid is/is not defined as snap target

Identifiable

Property of VcCalendarGrid

This property lets you set or retrieve whether or not a calendar grid can be identified. If this property was set to **True**, the calendar grid can be identified by the VcGantt method **IdentifyObjectAt**. Also, a tooltip text retrieved by **OnTooltipText** will only appear if this property was set to **True**. In the same way, the **VcCalendarGridRightClicking** event will only be triggered if the calendar grid is identifiable.

To produce specific tooltip texts, in addition the corresponding intervals of a calendar need to be identified: see VcGantt method **IdentifyInterval**.

This property can also be set in the **calendar grid** section of the **Edit time scale section** dialog.

	Data Type	Explanation
Property value	System.Boolean	Calendar grid can/cannot be identified Default value: False

Example Code VB.NET

```
Dim section As VcSection
Dim calendarGrid As VcCalendarGrid

section = VcGantt1.TimeScaleCollection.Active.Section(0)
calendarGrid = section.CalendarGrid(0)
calendarGrid.Identifiable = True
```

Example Code C#

```
VcSection section = vcGantt1.TimeScaleCollection.Active.get_Section(0);
VcCalendarGrid calendarGrid = section.get_CalendarGrid(0);
calendarGrid.Identifiable = true;
```

LineColor

Read Only Property of VcCalendarGrid

This property lets you specify/enquire the line color of a calendar grid and can also be set in the **Line attributes of calendar grid** dialog.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255} Default value: As defined in the dialog

LineColorDataFieldIndex

Read Only Property of VcCalendarGrid

This property lets you set or retrieve the data field index to be used with a map specified by the property **LineColorMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int32	Data field index

LineColorMapName

Read Only Property of VcCalendarGrid

This property lets you set or retrieve the name of a map for the line color. If set to "" or if the property **LineColorDataFieldIndex** is set to **-1**, then no map will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

LineThickness

Read Only Property of VcCalendarGrid

This property lets you set or retrieve the line thickness of the calendar grid lines.

If you set this property to values between 1 and 4, an absolute line thickness is defined in pixels. Irrespective of the zoom factor a line will always show the same line thickness in pixels. When printing though, the line thickness is adapted for the sake of legibility and becomes dependent of the zoom factor:

Value	Points	mm
1	1/2 point	0.09 mm
2	1 point	0.18 mm
3	3/2 points	0.26 mm
4	2 points	0.35 mm

A point equals 1/72 inch and represents the unit of the font size.

If you set this property to values between 5 and 1,000, the line thickness is defined in 1/100 mm, so the lines will be displayed in a true thickness in pixels that depends on the zoom factor.

This property also can be set in the **Attributes of calendar grid** dialog.

	Data Type	Explanation
Property value	System.Int16	Line thickness LineType {1...4}: line thickness in pixels LineType {5...1000}: line thickness in 1/100 mm Default value: As defined in the dialog

LineType

Read Only Property of VcCalendarGrid

This property lets you specify/enquire the line type of a calendar grid.

This property also can be set in the **Attributes of calendar grid** dialog.

	Data Type	Explanation
Property value	VcLineType Possible Values: .vcDashed 4 .vcDashed 4 .vcDashedDotted 5 .vcDashedDotted 5 .vcDotted 3 .vcDotted 3 .vcLineType0 100 .vcLineType1 101 .vcLineType10 110 .vcLineType11 111 .vcLineType12 112 .vcLineType13 113 .vcLineType14 114 .vcLineType15 115 .vcLineType16 116	Line type Line dashed Line dashed Line dashed-dotted Line dashed-dotted Line dotted Line dotted Line Type 0 <hr/> Line Type 1 ----- Line Type 10 ----- Line Type 11 ----- Line Type 12 ----- Line Type 13 ----- Line Type 14 ----- Line Type 15 ----- Line Type 16 -----

.vcLineType17 117	Line Type 17
.vcLineType18 118	Line Type 18
.vcLineType2 102	Line Type 2
.vcLineType3 103	Line Type 3
.vcLineType4 104	Line Type 4
.vcLineType5 105	Line Type 5
.vcLineType6 106	Line Type 6
.vcLineType7 107	Line Type 7
.vcLineType8 108	Line Type 8
.vcLineType9 109	Line Type 9
.vcNone 1	No line type assigned
.vcNone 1	No line type
.vcSolid 2	Line solid
.vcSolid 2	Line solid

Name

Read Only Property of VcCalendarGrid


This property lets you specify/enquire the name of a calendar grid.








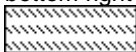
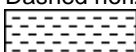



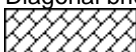
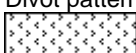
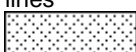
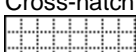
	Data Type	Explanation
Property value	System.String	Name of the calendar grid




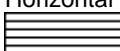
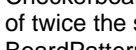


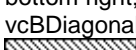
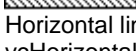
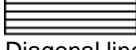
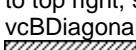
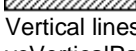


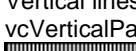
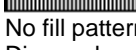


Pattern

Property of VcCalendarGrid

This property lets you set or retrieve the pattern of the calendar grid.

	Data Type	Explanation
Property value	VcFillPattern Possible Values: .vc05PercentPattern... vc90PercentPattern 01 - 11	Pattern type Dots in foreground color on background color, the density of the foreground color increasing with the percentage 

.vcAeroGlassPattern 44	Vertical color gradient in the color of the fill pattern 
.vcBDiagonalPattern 5	Diagonal lines slanting from bottom left to top right 
.vcCrossPattern 6	Cross-hatch pattern 
.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width 
.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 
.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width 
.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 
.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
.vcDashedHorizontalPattern 2026	Dashed horizontal lines 
.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 
.vcDashedVerticalPattern 2027	Dashed vertical lines 
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small 
.vcDiagonalBrickPattern 2032	Diagonal brick pattern 
.vcDivotPattern 2036	Divot pattern 
.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines 
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines 

.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right 
.vcHorizontalBrickPattern 2033	Horizontal brick pattern 
.vcHorizontalGradientPattern 52	Horizontal color gradient 
.vcHorizontalPattern 3	Horizontal lines 
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern 
.vcLargeConfettiPattern 2029	Confetti pattern, large 
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern 
.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern 
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern 
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern 
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern 
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern 
.vcNoPattern 1276	No fill pattern
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large 
.vcPlaidPattern 2035	Plaid pattern 
.vcShinglePattern 2039	Diagonal shingle pattern 
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern 
.vcSmallConfettiPattern 2028	Confetti pattern 
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern 

.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares
.vcSpherePattern 2041	Checkerboard of spheres
.vcTrellisPattern 2040	Trellis pattern
.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright
.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright
.vcVerticalGradientPattern 62	Vertical color gradient
.vcVerticalPattern 2	Vertical lines
.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark
.vcWavePattern 2031	Horizontal waves pattern
.vcWeavePattern 2034	Interwoven stripes pattern
.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern
.vcWideUpwardDiagonalPattern 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of vcBDiagonalPattern
.vcZigZagPattern 2030	Horizontal zig-zag lines

PatternColor

Property of VcCalendarGrid

This property lets you set or retrieve the pattern color of the calendar grid. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255.

An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color.

Also see **set/getBackgroundColor** and **set/getPattern**.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255})

PatternColorDataFieldIndex

Property of VcCalendarGrid

This property lets you set or retrieve the data field index that has to be specified if the property **PatternColorMapName** is used. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Parameter: ⇒ Rückgabewert	System.Int16	Data field index
Property value	System.Int16	Data field index

PatternColorMapName

Property of VcCalendarGrid

This property lets you set or retrieve the name of a color map (type `vcColorMap`). If set to "", no map will be used. Only if a map name and a data field index are specified in the property **PatternColorDataFieldIndex**, the pattern color is controlled by the map. If no data field entry applies, the pattern color of the calendar grid that is specified in the property **PatternColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

PatternDataFieldIndex

Property of VcCalendarGrid

This property lets you set or retrieve the data field index to be used with the property **PatternMapName**. If you set this property to -1, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

PatternMapName

Property of VcCalendarGrid

This property lets you set or retrieve the name of a pattern map (type vcPatternMap). If set to "", no map will be used. Only if a map name and in addition a data field index are specified in the property **PatternDataFieldIndex**, the pattern will be controlled by the map. If none of the data field entry applies, the pattern of the layer that is specified in the property **Pattern** will be used.

	Data Type	Explanation
Property value	System.String	Name of the pattern map

Priority

Property of VcCalendarGrid

This property lets you set or retrieve the priority of the calendar grid. If two objects are located in the same position of the diagram, the object of higher priority is displayed in front of the objects of lower priority. By default, calendar grid lines are of lowest priority. Nodes are assigned the value 0 and thus have the highest priority of all objects. If you want a calendar grid to be displayed in front of the nodes, its priority needs to be set to a positive value.

	Data Type	Explanation
Property value	System.Int16	Rank of Priority {-100 ... 100} Default value: -20

Example Code VB.NET

```
Dim section As VcSection
Dim calendarGrid As VcCalendarGrid

section = VcGantt1.TimeScaleCollection.Active.Section(0)
calendarGrid = section.CalendarGrid(0)
calendarGrid.Priority = 3
```

Example Code C#

```
VcSection section = vcGantt1.TimeScaleCollection.Active.get_Section(0);
VcCalendarGrid calendarGrid = section.get_CalendarGrid(0);
calendarGrid.Priority = 3;
```

SnapTarget

Read Only Property of VcCalendarGrid

This property lets you set or retrieve whether this calendar grid has a snap target at the date.

	Data Type	Explanation

Specification

Read Only Property of VcCalendarGrid

This property lets you retrieve the specification of a calendar grid. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored smoothly to text files or data bases. This allows for persistency. A specification can be used to create a calendar grid by the method **VcCalendarGridCollection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the calendar grid

Example Code VB.NET

```
Dim calendarGridCltn As VcCalendarGridCollection
Dim calendarGrid As VcCalendarGrid

calendarGridCltn = VcGantt1.CalendarGridCollection
calendarGrid = calendarGridCltn.FirstCalendarGrid
MsgBox(calendarGrid.Specification)
```

Example Code C#

```
VcCalendarGridCollection calendarGridCltn = vcGantt1.CalendarGridCollection;
VcCalendarGrid calendarGrid = calendarGridCltn.FirstCalendarGrid();
MessageBox.Show(calendarGrid.Specification);
```

StartSnapTarget


Read Only Property of VcCalendarGrid

This property lets you set or retrieve whether the start date of this calendar is to define as snap target.

	Data Type	Explanation
Property value	System.Boolean	Start date of this calendar grid is/is not defined as snap target

UseGraphicalAttributesOfIntervals

Read Only Property of VcCalendarGrid

This property lets you set or retrieve whether the graphical attributes that were set to intervals are to be displayed. This feature can be also set in the dialog **Administrative Intervals** (which you reach by clicking  in the **Specify Calendar** dialog). If this property is set to **False**, the settings of the property **VcInterval.UseGraphicalAttributes** will have no effect.

	Data Type	Explanation
Property value	System.Boolean	Graphical attributes of the intervals are displayed (True) / are not displayed (False)

Visible

Property of VcCalendarGrid

This property lets you set or retrieve whether a calendar grid is visible.

	Data Type	Explanation
Property value	System.Boolean	Calendar grid visible / invisible Default value: True

Example Code VB.NET

```
Dim section As VcSection
Dim calendarGrid As VcCalendarGrid

section = VcGantt1.TimeScaleCollection.Active.Section(0)
calendarGrid = section.CalendarGrid(0)
calendarGrid.Visible = False
```

Example Code C#

```
VcSection section = vcGantt1.TimeScaleCollection.Active.get_Section(0);
VcCalendarGrid calendarGrid = section.get_CalendarGrid(0);
calendarGrid.Visible = true;
```

VisibleDataFieldIndex**Property of VcCalendarGrid**

This property lets you set or retrieve the index of the data field to assign a visibility mode to the calendar grid: 1 (for "visible") or 0 (for invisible). This property also can be set in the **CalendarGrid** dialog.

	Data Type	Explanation
Property value	System.Int16	Index of the data field which contains the visibility mode

VisibleMapName**Property of VcCalendarGrid**

This property lets you set or retrieve the name of a map (type vcTextMap) to set the visibility mode. If set to "", no map will be used. If a map name and additionally a data field index is specified by the property **VisibilityDataFieldIndex**, the visibility mode is selected by the map. This property also can be set in the **CalendarGrid** dialog. If no data field entry from the map applies, the visibility will adopt the value set in the dialog.

	Data Type	Explanation
Property value	System.String	Name of the map that contains the visibility mode

Methods**IdentifyInterval****Method of VcCalendarGrid**

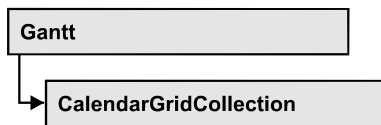
This method lets you identify an interval object of the calendar that was assigned to the calendar grid at the coordinates passed. Since usually copies of intervals exist in a calendar, intervals tend not to be unique (for instance, the same weekend interval may repeat 52 times per year). Therefore the

method also returns the start and end dates of the interval retrieved. This method is useful when being invoked within a tooltip event to return the interval at the position of the mouse cursor.

If there is an interval at the position specified, **True** will be returned, if there isn't, the method will deliver **False**.

	Data Type	Explanation
Parameter:		
⇒ x	System.Int32	X coordinate of the cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇐ identifiedIntervalParam	VcInterval	Interval found
⇐ startDateParam	System.DateTime	Start date of the interval identified
⇐ endDateParam	System.DateTime	End date of the interval identified
Return value	Boolean	An interval was found (True) / was not found (False)

7.12 VcCalendarGridCollection



An object of the type `VcCalendarGridCollection` contains all available calendar grids. You can access all objects in an iterative loop by **For Each calendarGrid In CalendarGridCollection** or by the methods **First...** and **Next...**. You can access a single calendar grid using the methods **CalendarGridByName** and **CalendarGridByIndex**. The number of calendar grids in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the calendar grids in the corresponding way.

Properties

- Count

Methods

- Add
- AddBySpecification
- CalendarGridByIndex
- CalendarGridByName
- Copy
- FirstCalendarGrid
- GetEnumerator
- NextCalendarGrid
- Remove
- Update

Properties

Count

Read Only Property of VcCalendarGridCollection

This property lets you retrieve the number of calendar grids in the `CalendarGridCollection` object.

	Data Type	Explanation
Property value	System.Int32	Number of calendar grids

Example Code VB.NET

```
Dim numberOfDateLine As Integer

numberOfDateLine = VcGantt1.DateLineCollection.Count
```

Example Code C#

```
int numberOfDateLines = vcGantt1.DateLineCollection.Count;
```

Methods

Add

Method of VcCalendarGridCollection

This method lets you create a calendar grid as a member of the CalendarGridCollection. If the name was not used before, the new calendar grid object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ calendarGridName	System.String	name of calendar grid
Return value	VcCalendarGrid	New calendar grid object

Example Code VB.NET

```
newCalendarGrid = VcGantt1.CalendarGridCollection.Add("calendarGrid1")
```

Example Code C#

```
newCalendarGrid = vcGantt1.CalendarGridCollection.Add("calendarGrid1");
```

AddBySpecification

Method of VcCalendarGridCollection

This method lets you create a calendar grid by using a calendar grid specification. This way of creating allows calendar grid objects to become persistent. The specification of a calendar grid can be saved and re-loaded (see VcCalendarGrid property **Specification**). In a subsequent session the

calendar grid can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter: ⇒ specification	System.String	calendar grid specification
Return value	VcCalendarGrid	New calendar grid object

Example Code VB.NET

```
Dim calendarGridCltn As VcCalendarGridCollection

calendarGridCltn = VcGantt1.CalendarGridCollection
calendarGridCltn.AddBySpecification(textSpecification)
calendarGridCltn.Update()
```

Example Code C#

```
VcCalendarGridCollection calendarGridCltn = vcGantt1.CalendarGridCollection;
calendarGridCltn.AddBySpecification(textSpecification);
calendarGridCltn.Update();
```

CalendarGridByIndex

Method of VcCalendarGridCollection

This method lets you access a calendar grid by its index. If a calendar grid of the specified index does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the calendar grid
Return value	VcCalendarGrid	calendar grid object returned

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection
Dim dateLine As VcDateLine

dateLineCltn = VcGantt1.DateLineCollection
dateLine = dateLineCltn.DateLineByIndex(0)
MsgBox(dateLine.Name)
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
VcDateLine dateLine = dateLineCltn.DateLineByIndex(0);
MessageBox.Show(dateLine.Name);
```

CalendarGridByName

Method of VcCalendarGridCollection

This method is used to access a calendar grid by its name. If a calendar grid of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ calendarGridName	System.String	Name of the calendar grid
Return value	VcCalendarGrid	calendar grid

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection
Dim dateLine As VcDateLine

dateLineCltn = VcGantt1.DateLineCollection
dateLine = dateLineCltn.DateLineByName("DateLineOne")
MsgBox(dateLine.Name)
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
VcDateLine dateLine = dateLineCltn.DateLineByName("DateLineOne");
MessageBox.Show(dateLine.Name);
```

Copy

Method of VcCalendarGridCollection

By this method you can copy a calendar grid. If the calendar grid that is to be copied exists, and if the name for the new calendar grid does not yet exist, the new calendar grid object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ calendarGridName	System.String	Name of the calendar grid to be copied
⇒ newCalendarGridName	System.String	Name of the new calendar grid
Return value	VcCalendarGrid	calendar grid object

Example Code VB.NET

```
Dim calendarGridCltn As VcCalendarGridCollection

calendarGridCltn = VcGantt1.CalendarGridCollection
calendarGridCltn.Copy("CalendarGridOne", "NewCalendarGrid")
calendarGridCltn.Update()
```

Example Code C#

```
VcCalendarGridCollection calendarGridCltn = vcGantt1.CalendarGridCollection;
calendarGridCltn.Copy("CalendarGridOne", "NewCalendarGrid");
calendarGridCltn.Update();
```

FirstCalendarGrid**Method of VcCalendarGridCollection**

This method can be used to access the initial value, i.e. the first calendar grid of a calendar grid collection and then to continue in a forward iteration loop by the method **NextCalendarGrid** for the calendar grids following. If there is no calendar grid in the CalendarGridCollection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcCalendarGrid	First calendar grid

Example Code VB.NET

```
Dim calendarGridCltn As VcCalendarGridCollection

calendarGridCltn = VcGantt1.CalendarGridCollection
calendarGrid = calendarGridCltn.FirstCalendarGrid
```

Example Code C#

```
VcCalendarGridCollection calendarGridCltn = vcGantt1.CalendarGridCollection;
VcCalendarGrid calendarGrid = calendarGridCltn.FirstCalendarGrid();
```

GetEnumerator**Method of VcCalendarGridCollection**

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the date line objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection
Dim dateLine As VcDateLine

dateLineCltn = VcGantt1.DateLineCollection
For Each dateLine In dateLineCltn
    ListBox1.Items.Add(dateLine.Name)
Next
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
foreach (VcDateLine dateLine in dateLineCltn)
    listBox1.Items.Add(dateLine.Name);
```

NextCalendarGrid**Method of VcCalendarGridCollection**

This method can be used in a forward iteration loop to retrieve subsequent calendar grids from a CalendarGridCollection after initializing the loop by the method **FirstCalendarGrid**. If there is no calendar grid left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcCalendarGrid	Subsequent calendar grid

Example Code VB.NET

```
Dim calendarGridCltn As VcCalendarGridCollection
Dim calendarGrid As VcCalendarGrid

calendarGridCltn = VcGantt1.CalendarGridCollection
calendarGrid = calendarGridCltn.FirstCalendarGrid

While Not calendarGrid Is Nothing
    ListBox1.Items.Add(calendarGrid.Name)
    calendarGrid = calendarGridCltn.NextCalendarGrid
End While
```

Example Code C#

```
VcCalendarGridCollection calendarGridCltn = vcGantt1.CalendarGridCollection;
VcCalendarGrid calendarGrid = calendarGridCltn.FirstCalendarGrid();

while (calendarGrid != null)
{
    ListBox.Items.Add(calendarGrid.Name);
    calendarGrid = calendarGridCltn.NextCalendarGrid();
}
```

Remove**Method of VcCalendarGridCollection**

This method lets you delete a calendar grid. If the calendar grid is used in another object, it cannot be deleted. Then False will be returned, otherwise True.

	Data Type	Explanation
Parameter: ⇒ calendarGridName	System.String	calendar grid name

Return value	System.Boolean	calendar grid deleted (True)/not deleted (False)
---------------------	----------------	--

Example Code VB.NET

```
Dim calendarGridCltn As VcCalendarGridCollection
Dim calendarGrid As VcCalendarGrid

calendarGridCltn = VcGantt1.CalendarGridCollection
calendarGrid = calendarGridCltn.CalendarGridByIndex(0)
calendarGridCltn.Remove(calendarGrid.Name)
calendarGridCltn.Update()
```

Example Code C#

```
VcCalendarGridCollection calendarGridCltn = vcGantt1.CalendarGridCollection;
VcCalendarGrid calendarGrid = calendarGridCltn.CalendarGridByIndex(0);
calendarGridCltn.Remove(calendarGrid.Name);
calendarGridCltn.Update();
```

Update

Method of VcCalendarGridCollection

This method has to be used when calendar grid modifications have been carried out. The method **Update** updates all objects that are concerned by the calendar grid you have edited. You should call this method at the end of the code that defines the calendar grids and the calendar grid collection. Otherwise the update will be processed before all calendar grid definitions are processed.

	Data Type	Explanation
Return value	System.Boolean	update successful (True)/ not successful (False)

Example Code VB.NET

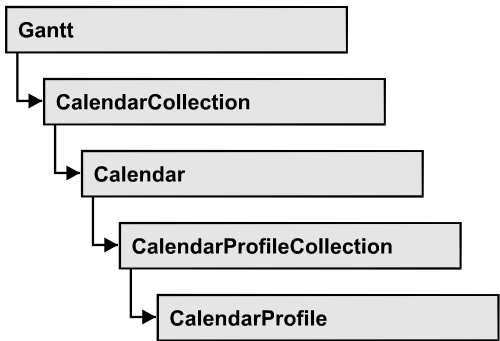
```
Dim calendarGridCltn As VcCalendarGridCollection
Dim calendarGrid As VcCalendarGrid

calendarGridCltn = VcGantt1.CalendarGridCollection
calendarGrid = calendarGridCltn.CalendarGridByIndex(0)
calendarGridCltn.Remove(calendarGrid.Name)
calendarGridCltn.Update()
```

Example Code C#

```
VcCalendarGridCollection calendarGridCltn = vcGantt1.CalendarGridCollection;
VcCalendarGrid calendarGrid = calendarGridCltn.CalendarGridByIndex(0);
calendarGridCltn.Remove(calendarGrid.Name);
calendarGridCltn.Update();
```

7.13 VcCalendarProfile



An object of the type **VcCalendarProfile** designates a calendar profile.

Properties

- IntervalCollection
- Name
- Specification
- Type

Methods

- PutInOrderAfter

Properties

IntervalCollection

Read Only Property of VcCalendarProfile

This property gives access to the IntervalCollection object that contains all intervals available.

	Data Type	Explanation
Property value	VcIntervalCollection	IntervalCollection object

Name

Read Only Property of VcCalendarProfile

This property lets you set or retrieve the name of a calendar profile

	Data Type	Explanation
Property value	System.String	Name of the calendar profile

Specification

Read Only Property of VcCalendarProfile

This property lets you retrieve the specification of a calendar profile. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored smoothly to text files or data bases. This allows for persistency. A specification can be used to create a calendar profile by the method **VcCalendarProfileCollection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the calendar profile

Example Code VB.NET

```
Dim calendarProfileCltn As VcCalendarProfileCollection
Dim calendarProfile As VcCalendarProfile

calendarProfileCltn = VcGantt1.CalendarProfileCollection
calendarProfile = calendarProfileCltn.FirstCalendarProfile
MsgBox(calendarProfile.Specification)
```

Example Code C#

```
VcCalendarProfileCollection calendarProfileCltn =
vcGantt1.CalendarProfileCollection;
VcCalendarProfile calendar = calendarProfileCltn.FirstCalendarProfile();
MessageBox.Show(calendarProfile.Specification);
```

Type

Read Only Property of VcCalendarProfile

This property lets you set or retrieve the calendar profile type. If you change the type, all properties of this calendar profile will be deleted.

	Data Type	Explanation
Property value	VcCalendarProfileType Possible Values: .vcDayProfile 4 .vcShiftProfile 5 .vcWeekProfile 3 .vcYearProfile 2	Type of the calendar profile

Methods

PutInOrderAfter

Method of VcCalendarProfile

This method lets you set the calendar profile behind the calendar profile specified by name, within the CalendarProfileCollection. If you set the name to "", the calendar profile will be put in the first position. The order of the calendar profiles within the collection determines the order by which they apply to the calendars.

	Data Type	Explanation
Parameter: refNameParam	String	Name of the calendar profile behind which the current calendar profile is to be put.

Example Code VB.NET

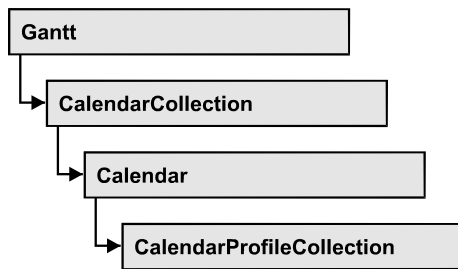
```
Dim calProfCltn As VcCalendarProfileCollection
Dim calProf1 As VcCalendarProfile
Dim calProf2 As VcCalendarProfile

calProfCltn = VcGantt1.CalendarProfileCollection()
calProf1 = calProfCltn.Add("calProf1")
calProf2 = calProfCltn.Add("calProf2")
calProf1.PutInOrderAfter("calProf2")
calProfCltn.Update()
```

Example Code C#

```
VcCalendar ProfileCollection calProfCltn = vcGantt1.Calendar ProfileCollection;
VcCalendar Profile calProf1 = calProfCltn.Add("calProf1");
VcCalendar Profile calProf2 = calProfCltn.Add("calProf2");
calProf1.PutInOrderAfter("calProf2");
calProfCltn.Update();
```


7.14 VcCalendarProfileCollection



An object of the type `VcCalendarProfileCollection` automatically contains all available calendar profiles. You can access all objects in an iterative loop by **For Each calendarProfile In CalendarProfileCollection** or by the methods **First...** and **Next...**. You can access a single calendar profile using the methods **CalendarProfileByName** and **CalendarProfileByIndex**. The number of calendar profiles in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the calendar profiles in the corresponding way.

Properties

- Count

Methods

- Add
- AddBySpecification
- CalendarProfileByIndex
- CalendarProfileByName
- Copy
- FirstCalendarProfile
- NextCalendarProfile
- Remove
- SelectCalendarProfiles
- Update
- Update

Properties

Count

Read Only Property of VcCalendarProfileCollection

This property lets you retrieve the number of calendar profiles in the calendar profile collection.

	Data Type	Explanation
Property value	System.Int32	Number of CalendarProfile objects

Methods

Add

Method of VcCalendarProfileCollection

By this method you can create a calendar profile as a member of the CalendarProfileCollection. If the name has not been used before, the new filter object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ profileName	System.String	Calendar profile name
Return value	VcCalendarProfile	New calendar profile object

AddBySpecification

Method of VcCalendarProfileCollection

This method lets you create a calendar profile by using a calendar profile specification. This way of creating allows calendar profile objects to become persistent. The specification of a calendar profile can be saved and re-loaded (see VcCalendarProfile property **Specification**). In a subsequent the calendar profile can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter: ⇒ Specification	System.String	Calendar profile specification
Return value	VcCalendarProfile	New calendarprofile object

CalendarProfileByIndex

Method of VcCalendarProfileCollection

This method lets you access a calendar profile by its index. If no calendar profile of the specified index does exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the calendar profile
Return value	VcCalendarProfile	Calendar profile object returned

CalendarProfileByName

Method of VcCalendarProfileCollection

By this method you can retrieve a calendar profile by its name. If no calendar profile of the specified name does exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ profileName	System.String	Name of the calendar profile object
Return value	VcCalendarProfile	Calendar profile object returned

Copy

Method of VcCalendarProfileCollection

By this method you can copy a calendar profile. If the calendar profile that is to be copied exists, and if the name for the new calendar profile does not yet exist, the new calendar profile object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter:		
⇒ profileName	System.String	Name of the calendar profile to be copied
⇒ newProfileName	System.String	Name of the new calendar profile
Return value	VcCalendarProfile	Calendar profile object

FirstCalendarProfile

Method of VcCalendarProfileCollection

This method can be used to access the initial value, i.e. the first calendar profile of a calendar profile collection, and then to continue in a forward iteration loop by the method **NextCalendarProfile** for the calendar profiles following. If there is no calendar profile in the FilterCollection object, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcCalendarProfile	First calendar profile object

NextCalendarProfile

Method of VcCalendarProfileCollection

This method can be used in a forward iteration loop to retrieve subsequent calendar profiles from a calendar profile collection after initializing the loop by the method **FirstCalendarProfile**. If there is no calendar profile left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcCalendarProfile	Subsequent calendar profile object

Remove

Method of VcCalendarProfileCollection

This method lets you delete a calendar profile. If the calendar profile is used in another object, it cannot be deleted. Then False will be returned, otherwise True.

	Data Type	Explanation
Parameter: ⇒ profileName	System.String	Calendar profile name
Return value	System.Boolean	Calendar profile deleted (True)/not deleted (False)

SelectCalendarProfiles

Method of VcCalendarProfileCollection

This method lets you specify the calendar profiles that the calendar profile collection is to contain.

	Data Type	Explanation
Parameter: ⇒ selectionType	CalendarProfileTypeEnum	Type of calendar profile to be selected
Return value	System.Int32	Number of calendar profiles selected

Example Code VB.NET

```
Dim calendarProfileCltn As VcCalendarProfileCollection

Set calendarProfileCltn = VcGantt1.CalendarProfileCollection
calendarProfileCltn.SelectCalendarProfile (vcSelected)
```

Update

Method of VcCalendarProfileCollection

This method lets you update a calendar profile collection after having modified it.

	Data Type	Explanation
Return value	System.Boolean	update successful (True)/ not successful (False)

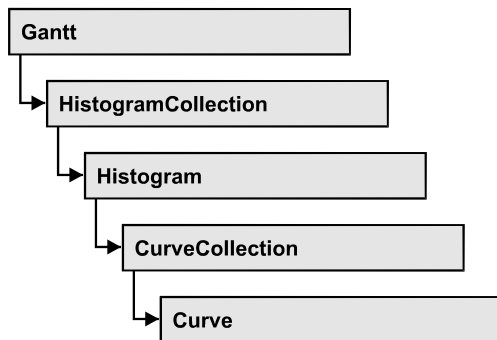
Update

Method of VcCalendarProfileCollection

This method lets you update a calendar profile collection after having modified it.

	Data Type	Explanation
Return value	System.Boolean	update successful (True)/ not successful (False)

7.15 VcCurve



A VcCurve object represents a stacked curve in the histogram which allows you, for example, to display the capacity and availability of resources. The values for the histogram curves can be entered directly or derived from layers. To enter the values directly, select the option **Data specified manually** in the **Select Curve Data Source** dialog box and generate the curve in your application using the **SetValues** method. To derive the curve from activity values, select the option **Data generated by layer** in the **Select Curve Data Source** dialog box and select a layer.

Properties

- Addend
- FillReference1BackgroundColor
- FillReference1Name
- FillReference1Pattern
- FillReference1PatternColor
- FillReference2Color
- FillReference2Name
- FillReference2Pattern
- FillReference2PatternColor
- FilterName
- Histogram
- LayerName
- LineColor
- LineThickness
- LineType
- Marked
- Name
- OverloadResultsCalendarName
- PointsEquidistant
- Source

- Specification
- StackReferenceName
- TimeUnit
- Type
- UnitsPerStep
- UpdateBehaviorName
- ValencyDataFieldIndex
- Visible

Methods

- Clear
- DeletePoint
- GetFirstOverload
- GetFirstOverloadEx
- GetNextOverload
- GetNextOverloadEx
- GetValues
- GetValuesEx
- SetValues

Properties

Addend

Property of VcCurve

This property lets you add the value passed to all y values of a histogram curve generated by API commands.

	Data Type	Explanation
Property value	System.Int32	Value that is added to the y values of the histogram curve

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim fixCurve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
fixCurve = histogram.CurveCollection.CurveByName("Availability")

fixCurve.Addend = 1
```


Example Code C#

```
VcHistogram histogram =  
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");  
VcCurve fixCurve = histogram.CurveCollection.CurveByName("Availability");  
  
fixCurve.Addend = 1;
```

FillReference1BackgroundColor

Property of VcCurve

This property lets you set or retrieve the color of the area between a histogram curve and the fill reference object set. You can also set this property in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values Default value: As defined in the Edit histogram dialog

Example Code VB.NET

```
Dim histogram As VcHistogram  
Dim curve As VcCurve  
  
histogram = vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")  
curve = histogram.CurveCollection.CurveByName("Curve1")  
  
curve.FillReference1BackgroundColor = Color.Blue
```

Example Code C#

```
VcHistogram histogram =  
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");  
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");  
  
curve.FillReference1BackgroundColor = Color.LightSteelBlue;
```

FillReference1Name

Property of VcCurve

This property lets you retrieve the name of the fill reference (for example a different curve or the x axis) of a histogram curve. The fill reference limits an area to be filled by colors and/or patterns. This property can also be set in the **Edit Histogram** dialog.

Note: The name of the x axis as fill reference has to be "VC_AXIS".

	Data Type	Explanation
Property value	System.String	Name of the reference curve

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")
curve.FillReferenceName = "VC_AXIS"
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

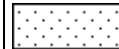
curve.FillReferenceName = "VC_AXIS";
```

FillReference1Pattern**Property of VcCurve**

This property lets you set or retrieve the fill pattern of the area between a histogram curve and the fill reference object set. You can also set this property in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	VcFillPattern	<p>Pattern type</p> <p>Default value: As defined in the Edit histogram dialog</p> <p>Possible Values:</p> <p>.vc05PercentPattern...</p> <p>vc90PercentPattern 01 - 11</p> <p>.vcAeroGlassPattern 44</p> <p>.vcBDiagonalPattern 5</p> <p>.vcCrossPattern 6</p>

Dots in foreground color on background color, the density of the foreground color increasing with the percentage



Vertical color gradient in the color of the fill pattern






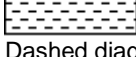
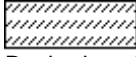
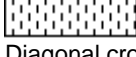

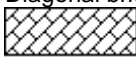
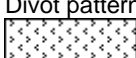

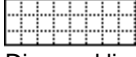



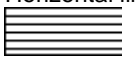


Diagonal lines slanting from bottom left to top right



Cross-hatch pattern



.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width 
.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 
.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width 
.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 
.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
.vcDashedHorizontalPattern 2026	Dashed horizontal lines 
.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 
.vcDashedVerticalPattern 2027	Dashed vertical lines 
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small 
.vcDiagonalBrickPattern 2032	Diagonal brick pattern 
.vcDivotPattern 2036	Divot pattern 
.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines 
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines 
.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right 
.vcHorizontalBrickPattern 2033	Horizontal brick pattern 
.vcHorizontalGradientPattern 52	Horizontal color gradient 
.vcHorizontalPattern 3	Horizontal lines 

.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern
.vcLargeConfettiPattern 2029	Confetti pattern, large
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern
.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern
.vcNoPattern 1276	No fill pattern
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large
.vcPlaidPattern 2035	Plaid pattern
.vcShinglePattern 2039	Diagonal shingle pattern
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern
.vcSmallConfettiPattern 2028	Confetti pattern
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern
.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares
.vcSpherePattern 2041	Checkerboard of spheres
.vcTrellisPattern 2040	Trellis pattern
.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright

.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright
.vcVerticalGradientPattern 62	Vertical color gradient
.vcVerticalPattern 2	Vertical lines
.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark
.vcWavePattern 2031	Horizontal waves pattern
.vcWeavePattern 2034	Interwoven stripes pattern
.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern
.vcWideUpwardDiagonalPattern 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of vcBDiagonalPattern
.vcZigZagPattern 2030	Horizontal zig-zag lines

Example Code VB.NET

```

Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

curve.FillReference1Pattern = VcFillPattern.vcCrossPattern

```

Example Code C#

```

VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

curve.FillReference1Pattern = VcFillPattern.vcDiagCrossPattern;

```

FillReference1PatternColor

Property of VcCurve

This property lets you set or retrieve the color of the pattern of the area between a histogram curve and the fill reference object set. You can also set this property in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255}) Default value: As defined in the Edit histogram dialog

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

curve.FillReference1PatternColor = Color.Blue
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

curve.FillReference1PatternColor = Color.LightSteelBlue;
```

FillReference2Color

Property of VcCurve

This property lets you set or retrieve the background color of pattern in the area above the second reference curve. The filling of the second reference curve will be displayed only if the values of the current curve are greater than those of the second reference curve.

You can also set this property in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values Default value: As defined in the Edit histogram dialog

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

curve.FillReference2BackgroundColor = Color.Blue
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

curve.FillReference2BackgroundColor = Color.LightSteelBlue;
```

FillReference2Name**Property of VcCurve**

This property lets you set or retrieve the name of the second reference curve of a curve. The area between the curve and its second reference curve specifies can be filled by a pattern. This property is set in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.String	Name of the 2nd reference curve

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve
Dim fillRef As Object

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")
fillRef = histogram.CurveCollection.CurveByName(curve.FillReference2Name)
```









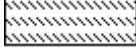
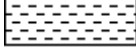



Example Code C#



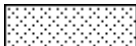
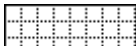



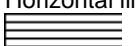
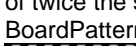
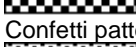
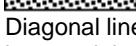

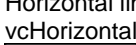
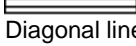


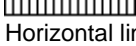


```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

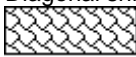


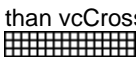


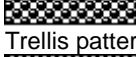
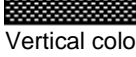


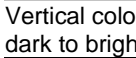






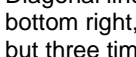
object fillRef =
histogram.CurveCollection.CurveByName(curve.FillReference2Name);
```

FillReference2Pattern**Property of VcCurve**

This property lets you set or retrieve the fill pattern of the area between a histogram curve and the second reference curve. You can also set this property in the **Edit Histogram** dialog.

Property value	Data Type	Explanation
	VcFillPattern	Pattern type
		Default value: As defined in the Edit histogram dialog
	Possible Values: .vc05PercentPattern... .vc90PercentPattern 01 - 11	Dots in foreground color on background color, the density of the foreground color increasing with the percentage 
	.vcAeroGlassPattern 44	Vertical color gradient in the color of the fill pattern 
	.vcBDiagonalPattern 5	Diagonal lines slanting from bottom left to top right 
	.vcCrossPattern 6	Cross-hatch pattern 
	.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width 
	.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 
	.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width 
	.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 
	.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
	.vcDashedHorizontalPattern 2026	Dashed horizontal lines 
	.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 
	.vcDashedVerticalPattern 2027	Dashed vertical lines 
	.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small 

.vcDiagonalBrickPattern 2032	Diagonal brick pattern 
.vcDivotPattern 2036	Divot pattern 
.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines 
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines 
.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right 
.vcHorizontalBrickPattern 2033	Horizontal brick pattern 
.vcHorizontalGradientPattern 52	Horizontal color gradient 
.vcHorizontalPattern 3	Horizontal lines 
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern 
.vcLargeConfettiPattern 2029	Confetti pattern, large 
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern 
.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern 
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern 
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern 
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern 
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern 
.vcNoPattern 1276	No fill pattern 
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large 
.vcPlaidPattern 2035	Plaid pattern 

.vcShinglePattern 2039	Diagonal shingle pattern 
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern 
.vcSmallConfettiPattern 2028	Confetti pattern 
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern 
.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares 
.vcSpherePattern 2041	Checkerboard of spheres 
.vcTrellisPattern 2040	Trellis pattern 
.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright 
.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark 
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright 
.vcVerticalGradientPattern 62	Vertical color gradient 
.vcVerticalPattern 2	Vertical lines 
.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark 
.vcWavePattern 2031	Horizontal waves pattern 
.vcWeavePattern 2034	Interwoven stripes pattern 
.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern 
.vcWideUpwardDiagonalPattern 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of vcBDiagonalPattern 
.vcZigZagPattern 2030	Horizontal zig-zag lines 

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

curve.FillReference2Pattern = VcFillPattern.vcCrossPattern
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

curve.FillReference2Pattern = VcFillPattern.vcDiagCrossPattern;
```

FillReference2PatternColor**Property of VcCurve**

This property lets you set or retrieve the foreground color of the pattern of the area above the second reference curve. The filling of the second reference curve will be displayed only if the values of the current curve are greater than those of the second reference curve.

You can also set this property in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255}) Default value: As defined in the Edit histogram dialog

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

curve.FillReference2PatternColor = Color.Blue
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

curve.FillReference2PatternColor = Color.LightSteelBlue;
```

FilterName

Property of VcCurve

This property lets you assign a filter to the curve or retrieve an existing one.

	Data Type	Explanation
Property value	System.String	Filter name

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

curve.FilterName = "Critical"
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

curve.FilterName = "Critical";
```

Histogram

Read Only Property of VcCurve

This property lets you retrieve the histogram, that the curve belongs to.

	Data Type	Explanation
Property value	VcHistogram	Histogram object

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

curve =
VcGantt1.HistogramCollection.FirstHistogram.CurveCollection.CurveByName("Curve1"
)
histogram = curve.Histogram
```

Example Code C#

```
VcCurve curve =
vcGantt1.HistogramCollection.FirstHistogram().CurveCollection.CurveByName("Curve
1");
VcHistogram histogram = curve.Histogram;
```

LayerName

Property of VcCurve

This property lets you assign a layer to the curve or retrieve the existing one.

	Data Type	Explanation
Property value	System.String	Name of the layer

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

curve.LayerName = "Start-End"
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

curve.LayerName = "Start-End";
```

LineColor

Property of VcCurve

This property lets you set or retrieve the line color of a histogram curve. This property you can also set in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values Default value: As defined in the Edit histogram dialog

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

curve.LineColor = Color.Blue
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

curve.LineColor = Color.LightSteelBlue;
```

LineThickness

Property of VcCurve

This property lets you set or retrieve the line thickness of a histogram curve.

If you set this property to values between 1 and 4, an absolute line thickness is defined in pixels. Irrespective of the zoom factor a line will always show the same line thickness in pixels. When printing though, the line thickness is adapted for the sake of legibility and becomes dependent of the zoom factor:

Value	Points	mm
1	1/2 point	0.09 mm
2	1 point	0.18 mm
3	3/2 points	0.26 mm
4	2 points	0.35 mm

A point equals 1/72 inch and represents the unit of the font size.

If you set this property to values between 5 and 1,000, the line thickness is defined in 1/100 mm, so the lines will be displayed in a true thickness in pixels that depends on the zoom factor.

This property also can be set in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.Int16	Line thickness LineType {1...4}: line thickness in pixels LineType {5...1000}: line thickness in 1/100 mm Default value: As defined in the Edit histogram dialog

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

curve.LineType = VcLineType.vcSolid
curve.LineThickness = 3

'or
curve.LineType = VcLineType.vcLineType5
curve.LineThickness = 20
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

curve.LineType = VcLineType.vcSolid;
curve.LineThickness = 3;

//or:
curve.LineType = VcLineType.vcLineType5;
curve.LineThickness = 20;
```

LineType

Property of VcCurve

This property lets you set or retrieve the line type of a histogram curve. If for stacked curves you do not wish the lines to be displayed, you can select **vcNone**. This property also can be set in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	VcLineType	Line type
		Default value: vcSolid
	Possible Values:	
	.vcDashed 4	Line dashed
	.vcDashed 4	Line dashed
	.vcDashedDotted 5	Line dashed-dotted
	.vcDashedDotted 5	Line dashed-dotted
	.vcDotted 3	Line dotted
	.vcDotted 3	Line dotted
	.vcLineType0 100	Line Type 0

	.vcLineType1 101	Line Type 1

	.vcLineType10 110	Line Type 10
	
	.vcLineType11 111	Line Type 11
	
	.vcLineType12 112	Line Type 12
	
	.vcLineType13 113	Line Type 13
	
	.vcLineType14 114	Line Type 14
	
	.vcLineType15 115	Line Type 15
	
	.vcLineType16 116	Line Type 16
	
	.vcLineType17 117	Line Type 17

	.vcLineType18 118	Line Type 18
	
	.vcLineType2 102	Line Type 2
	
	.vcLineType3 103	Line Type 3
	

.vcLineType4 104	Line Type 4
.vcLineType5 105	Line Type 5
.vcLineType6 106	Line Type 6
.vcLineType7 107	Line Type 7
.vcLineType8 108	Line Type 8
.vcLineType9 109	Line Type 9
.vcNone 1	No line type assigned
.vcNone 1	No line type
.vcSolid 2	Line solid
.vcSolid 2	Line solid

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

curve.LineType = VcLineType.vcSolid
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

curve.LineType = VcLineType.vcSolid;
```

Marked

Property of VcCurve

This property lets you set or retrieve the marking status of an histogram curve set by the API.

	Data Type	Explanation
Property value	System.Boolean	Curve marked/not marked

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim fixCurve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
fixCurve = histogram.CurveCollection.CurveByName("Availability")

fixCurve.Marked = True
```


Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve fixCurve = histogram.CurveCollection.CurveByName("Availability");

fixCurve.Marked = true;
```

Name**Read Only Property of VcCurve**

This property lets you retrieve the name of a histogram curve.

	Data Type	Explanation
Property value	System.String	Curve name

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve
Dim curveName As String

histogram = vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

curveName = curve.Name
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

string curveName = curve.Name;
```

OverloadResultsCalendarName**Property of VcCurve**

This property lets you set or retrieve a calendar to store the intervalls that have been calculated by the overload dates. You could use this calendar, for instance, to display a calendar grid in a group

	Data Type	Explanation
Property value	System.String	Name of overload results calendar object

PointsEquidistant

Property of VcCurve

This property lets you set or retrieve whether the curve points are to be equidistant. In case of **False**, the curve points will be created only in those points where the y values are changing. This property also can be set in the **Select Curve Data Source** dialog.

	Data Type	Explanation
Property value	System.Boolean	Curve points equidistant (True)/not equidistant (False)

Source

Property of VcCurve

This property lets you set or retrieve the source that the data of a histogram curve are taken from. You can set this property in the **Select Curve Data Source** dialog box. If **vcSetCurve** is returned (**Data specified manually** in the **Select Curve Data Source** dialog box), you can set the data in your application by the **SetValues** method. If **vcCalculateFromLayer** is returned (**Data generated by layer**), the data will be calculated from the layers.

	Data Type	Explanation
Property value	VcCurveSource	Calculation from field data, from dc data, from layer data, curve set
	Possible Values: .vcCalculateFromLayer 1 .vcSetCurve 3	Curve values calculated from layer Curve values are set manually

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve
Dim source As VcSource

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

source = curve.Source
```

Example Code C#

```
VcHistogram histogram =
VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");
VcSource source = curve.Source;
```

Specification

Read Only Property of VcCurve

This property lets you retrieve the specification of a curve. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored without problems to text files or data bases. This allows for persistency. A specification can be used to create a map by the method **VcCurveCollection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the curve

Example Code VB.NET

```
Dim curveCltn As VcCurveCollection
Dim curve As VcCurve

curveCltn = VcGantt1.CurveCollection
curve = curveCltn.FirstCurve
MsgBox(curve.Specification)
```

Example Code C#

```
VcCurveCollection curveCltn = vcGantt1.CurveCollection;
VcCurve curve = curveCltn.FirstCurve();
MessageBox.Show(curve.Specification);
```

StackReferenceName

Property of VcCurve

This property lets you set or retrieve the name of the stack reference curve of a histogram curve. The stack reference name has to be specified if curves are to be stacked. It specifies the curve onto which a different curve is to be stacked. You can also set this property in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.String	Name of the stack curve

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve
Dim referenceCurve As Object

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

referenceCurve = histogram.CurveCollection.CurveByName(curve.StackReferenceName)
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

object referencecurve =
histogram.CurveCollection.CurveByName(curve.StackReferenceName);
```

TimeUnit**Read Only Property of VcCurve**

This property lets you retrieve the time unit of a histogram curve. The property can be applied to equidistant curves that were generated by the API only. If applied to a curve generated from layer values, the property will return the result of -1. You can set the time unit on the property page **General**.

	Data Type	Explanation
Property value	VcTimeUnit	Time unit Default value: -1
	Possible Values: .vcDay 5 .vcHour 6 .vcMinute 7 .vcSecond 8	Time unit day Time unit hour Time unit minute Time unit second

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve
Dim timeUnit As VcTimeUnit

histogram = vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

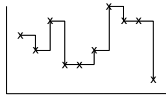
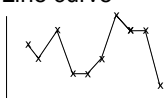
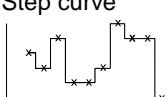
timeUnit = curve.TimeUnit
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");
VcTimeUnit timeUnit = curve.TimeUnit;
```

Type**Read Only Property of VcCurve**

This property lets you enquire the type of histogram curve.

	Data Type	Explanation
Property value	VcCurveType	Capacity curve Default value: vcCapacityCurve
	Possible Values: .vcCapacityCurve 215	Capacity curve 
	.vcLineCurve 214	Line curve 
	.vcStepCurve 216	Step curve 

Example Code VB.NET

```

Dim histogram As VcHistogram
Dim curve As VcCurve
Dim type As VcType

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")
type = curve.Type

```

Example Code C#

```

VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");
VcType type = curve.Type;

```

UnitsPerStep**Read Only Property of VcCurve**

This property lets you retrieve the number of units per step of a histogram curve. The property can be applied to equidistant curves that were generated by the API only. The number can be set on the property page **General**.

	Data Type	Explanation
Property value	System.Int16	Number of units Default value: -1

Example Code VB.NET

```

Dim histogram As VcHistogram
Dim curve As VcCurve
Dim unitsPerStep As Integer

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

unitsPerStep = curve.UnitsPerStep

```

Example Code C#

```

VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

int unitsPerStep = curve.UnitsPerStep;

```

UpdateBehaviorName**Property of VcCurve**

This property lets you set or retrieve the name of the UpdateBehavior.

	Data Type	Explanation
Property value	System.String	Name of the UpdateBehavior

ValencyDataFieldIndex**Property of VcCurve**

This property lets you set or retrieve the valency field of a curve generated by layer. The valency field is the data field from which for each activity the valency for the capacity sum is to be taken.

	Data Type	Explanation
Property value	System.Int16	Index of the valency field

Visible**Property of VcCurve**

This property lets you set or retrieve whether a curve is visible. You can also set this property on the **Administrate Histograms** dialog.

	Data Type	Explanation
Property value	System.Boolean	Curve visible/invisible Default value: True

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

curve.Visible = True
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

curve.Visible = true;
```

Methods

Clear

Method of VcCurve

This method lets you set all y values of a curve to zero. The method can be applied only to those curves the values of which were generated by the API.

	Data Type	Explanation
Return value	Void	

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim fixCurve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
fixCurve = histogram.CurveCollection.CurveByName("Availability")

fixCurve.Clear()
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve fixCurve = histogram.CurveCollection.CurveByName("Availability");

fixCurve.Clear();
```

DeletePoint

Method of VcCurve

This method lets you remove the curve point nearest to the x-coordinate.

	Data Type	Explanation
Parameter:		
⇒ x	System.Int32	X value of the curve point to be deleted
⇒ y	System.Int32	Y value of the curve point to be deleted
⇐ pointDate	System.DateTime	Date of the curve point which was deleted
Return value	System.Boolean	Curve point was/was not deleted successfully

Example Code VB.NET

```
Private Sub VcGantt1_VcCurveRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcCurveClickingEventArgs) Handles VcGantt1.VcCurveRightClicking

    Dim pointDate As Date
    Dim deleted As Boolean

    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup
    deleted = e.Curve.DeletePoint(e.X, e.Y, pointDate)
    If deleted = True Then
        Call MsgBox(pointDate)
    End If

End Sub
```

Example Code C#

```
private void vcGantt1_VcCurveRightClicking(object sender,
NETRONIC.XGantt.VcCurveClickingEventArgs e)
{
    DateTime pointDate = new DateTime();
    bool deleted;
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
    deleted = e.Curve.DeletePoint(e.X, e.Y, ref pointDate);
    if (deleted == true)
        MessageBox.Show(pointDate.ToString());
}
```

GetFirstOverload

Method of VcCurve

An **overload** is the area between the current curve and a reference curve with the former showing higher values than the latter. The reference curve is the curve defined as the second fill reference (**2nd Ref**) in the **Edit Histogram** dialog.

This method can be used to access the initial value, i.e. the first overload, and then to continue in a forward iteration loop by the method **GetNextOverload** for the overloads following.

Please note: For floating point numbers in the parameters **fromValue** and **toValue** please use the method **GetFirstOverloadEx**.

	Data Type	Explanation
Parameter:		
↩ fromDate	System.DateTime	Start date of the overload area
↩ fromValue	System.Int32	Y-value of the start date of the overload area
↩ toDate	System.DateTime	Final date of the overload area
↩ toValue	System.Int32	Y-value of the final date of the overload area
Return value	System.Boolean	Overload was/was not retrieved successfully

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve
Dim fixCurve As VcCurve
Dim fromDate As Date
Dim toDate As Date
Dim fromValue As Integer
Dim toValue As Integer
Dim yValues As String
Dim bOk As Boolean

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("LayerCurve")
fixCurve = histogram.CurveCollection.CurveByName("FixCurve")
yValues = "6;6;6;6;6;0;0;6;6;6;6;6;0;0;6;6;6;6;6;0;0;6;6;6;6;6;0;0;6;6;6;6;6"
fixCurve.SetValues("31.08.14", yValues)
bOk = curve.GetFirstOverload(fromDate, fromValue, toDate, toValue)
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("LayerCurve");
VcCurve fixCurve = histogram.CurveCollection.CurveByName("FixCurve");
DateTime fromDate = new DateTime();
DateTime toDate = new DateTime();
int fromValue = 0;
int toValue = 0;
string yValues =
"6;6;6;6;6;0;0;6;6;6;6;6;0;0;6;6;6;6;6;0;0;6;6;6;6;6;0;0;6;6;6;6;6";
fixCurve.SetValues(Convert.ToDateTime("31.08.14"), yValues);
bool bOk = curve.GetFirstOverload(ref fromDate, ref fromValue, ref toDate, ref
toValue);
```

GetFirstOverloadEx

Method of VcCurve

An **overload** is the area between the current curve and a reference curve with the former showing higher values than the latter. The reference curve is the curve defined as the second fill reference (**2nd Ref**) in the **Edit Histogram** dialog.

This method can be used to access the initial value, i.e. the first overload, and then to continue in a forward iteration loop by the method **GetNextOverloadEx** for the overloads following.

Please note: Compared to the method **GetFirstOverload** this method allows for floating point numbers in the parameters **fromValue** and **toValue**.

	Data Type	Explanation
Parameter:		
⇐ fromDate	System.DateTime	Start date of the overload area
⇐ fromValue	System.Double	Y-value of the start date of the overload area
⇐ toDate	System.DateTime	Final date of the overload area
⇐ toValue	System.Double	Y-value of the final date of the overload area
Return value	System.Boolean	Overload was/was not retrieved successfully

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve
Dim fixCurve As VcCurve
Dim fromDate As Date
Dim toDate As Date
Dim fromValue As Integer
Dim toValue As Integer
Dim yValues As String
Dim bOk As Boolean

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("LayerCurve")
fixCurve = histogram.CurveCollection.CurveByName("FixCurve")
yValues = "6;6;6;6;6;0;0;6;6;6;6;6;0;0;6;6;6;6;0;0;6;6;6;6;0;0;6;6;6;6;6;0;0;6;6;6;6;6;0;0;6;6;6;6;6;"
fixCurve.SetValues("31.08.14", yValues)
bOk = curve.GetFirstOverload(fromDate, fromValue, toDate, toValue)
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("LayerCurve");
VcCurve fixCurve = histogram.CurveCollection.CurveByName("FixCurve");
DateTime fromDate = new DateTime();
DateTime toDate = new DateTime();
int fromValue = 0;
int toValue = 0;
string yValues =
"6;6;6;6;6;0;0;6;6;6;6;6;0;0;6;6;6;6;6;0;0;6;6;6;6;6;0;0;6;6;6;6;6";
fixCurve.SetValues(Convert.ToDateTime("31.08.14"),yValues);
bool bOk = curve.GetFirstOverload(ref fromDate, ref fromValue, ref toDate, ref
toValue);
```

GetNextOverload

Method of VcCurve

An **overload** is the area between the current curve and a reference curve with the former showing higher values than the latter. The reference curve is the curve defined as the second fill reference (**2nd Ref**) in the **Edit Histogram**.

This method can be used in a forward iteration loop to retrieve subsequent calendars from a calendar collection after initializing the loop by the method **GetFirstOverload**.

Please note: For floating point numbers in the parameters **fromValue** and **toValue** please use the method **GetNextOverloadEx**.

	Data Type	Explanation
Parameter:		
↩ fromDate	System.DateTime	Start date of the overload area
↩ fromValue	System.Int32	Y-value of the start date of the overload area
↩ toDate	System.DateTime	Final date of the overload area
↩ toValue	System.Int32	Y-value of the final date of the overload area
Return value	System.Boolean	Overload was/was not retrieved successfully.

Example Code VB.NET

```

...
Dim bOk As Boolean

bOk = curve.GetFirstOverload(fromDate, fromValue, toDate, toValue)
MsgBox(fromDate.ToString() + " ( " + fromValues.ToString() + " ) - " +
toDate.ToString() + " ( " + toValues.ToString() + " ) ")
While bOk
    bOk = curve.GetNextOverload(fromDate, fromValue, toDate, tovalue)
    If bOk Then
        MsgBox(fromDate.ToString() + " ( " + fromValues.ToString() + " )
- " + toDate.ToString() + " ( " + toValues.ToString() + " ) ")
    End If
End While

```

Example Code C#

```

...
bool bOk = curve.GetFirstOverload(ref fromDate, ref fromValue, ref toDate, ref
toValue);
MessageBox.Show(fromDate.ToString() + " ( " + fromValue.ToString() + " ) - " +
toDate.ToString() + " ( " + toValue.ToString() + " )");
while (bOk == true)
{
    bOk = curve.GetNextOverload(ref fromDate, ref fromValue, ref toDate, ref
toValue);
    if (bOk == true)
        MessageBox.Show(fromDate.ToString() + " ( " + fromValue.ToString() + " ) -
+ toDate.ToString() + " ( " + toValue.ToString() + " )");
}

```

GetNextOverloadEx**Method of VcCurve**

An **overload** is the area between the current curve and a reference curve with the former showing higher values than the latter. The reference curve is the curve defined as the second fill reference (**2nd Ref**) in the **Edit Histogram**.

This method can be used in a forward iteration loop to retrieve subsequent overloads from an overload collection after initializing the loop by the method **GetFirstOverloadEx**.

Please note: Compared to the method **GetNextOverload** this method allows for floating point numbers in the parameters **fromValue** and **toValue**.

	Data Type	Explanation
Parameter:		
⇐ fromDate	System.DateTime	Start date of the overload area
⇐ fromValue	System.Double	Y-value of the start date of the overload area
⇐ toDate	System.DateTime	Final date of the overload area
⇐ toValue	System.Double	Y-value of the final date of the overload area

Return value	System.Boolean	Overload was/was not retrieved successfully.
--------------	----------------	--

Example Code VB.NET

```

...
Dim bOk As Boolean

bOk = curve.GetFirstOverload(fromDate, fromValue, toDate, toValue)
MsgBox(fromDate.ToString() + " ( " + fromValues.ToString() + " ) - " +
toDate.ToString() + " ( " + toValues.ToString() + " ) ")
While bOk
    bOk = curve.GetNextOverload(fromDate, fromValue, toDate, tovalue)
    If bOk Then
        MsgBox(fromDate.ToString() + " ( " + fromValues.ToString() + " )
- " + toDate.ToString() + " ( " + toValues.ToString() + " ) ")
    End If
End While

```

Example Code C#

```

...
bool bOk = curve.GetFirstOverload(ref fromDate, ref fromValue, ref toDate, ref
toValue);
MessageBox.Show(fromDate.ToString() + " ( " + fromValue.ToString() + " ) - " +
toDate.ToString() + " ( " + toValue.ToString() + " )");
while (bOk == true)
{
    bOk = curve.GetNextOverload(ref fromDate, ref fromValue, ref toDate, ref
toValue);
    if (bOk == true)
        MessageBox.Show(fromDate.ToString() + " ( " + fromValue.ToString() + " ) -
" + toDate.ToString() + " ( " + toValue.ToString() + " )");
}

```

GetValues**Method of VcCurve**

This method lets you retrieve the value of a histogram curve that belongs to a specified date. Since the date specified may not be located in a defined point (pair of coordinates) of the curve, the date and value of the closest defined point before resp. after the specified date will be returned. If a point was hit exactly, its corresponding value will be returned two times i.e. as previous and next value.

	Data Type	Explanation
Parameter:		
⇒ inputDate	System.DateTime	Date that the value of the histogram curve is to be retrieved
⇐ leftDate	System.DateTime	Date of the last defined point of the curve before the specified date
⇐ leftValue	System.Int32	Value of the last defined point of the curve before the specified date

⇨ rightDate	System.DateTime	Date of the next defined point of the curve after the specified date
⇨ rightValue	System.Int32	Value of the next defined point of the curve after the specified date
Return value	void	

Example Code VB.NET

```

Dim histogram As VcHistogram
Dim curve As VcCurve
Dim inputDate As String
Dim leftDate As Date
Dim rightDate As Date
Dim leftValue As Integer
Dim rightValue As Integer

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("LayerCurve")
inputDate = InputBox("Date: ")
curve.GetValues(inputDate, leftDate, leftValue, rightDate, rightValue)
MsgBox(leftDate.ToString() & " ( " & leftValue.ToString() & " ) " &
rightDate.ToString() & " ( " & rightValue.ToString() & " ) ")

```

Example Code C#

```

DateTime leftDate = new DateTime();
DateTime rightDate = new DateTime();
int leftValue = 0;
int rightValue = 0;

VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("LayerCurve");
curve.GetValues(Convert.ToDateTime("01.05.2014"), ref leftDate, ref leftValue,
ref rightDate, ref rightValue);
MessageBox.Show(leftDate.ToString() + " ( " + leftValue.ToString() + " ) " +
rightDate.ToString() + " ( " + rightValue.ToString() + " ) ");

```

GetValuesEx**Method of VcCurve**

This method lets you retrieve the value of a histogram curve that belongs to a specified date. Compared to the method **GetValues** this method is appropriate for floating point values. Since the date specified may not be located in a defined point (pair of coordinates) of the curve, the date and value of the closest defined point before and after the specified date will be returned. If a point was hit exactly, its corresponding value will be returned twice, i.e. as the previous and the following value.

	Data Type	Explanation
Parameter:		
⇨ inputDate	System.DateTime	Date that the value of the histogram curve is to be retrieved

↩ leftDate	System.DateTime	Date of the last defined point of the curve before the specified date
↩ leftValue	System.Double	Value of the last defined point of the curve before the specified date
↩ rightDate	System.DateTime	Date of the next defined point of the curve after the specified date
↩ rightValue	System.Double	Value of the next defined point of the curve after the specified date
Return value	void	

Example Code VB.NET

```

Dim histogram As VcHistogram
Dim curve As VcCurve
Dim inputDate As String
Dim leftDate As Date
Dim rightDate As Date
Dim leftValues As Double
Dim rightValues As Double

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("LayerCurve")
inputDate = InputBox("Date: ")
curve.GetValuesEx(inputDate, leftDate, leftValues, rightDate, rightValues)
MsgBox(leftDate.ToString() + " ( " + leftValues.ToString() + " ) " +
rightDate.ToString() + " ( " + rightValues.ToString() + " ) ")

```

Example Code C#

```

DateTime leftDate = new DateTime();
DateTime rightDate = new DateTime();
double leftValue = 0;
double rightValue = 0;

VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("LayerCurve");
curve.GetValuesEx(Convert.ToDateTime("01.05.2009"), ref leftDate, ref
leftValues, ref rightDate, ref rightValues);
MessageBox.Show(leftDate.ToString() + " ( " + leftValues.ToString() + " ) " +
rightDate.ToString() + " ( " + rightValues.ToString() + " ) ");

```

SetValues**Method of VcCurve**

This method lets you set the values of a histogram curve that was generated by the API. A curve built by **SetValues** can be used as a capacity curve to display engine resources or be used as a reference curve.

The usage of the VcCurve.SetValues method depends on the **Curve points equidistant** check box in the **Select Curve Data Source** dialog box:

Curve points equidistant: You can transfer a start value (**startValue**) and a string separated by semicolons that contains the y values. The coordinates of points that form the curve are calculated from the start value and the y values, combined with the **Time Unit** and **Smallest time interval** (property page **General**). Curves generated in this way cannot be edited interactively.

Curve points not equidistant: You have to call the method for each pair of (x,y) values. The **Time Unit** and **Smallest time interval** are not relevant. The curve can be edited interactively.

	Data Type	Explanation
Parameter:		
⇒ startDate	System.DateTime	Start date
⇒ values	System.String	Y values as a string
Return value	System.Boolean	Values were/were not set successfully

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve
Dim yValues As String

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curve = histogram.CurveCollection.CurveByName("Curve1")

' If the option Curve points equidistant is checked for the curve:
yValues = "5;1;1;2;2;2;4;5;5;3;2;1;"
curve.SetValues("01.05.2014", yValues)

' If the option Curve points equidistant is not checked for the curve:
curve.SetValues("01.05.2014", 5)
curve.SetValues("03.05.2014", 1)
curve.SetValues("07.05.2014", 1)
curve.SetValues("16.05.2014", 2)
```

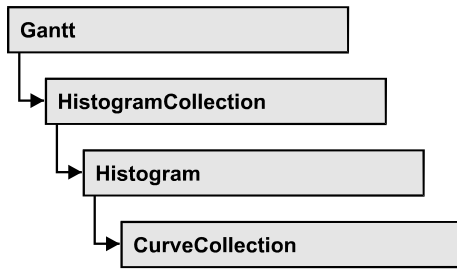
Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HiSTOGRAM_1");
VcCurve curve = histogram.CurveCollection.CurveByName("Curve1");

//If the option Curve points equidistant is checked for the curve:
string yValues = "5;1;1;2;2;2;4;5;5;3;2;1;";
curve.SetValues(Convert.ToDateTime("01.05.2014"), yValues);

//If the option Curve points equidistant is not checked for the curve:
curve.SetValues(Convert.ToDateTime("01.05.2014"), "5");
curve.SetValues(Convert.ToDateTime("03.05.2014"), "1");
curve.SetValues(Convert.ToDateTime("07.05.2014"), "1");
curve.SetValues(Convert.ToDateTime("16.05.2014"), "2");
```


7.16 VcCurveCollection



An object of the type `VcCurveCollection` automatically contains all curves of the histogram. You can access all objects in an iterative loop by **For Each curve In CurveCollection** or by the methods **First...** and **Next...**. You can access a single curve using the methods **CurveByName** and **CurveByIndex**. The number of curves in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the curves in the corresponding way.

Properties

- Count

Methods

- Add
- AddBySpecification
- Copy
- CurveByIndex
- CurveByName
- FirstCurve
- GetEnumerator
- NextCurve
- Remove

Properties

Count

Read Only Property of VcCurveCollection

This property lets you retrieve the number of curves in the `CurveCollection`.

	Data Type	Explanation
Property value	System.Int32	Number of curves

Example Code VB.NET

```

Dim histogram As VcHistogram
Dim curveCltn As VcCurveCollection
Dim numberOfCurves As Integer

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curveCltn = histogram.CurveCollection

numberOfCurves = curveCltn.Count

```

Example Code C#

```

VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurveCollection curveCltn = histogram.CurveCollection;

int numberOfCurves = curveCltn.Count;

```

Methods

Add

Method of VcCurveCollection

By this method you can create a curve as a member of the CurveCollection. If the name has not been used before, the new curve object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ curveName	System.String	Curve name
Return value	VcCurve	New curve object

Example Code VB.NET

```

newCurve =
VcGantt1.HistogramCollection.HistogramByName("a").CurveCollection.Add("test1")

```

Example Code C#

```

newCurve =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").CurveCollection.Add(
"test1");

```

AddBySpecification

Method of VcCurveCollection

This method lets you create a curve by using a curve specification. This way of creating allows curve objects to become persistent. The specification of a curve can be saved and re-loaded (see VcCurve property **Specification**) In a subsequent session the curve can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter: ⇒ specification	System.String	Curve specification
Return value	VcCurve	New curve object

Copy

Method of VcCurveCollection

By this method you can copy a curve. If the curve that is to be copied exists, and if the name for the new curve does not yet exist, the new curve object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ curveName	System.String	Name of the curve to be copied
⇒ newCurveName	System.String	Name of the new curve
Return value	VcCurve	Curve object

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curveCltn As VcCurveCollection

histogram = VcGantt1.HistogramCollection.FirstHistogram
curveCltn = histogram.CurveCollection
curveCltn.Copy("CurrentCurve", "NewCurve")
```

Example Code C#

```
VcHistogram histogram = vcGantt1.HistogramCollection.FirstHistogram();
VcCurveCollection curveCltn = histogram.CurveCollection;
curveCltn.Copy("CurrentCurve", "NewCurve");
```

CurveByIndex

Method of VcCurveCollection

This method lets you access a curve by its index. If a curve does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the curve
Return value	VcCurve	Curve object returned

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curveCltn As VcCurveCollection
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curveCltn = histogram.CurveCollection

curve = curveCltn.CurveByIndex(2)
```

Example Code C#

```
VcHistogram histogram =
VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurveCollection curveCltn = histogram.CurveCollection;
VcCurve curve = curveCltn.CurveByIndex(2);
```

CurveByName

Method of VcCurveCollection

By this method you can retrieve a curve by its name. If a curve of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ curveName	System.String	Name of the curve
Return value	VcCurve	Curve

Example Code VB.NET

```

Dim histogram As VcHistogram
Dim curveCltn As VcCurveCollection
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curveCltn = histogram.CurveCollection

curve = curveCltn.CurveByName("Curve1")

```

Example Code C#

```

VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurveCollection curveCltn = histogram.CurveCollection;

VcCurve curve = curveCltn.CurveByName("Curve1");

```

FirstCurve

Method of VcCurveCollection

This method can be used to access the initial value, i.e. the first curve of a CurveCollection, and to continue in a forward iteration loop by the method **NextCurve** for the curves following. If there is no curve in the CurveCollection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcCurve	First curve

Example Code VB.NET

```

Dim histogram As VcHistogram
Dim curveCltn As VcCurveCollection
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curveCltn = histogram.CurveCollection

curve = curveCltn.FirstCurve

```

Example Code C#

```

VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurveCollection curveCltn = histogram.CurveCollection;
VcCurve curve = curveCltn.FirstCurve;

```

GetEnumerator

Method of VcCurveCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the curve objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.FirstHistogram
For Each curve In histogram.CurveCollection
    ListBox1.Items.Add(curve.Name)
Next
```

Example Code C#

```
VcHistogram histogram = vcGantt1.HistogramCollection.FirstHistogram();
foreach (VcCurve curve in histogram.CurveCollection)
    listBox1.Items.Add(curve.Name);
```

NextCurve**Method of VcCurveCollection**

This method can be used in a forward iteration loop to retrieve subsequent curves from a curve collection after initializing the loop by the method **FirstCurve**. If there is no curve left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcCurve	Succeeding Curve

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim curveCltn As VcCurveCollection
Dim curve As VcCurve

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
curveCltn = histogram.CurveCollection

curve = curveCltn.FirstCurve

While Not newCurve Is Nothing
    newCurve = curveCltn.NextCurve
End While
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcCurveCollection curveCltn = histogram.CurveCollection;
VcCurve curve = curveCltn.FirstCurve();

while (curve == null)
{
    curve = curveCltn.NextCurve();
}
```

Remove

Method of VcCurveCollection

This method lets you delete a curve. If the curve is used in another object, it cannot be deleted. Then False will be returned, otherwise True.

	Data Type	Explanation
Parameter:		
⇒ curveName	System.String	Curve name
Return value	System.Boolean	Curve deleted (True)/not deleted (False)

Example Code VB.NET

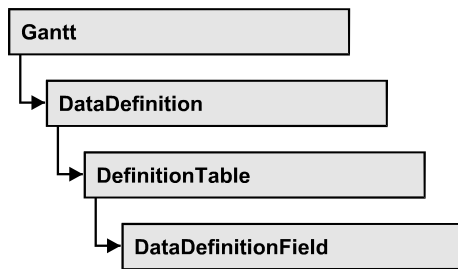
```
Dim histogram As VcHistogram
Dim curveCltn As VcCurveCollection

histogram = VcGantt1.HistogramCollection.FirstHistogram
curveCltn = histogram.CurveCollection
curveCltn.Remove("CurrentCurve")
```

Example Code C#

```
VcHistogram histogram = vcGantt1.HistogramCollection.FirstHistogram();
VcCurveCollection curveCltn = histogram.CurveCollection;
curveCltn.Remove("CurrentCurve");
```

7.17 VcDataDefinitionField



An object of the type VcDefinitionField defines a field of the data definition table. The definition basically consists of a name and a data type.

Properties

- DateFormat
- Editable
- Hidden
- Index
- Name
- Type

Properties

DateFormat

Property of VcDataDefinitionField

This property lets you set or retrieve the date format of the field of a data definition table. This property only works if the data type of the field was set to **vcDataTableFieldDateTime**. The dateFormat setting is used when reading or storing CSV files and when the format type **String** is used when adding a data record by the methods **InsertNodeRecord** or **InsertLinkRecord**. The format of the date output in the chart is controlled by the property **DateOutputFormat**.

Note: You should set the property Type first before setting the property DateFormat.

	Data Type	Explanation
Property value	System.String	Date format {DMYhms:;./} Default value: bei vcDefFieldDateTime DD.MM.YYYY hh:mm:ss

Example Code VB.NET

```
Dim dataDefTable As VcDataDefinitionTable
Dim dataDefField As VcDataDefinitionField

dataDefTable =
VcGantt1.DataDefinition.DataDefinitionTable(VcDataTableType.vcMaindata)
dataDefField = dataDefTable.DataDefinitionFieldByName("Start")
dataDefField.Type = VcDataDefinitionFieldType.vcDefFieldDateTimeType
'DateFormat = "01.12.2014"
dataDefField.DateFormat = "DD.MM.YYYY"
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataDefinitionTable dataDefTable =
vcGantt1.DataDefinition.get_DataDefinitionTable(VcDataTableType.vcMaindata);
VcDataDefinitionField dataDefField =
dataDefTable.DataDefinitionFieldByName("Start");
dataDefField.Type = VcDataDefinitionFieldType.vcDefFieldDateTimeType;
//DateFormat = "01.12.2014"
dataDefField.DateFormat = "DD.MM.YYYY";
vcGantt1.DataTableCollection.Update();
```

Editable**Property of VcDataDefinitionField**

This property lets you set or retrieve whether the data field should be editable at run time in the chart table and in the dialog **EditNode**.

	Data Type	Explanation
Property value	System.Boolean	Definition field editable/not editable Default value: True

Example Code VB.NET

```
Dim dataDefTable As VcDataDefinitionTable
Dim dataDefField As VcDataDefinitionField

dataDefTable =
VcGantt1.DataDefinition.DataDefinitionTable(VcDataTableType.vcMaindata)
dataDefField = dataDefTable.DataDefinitionFieldByName("Start")
dataDefField.Editable = False
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataDefinitionTable dataDefTable =
vcGantt1.DataDefinition.get_DataDefinitionTable(VcDataTableType.vcMaindata);
VcDataDefinitionField dataDefField =
dataDefTable.DataDefinitionFieldByName("Start");
dataDefField.Editable = false;
vcGantt1.DataTableCollection.Update();
```

Hidden**Property of VcDataDefinitionField**

This property lets you require/set whether a data field is hidden at run time.

	Data Type	Explanation
Property value	System.Boolean	Definition field hidden/not hidden Default value: False

Example Code VB.NET

```
Dim dataDefTable As VcDataDefinitionTable
Dim dataDefField As VcDataDefinitionField

dataDefTable =
VcGantt1.DataDefinition.DataDefinitionTable(VcDataTableType.vcMaindata)
dataDefField = dataDefTable.DataDefinitionFieldByName("Start")
dataDefField.Hidden = True
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataDefinitionTable dataDefTable =
vcGantt1.DataDefinition.get_DataDefinitionTable(VcDataTableType.vcMaindata);
VcDataDefinitionField dataDefField =
dataDefTable.DataDefinitionFieldByName("Start");
dataDefField.Hidden = true;
vcGantt1.DataTableCollection.Update();
```

Index**Read Only Property of VcDataDefinitionField**

This property lets you retrieve the index of the field of a data definition table.

	Data Type	Explanation
Property value	System.Int16	Index of the definition field

Example Code VB.NET

```
Dim dataDefTable As VcDataDefinitionTable
Dim dataDefField As VcDataDefinitionField

dataDefTable =
VcGantt1.DataDefinition.DataDefinitionTable(VcDataTableType.vcMaindata)
dataDefField = dataDefTable.DataDefinitionFieldByName("Start")
MsgBox(dataDefField.Index.ToString())
```

Example Code C#

```
VcDataDefinitionTable dataDefTable =
vcGantt1.DataDefinition.get_DataDefinitionTable(VcDataTableType.vcMaindata);
VcDataDefinitionField dataDefField =
dataDefTable.DataDefinitionFieldByName("Start");
MessageBox.Show(dataDefField.Index.ToString());
```

Name

Property of VcDataDefinitionField

This property lets you set or retrieve the name of the field of a data definition table.

	Data Type	Explanation
Property value	System.String	Name of the definition field

Example Code VB.NET

```
Dim dataDefTable As VcDataDefinitionTable
Dim dataDefField As VcDataDefinitionField

dataDefTable =
VcGantt1.DataDefinition.DataDefinitionTable(VcDataTableType.vcMaindata)
dataDefField = dataDefTable.CreateDataDefinitionField("Start")
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataDefinitionTable dataDefTable =
vcGantt1.DataDefinition.get_DataDefinitionTable(VcDataTableType.vcMaindata);
VcDataDefinitionField dataDefField =
dataDefTable.CreateDataDefinitionField("Start");
vcGantt1.DataTableCollection.Update();
```

Type

Property of VcDataDefinitionField

This property lets you set or retrieve the type of the field of a data definition table.

Note: By setting the property **Type** the property **DateFormat** will change!

`vcDefFieldAlphanumericType: DateFormat = ""`

`vcDefFieldDateTimeType: DateFormat = "DD.MM.YYYY hh:mm:ss"`

`vcDefFieldIntegerType: DateFormat = ""`

	Data Type	Explanation
Property value	VcDataDefinitionFieldType	Type of the definition field Default value: vcDefFieldIntegerType
	Possible Values: .vcDefFieldAlphanumericType 1 .vcDefFieldDateTimeType 4 .vcDefFieldIntegerType 2	Data type alphanumeric Data type date Data type integer (32 bits)

Example Code VB.NET

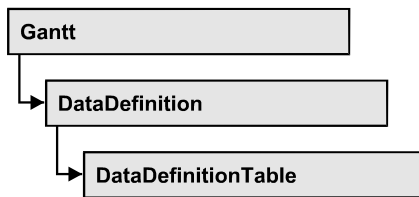
```
Dim dataDefTable As VcDataDefinitionTable
Dim dataDefField As VcDataDefinitionField

dataDefTable =
VcGantt1.DataDefinition.DataDefinitionTable(VcDataTableType.vcMaindata)
dataDefField = dataDefTable.DataDefinitionFieldByName("Start")
dataDefField.Type = VcDataDefinitionFieldType.vcDefFieldDateTimeType
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataDefinitionTable dataDefTable =
vcGantt1.DataDefinition.get_DataDefinitionTable(VcDataTableType.vcMaindata);
VcDataDefinitionField dataDefField =
dataDefTable.DataDefinitionFieldByName("Start");
dataDefField.Type = VcDataDefinitionFieldType.vcDefFieldDateTimeType;
vcGantt1.DataTableCollection.Update();
```

7.18 VcDataDefinitionTable



A **VcDataDefinitionTable** object is an element of a data definition. It represents a table of data definition fields. You can access these fields individually by the methods **DataDefinitionFieldByIndex** or **DataDefinitionFieldByName** or retrieve them in an iterative loop by the methods **FirstDataDefinitionField** and **NextDataDefinitionField**. By the **Count** property you can enquire the number of the fields of the table. You can set data field definitions on the property page **Administrate Data Tables**.

Properties

- Count

Methods

- CreateDataDefinitionField
- DataDefinitionFieldByIndex
- DataDefinitionFieldByName
- FirstDataDefinitionField
- GetEnumerator
- NextDataDefinitionField

Properties

Count

Read Only Property of VcDataDefinitionTable

This property lets you retrieve the number of fields in the data definition table. You can add fields by the **Administrate Data Tables** dialog or at run time by the method **CreateDataDefinitionField**.

	Data Type	Explanation
Property value	System.Int32	Number of fields

Example Code VB.NET

```

Dim dataDefinition As VcDataDefinition
Dim dataDefinitionTable As VcDataDefinitionTable
Dim numberOfFields As Integer

dataDefinition = VcGantt1.DataDefinition
dataDefinitionTable = dataDefinition.DefinitionTable(VcDataTableType.vcMaindata)

numberOfFields = dataDefinitionTable.Count

```

Example Code C#

```

VcDataDefinition dataDefinition = vcGantt1.DataDefinition;
VcDataDefinitionTable dataDefinitionTable =
dataDefinition.get_DefinitionTable(VcDataTableType.vcMaindata);

int numberOfFields = dataDefinitionTable.Count;

```

Methods

CreateDataDefinitionField

Method of VcDataDefinitionTable

This method lets you add a new data field to the end of the data definition table at run time. The data field of the new data field is Integer. You can change the data type by the property **Type** of **VcDataDefinitionField**.

	Data Type	Explanation
Parameter: ⇒ newfieldName	System.String	Name of the new field
Return value	VcDataDefinitionField	Data definition field

Example Code VB.NET

```

Dim dataDefinition As VcDataDefinition
Dim dataDefinitionTable As VcDataDefinitionTable

dataDefinition = VcGantt1.DataDefinition
dataDefinitionTable = dataDefinition.DefinitionTable(VcDataTableType.vcMaindata)
dataDefinitionTable.CreateDataDefinitionField("New data field 1")
VcGantt1.DataTableCollection.Update()

```

Example Code C#

```
VcDataDefinition dataDefinition = VcGantt1.DataDefinition;
VcDataDefinitionTable dataDefinitionTable =
dataDefinition.get_DefinitionTable(VcDataTableType.vcMaindata);
dataDefinitionTable.CreateDataDefinitionField("New data field 1");
vcGantt1.DataTableCollection.Update();
```

DataDefinitionFieldByIndex**Method of VcDataDefinitionTable**

By this method you can access a field of the data definition table by its index. A field can be referred to by its name or by its index. You can edit data definitions in the **Administrate Data Tables** dialog.

	Data Type	Explanation
Parameter: ⇒ fieldIndex	System.Int16	Field index
Return value	VcDataDefinitionField	Data definition field

Example Code VB.NET

```
Dim dataDefinitionTable As VcDataDefinitionTable
Dim definitionField As VcDataDefinitionField

dataDefinitionTable =
VcGantt1.DataDefinition.DefinitionTable(VcDataTableType.vcMaindata)
definitionField = dataDefinitionTable.DataDefinitionFieldByIndex(2)
```

Example Code C#

```
VcDataDefinitionTable dataDefinitionTable =
VcGantt1.DataDefinition.get_DefinitionTable(VcDataTableType.vcMaindata);VcDataDe
finitionField dataDefinitionField =
dataDefinitionTable.DataDefinitionFieldByIndex(2);
```

DataDefinitionFieldByName**Method of VcDataDefinitionTable**

By this method you can access a field of the data definition table by its name. If a field of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic). A field can be referred to by its name or by its index. You can edit data definitions in the **Administrate Data Tables** dialog.

	Data Type	Explanation
Parameter: ⇒ fieldName	System.String	Field name
Return value	VcDataDefinitionField	Data definition field

Example Code VB.NET

```

Dim dataDefinitionTable As VcDataDefinitionTable
Dim definitionField As VcDataDefinitionField

dataDefinitionTable =
VcGantt1.DataDefinition.DefinitionTable(VcDataTableType.vcMaindata)
definitionField = dataDefinitionTable.DataDefinitionFieldByName("Start")

```

Example Code C#

```

VcDataDefinitionTable dataDefinitionTable =
VcGantt1.DataDefinition.get_DefinitionTable(VcDataTableType.vcMaindata);VcDataDe
finitionField dataDefinitionField =
dataDefinitionTable.DataDefinitionFieldByName("Start");

```

FirstDataDefinitionField**Method of VcDataDefinitionTable**

This method can be used to access the first field of a data definition table and to continue in a forward iteration loop by the method **NextDataDefinitionField** for the fields following. If there is no field in the data definition table, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcDataDefinitionField	First Data definition field

Example Code VB.NET

```

Dim dataDefinition As VcDataDefinition
Dim dataDefinitionTable As VcDataDefinitionTable
Dim dataDefinitionField As VcDataDefinitionField

Set dataDefinition = VcGantt1.DataDefinition
Set dataDefinitionTable = dataDefinition.DefinitionTable(vcMaindata)

Set dataDefinitionField = dataDefinitionTable.FirstDataDefinitionField

```

Example Code C#

```

VcDataDefinition dataDefinition = vcGantt1.DataDefinition;
VcDataDefinitionTable dataDefinitionTable =
vcGantt1.DataDefinition.get_DefinitionTable(VcDataTableType.vcMaindata);
VcDataDefinitionField dataDefinitionField =
dataDefinitionTable.FirstDataDefinitionField();

```


GetEnumerator

Method of VcDataDefinitionTable

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the fields of a dataDefinitionTable.

	Data Type	Explanation
Return value	VcObject	Enumerator object

Example Code VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcGantt1.DataTableCollection
For Each dataTable In dataTableCltn
    ListBox1.Items.Add(dataTable.Name)
Next
```

Example Code C#

```
VcDataTableCollection dataTableCltn = vcGantt1.DataTableCollection;
foreach (VcDataTable dataTable in dataTableCltn)
    listBox1.Items.Add(dataTable.Name);
```

NextDataDefinitionField

Method of VcDataDefinitionTable

This method can be used in a forward iteration loop to retrieve subsequent fields from a data definition table after initializing the loop by the method **FirstDataDefinitionField**. If there is no field left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcDataDefinitionField	Subsequent data definition field

Example Code VB.NET

```

Dim dataDefinition As VcDataDefinition
Dim dataDefinitionTable As VcDataDefinitionTable
Dim definitionField As VcDataDefinitionField

dataDefinition = VcGantt1.DataDefinition
dataDefinitionTable =
VcGantt1.DataDefinition.DefinitionTable(VcDataTableType.vcMaindata)
definitionField = dataDefinitionTable.FirstDataDefinitionField

While Not definitionField Is Nothing
    ListBox1.Items.Add(definitionField.Name)
    definitionField = dataDefinitionTable.NextField
End While

```

Example Code C#

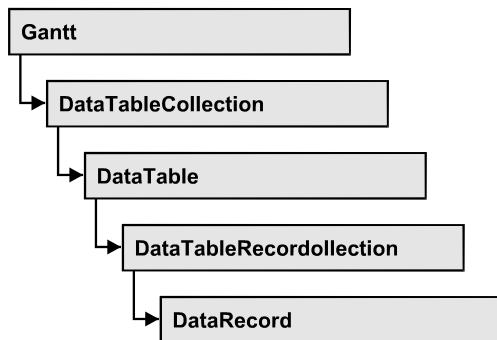
```

VcDataDefinition dataDefinition = vcGantt1.DataDefinition;
VcDataDefinitionTable dataDefinitionTable =
vcGantt1.DataDefinition.get_DefinitionTable(VcDataTableType.vcMaindata);
VcDataDefinitionField dataDefinitionField =
dataDefinitionTable.FirstDataDefinitionField();

while (dataDefinitionField != null)
{
    ListBox.Items.Add(dataDefinitionField.Name);
    dataDefinitionField = dataDefinitionTable.NextDataDefinitionField;
}

```

7.19 VcDataRecord



A data record is the logical base of an object in a Gantt diagram, for example of a node, of a group node, of a link, of an operation or of a task. Objects have specific features, that are described in the fields of the record. For the fields of a data record, descriptions exist that are stored to data table fields. Data records and data table fields are collected in corresponding collection objects, which form a data table.

Properties

- AllData
- DataField
- DataTableName
- ID

Methods

- Delete
- IdentifyObject
- RelatedDataRecord
- Update

Properties

AllData

Property of VcDataRecord

This property lets you set or retrieve the complete data of a data record. When setting the property, a CSV string (using semicolons as separators) or the data type "object" are allowed, that contains all data fields of the record in an array. When retrieving the property, a string will be returned.

	Data Type	Explanation
Property value	System.Object	All data of the data record

Example Code VB.NET

```

Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection
Dim dataRecValue() As Object
Dim dataRecord As VcDataRecord

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata1")
dataRecCltn = dataTable.DataRecordCollection
ReDim dataRecValue(dataTable.DataTableFieldCollection.Count)
dataRecValue(0) = 1
dataRecValue(1) = "Node One"

'Object
dataRecord = dataRecCltn.Add(dataRecValue)
'CSV
dataRecord.AllData = "1;Node One;"

dataRecord.Update()

```

Example Code C#

```

VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
Object [] dataRecVal = new object[dataTable.DataTableFieldCollection.Count];

dataRecVal[0] = 1;
dataRecVal[1] = "Node One";

//Object
VcDataRecord dataRecord = dataRecordCltn.Add(dataRecVal);
//CSV
dataRecord.AllData = "1;Node One;";

dataRecord.Update();

```

DataField**Property of VcDataRecord**

This property lets you assign or retrieve data to/from a field of a data record. After the data field was modified by the **DataField** property, the graphical display in the diagram needs to be updated by the **UpdateDataRecord** method.

The property DataField is an Indexed Property, which in C# is addressed by the methods set_DataField (index, pvn) and get_DataField (index).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of data field

Property value	System.Object	Content of the data field
----------------	---------------	---------------------------

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcGantt1.DataTableCollection.FirstDataTable
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.DataRecordByID(1)

dataRecord.DataField(1) = "Node Two"
dataRecord.Update()
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(1);
dataRecord.set_DataField(1, "Node Two");
dataRecord.Update();
```

DataTableName**Read Only Property of VcDataRecord**

This property lets you retrieve the name of the data table that this data record belongs to.

	Data Type	Explanation
Property value	System.String	Name of the associated table

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcGantt1.DataTableCollection.FirstDataTable
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.DataRecordByID(1)

MsgBox(dataRecord.DataTableName)
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(1);

MessageBox.Show(dataRecord.DataTableName);
```

ID

Read Only Property of VcDataRecord

By this property you can retrieve the ID of a data record.

	Data Type	Explanation
Property value	System.String	Data record ID

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord
dataTable = VcGantt1.DataTableCollection.FirstDataTable
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.DataRecordByID(1)
MsgBox(dataRecord.ID)
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(1);
MessageBox.Show(dataRecord.ID);
```

Methods

Delete

Method of VcDataRecord

This method lets you delete a data record.

	Data Type	Explanation
Return value	System.Boolean	Data record was (true) / was not (false) deleted successfully

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcGantt1.DataTableCollection.FirstDataTable
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.DataRecordByID(1)

dataRecord.Delete()
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(1);

dataRecord.Delete();
```

IdentifyObject**Method of VcDataRecord**

This method lets you identify the object having been established via this VcDataRecord object.

The return value will be **true** if a data-based object could be identified, i.e. if a data-based object could be created for the graphic from the record.

	Data Type	Explanation
Parameter:		
⇒ establishedObject Param	System.Object	Identified object
establishedObjectTypeParam	VcObjectType	Object type
	Possible Values: .vcObjTypeBox 15 .vcObjTypeCalendarGrid 18 .vcObjTypeCurve 12 .vcObjTypeDateLine 9 .vcObjTypeGroup 7 .vcObjTypeGroupInDiagram 11 .vcObjTypeGroupInTable 7 .vcObjTypeHistogram 13 .vcObjTypeLayer 8 .vcObjTypeLinkCollection 3 .vcObjTypeNodeInDiagram 2 .vcObjTypeNodeInLegend 17 .vcObjTypeNodeInTable 1 .vcObjTypeNone 0 .vcObjTypeNumericScale 10 .vcObjTypeSummaryNode 14 .vcObjTypeTable 4 .vcObjTypeTableCaption 5 .vcObjTypeTimeScale 6	object type box object type calendar grid object type curve object type date line object type group object type group in diagram area object type group in table area object type histogram object type layer object type link collection object type node in diagram area object type node in legend area object type node in table area no object object type numeric scale object type summary bar object type table object type table caption object type time scale
Return value	System.Boolean	data-based object has been/has not been established

RelatedDataRecord

Method of VcDataRecord

This property lets you relate a data record to a different one or retrieve a related data set. When using extended data tables, the data records of a table can be related to the data records of another table by primary keys.

	Data Type	Explanation
Parameter: ⇨ index	System.Int16	Index of data field
Return value	VcDataRecord	Related data record

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles VcGantt1.VcNodeLeftClicking
    Dim dataTable As VcDataTable
    Dim dataRecordCltn As VcDataRecordCollection
    Dim firstDataRecord As VcDataRecord
    Dim secondDataRecord As VcDataRecord

    dataTable = VcGantt1.DataTableCollection.DataTableByIndex(0)
    dataRecordCltn = dataTable.DataRecordCollection

    firstDataRecord = dataRecordCltn.DataRecordByID(e.Node.DataField(0))
    secondDataRecord = firstDataRecord.RelatedDataRecord(2)

    MsgBox(secondDataRecord.AllData)
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeLeftClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    VcDataTable dataTable = vcGantt1.DataTableCollection.DataTableByIndex(0);
    VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
    VcDataRecord firstDataRecord =
dataRecordCltn.DataRecordByID(e.Node.get_DataField(0));
    VcDataRecord secondDataRecord = firstDataRecord.RelatedDataRecord(2);

    MessageBox.Show(secondDataRecord.AllData.ToString());
}
```

Update

Method of VcDataRecord

If data fields of a data record were modified by the **DataField** property, the diagram needs to be updated by the **UpdateDataRecord** method.

612 API Reference: VcDataRecord

	Data Type	Explanation
Return value	System.Boolean	Data record was (true) / was not (false) updated successfully

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcGantt1.DataTableCollection.FirstDataTable
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.DataRecordByID(1)

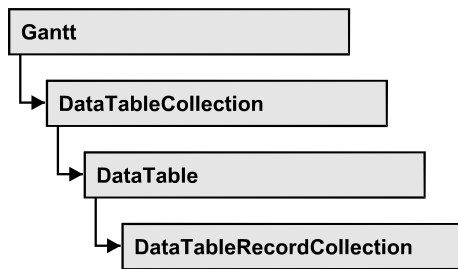
dataRecord.DataField(1) = "Node Two"
dataRecord.Update()
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(1);

dataRecord.set_DataField(1, "Node Two");
dataRecord.Update();
```

7.20 VcDataRecordCollection



An object of the type `VcDataRecordCollection` contains the data records of a table. The property **Count** retrieves the number of records present in the collection; the `Enumerator` object and the methods **FirstDataRecord** and **NextDataRecord** allow to access data records by iteration while by **DataRecordByID** single data records can be accessed. **Add** and **Remove** are basic administering methods, and **Update** lets you refresh the graphical display of objects by data of the records recently modified.

Properties

- Count

Methods

- Add
- DataRecordByID
- FirstDataRecord
- GetEnumerator
- GetNewUniqueID
- NextDataRecord
- Remove
- Update

Properties

Count

Read Only Property of VcDataRecordCollection

This property lets you retrieve the number of data records in the `DataRecordCollection` object.

	Data Type	Explanation
Property value	System.Int32	Number of data records in the collection object

Example Code VB.NET

```

Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection
MsgBox("Number of DataRecords: " & dataRecordCltn.Count)

```

Example Code C#

```

VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
MessageBox.Show("Number of DataRecords: " + dataRecordCltn.Count);

```

Methods

Add

Method of VcDataRecordCollection

By this method you can create a data record as a member of the DataRecordCollection. If the ID was not used before, the new data record will be returned; otherwise a **VcPrimaryKeyNotUniqueException** will be thrown. After adding the data record, the method **VcGantt.EndLoading** needs to be invoked to make the modification take effect.

	Data Type	Explanation
Parameter: ⇒ dataRecordContent	VcObject	Content of the data record (as an array or a string)
Return value	VcDataRecord	Data record created

Example Code VB.NET

```

Const Main_ID = 0
Const Main_Name = 1
Const Main_Start = 2
Const Main_Duration = 4
'...

Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection
Dim dataRec1 As VcDataRecord
Dim dataRecVal() As Object

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection

Dim dataRec1 As VcDataRecord
ReDim dataRecVal(dataTable.DataTableFieldCollection.Count)

dataRecVal(Main_ID) = 1
dataRecVal(Main_Name) = "Node 1"
dataRecVal(Main_Start) = DateSerial(2014, 1, 8)
dataRecVal(Main_Duration) = 8
dataRec1 = dataRecCltn.Add(dataRecVal)
VcGantt1.EndLoading()

' equivalent
' dataRec1 = dataRecCltn.Add("1;Node 1;01.08.14;;8")

```

Example Code C#

```

const int Main_ID = 0;
const int Main_Name = 1;
const int Main_Start = 2;
const int Main_Duration = 4;

//...

VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");

VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

Object [] dataRecVal = new object[dataTable.DataTableFieldCollection.Count];

VcDataRecord dataRec1;

dataRecVal[Main_ID] = "1";
dataRecVal[Main_Name] = "Node 1";
dataRecVal[Main_Start] = "08.01.2014";
dataRecVal[Main_Duration] = 8;
dataRec1 = dataRecCltn.Add(dataRecVal);
VcGantt1.EndLoading();

// equivalent
// dataRec2 = dataRecCltn.Add("1;Node 1;01.08.14;;8")

```

DataRecordByID

Method of VcDataRecordCollection

This method lets you access a data record by its identification. If a data record of the specified ID does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

If the identification consists of several fields (composite primary key), this multipart ID has to be specified as follows:

ID=ID1|ID2|ID3

	Data Type	Explanation
Parameter: ⇒ dataRecordID	System.String	ID of the data record
Return value	VcDataRecord	Data record object

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.DataRecordByID(0)
```

Example Code C#

```
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(0);
```

FirstDataRecord

Method of VcDataRecordCollection

This method can be used to access the initial value, i.e. the first data record of a data record collection, and to continue in a forward iteration loop by the method **NextDataRecord** for the data records following. If there is no data record in the data record collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcDataRecord	First data record

Example Code VB.NET

```

Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection
dataRecord = dataRecordCltn.FirstDataRecord

```

Example Code C#

```

VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;
VcDataRecord dataRecord = dataRecordCltn.FirstDataRecord();

```

GetEnumerator**Method of VcDataRecordCollection**

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the data records included.

	Data Type	Explanation
Return value	VcObject	Enumerator object

Example Code VB.NET

```

Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection

VcGantt1.SuspendUpdate(True)

dataRecord = dataRecordCltn.FirstDataRecord
While Not dataRecord Is Nothing
    dataRecord.DataField(4) = "10"
    dataRecord.Update()
    dataRecord = dataRecordCltn.NextDataRecord
End While

VcGantt1.SuspendUpdate(False)

```

Example Code C#

```

VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;

vcGantt1.SuspendUpdate(true);

foreach (VcDataRecord dataRecord in dataRecordCltn)
{
    dataRecord.set_DataField(4, "10");
    dataRecord.Update();
    dataRecordCltn.NextDataRecord();
}

vcGantt1.SuspendUpdate(false);

```

GetNewUniqueID**Method of VcDataRecordCollection**

By this method you can have a unique ID generated for a data record. This method is useful if you wish to add a data record for example by the method **Add** but do not wish to create the ID manually.

	Data Type	Explanation
Return value	System.Int32	New data record ID

NextDataRecord**Method of VcDataRecordCollection**

This method can be used in a forward iteration loop to retrieve subsequent data records from a data record collection after initializing the loop by the method **FirstDataRecord**. If there is no data record left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcDataRecord	Succeeding data record

Example Code VB.NET

```

Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection

VcGantt1.SuspendUpdate(True)

dataRecord = dataRecordCltn.FirstDataRecord
While Not dataRecord Is Nothing
    dataRecord.DataField(4) = "10"
    dataRecord.Update()
    dataRecord = dataRecordCltn.NextDataRecord
End While

VcGantt1.SuspendUpdate(False)

```

Example Code C#

```

VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecordCltn = dataTable.DataRecordCollection;

vcGantt1.SuspendUpdate(true);

foreach (VcDataRecord dataRecord in dataRecordCltn)
{
    dataRecord.set_DataField(4, "10");
    dataRecord.Update();
    dataRecordCltn.NextDataRecord();
}

vcGantt1.SuspendUpdate(false);

```

Remove**Method of VcDataRecordCollection**

This method lets you delete a data record. The method returns **true** after having deleted a data record and **false** when no data record was deleted. The content of the data record is used to identify the object by its identification.

	Data Type	Explanation
Parameter:		
⇒ dataRecordContent	VcObject	Content of the data record (as an array or a string)
Return value	System.Boolean	true

Example Code VB.NET

```

Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection
dataRecordCltn.Remove("1;1Activity; Y;Z;18.01.14;;5")
VcGantt1.EndLoading()

' equivalent
' dataRecord = dataRecordCltn.DataRecordByID(1)
' dataRecord.Delete()
' dataRecord.Update()

```

Example Code C#

```

VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

dataRecCltn.Remove("1;1Activity Y;Z;18.01.14;;5");
VcGantt1.EndLoading();

// equivalent
// VcDataRecord dataRecord = dataRecordCltn.DataRecordByID(1);
// dataRecord.Delete();
// dataRecord.Update();

```

Update

Method of VcDataRecordCollection

This method updates a data record in the the data record collection if it previously was created by the **Add()** method. If the data record to be updated does not exist, it will then be created by the **Update** method. Also see **VcDataRecordCollection.Add()**. After updating the data record, the method **VcGantt.EndLoading** needs to be invoked to make the modification take effect.

	Data Type	Explanation
Parameter: ⇒ dataRecordContent	VcObject	Content of the data record (as an array or a string)
Return value	System.Boolean	Update successful (true) / not successful (false)

Example Code VB.NET

```

Dim dataTable As VcDataTable
Dim dataRecordCltn As VcDataRecordCollection
Dim dataRecord As VcDataRecord

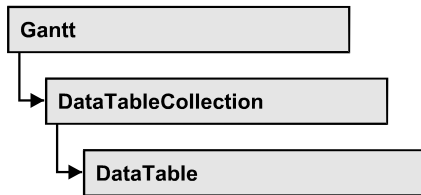
dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecordCltn = dataTable.DataRecordCollection
dataRecordCltn.Update("1;1.8.2017;;8")
VcGantt1.EndLoading()

```

Example Code C#

```
VcDataTable dataTable =  
vcGantt1.DataTableCollection.DataTableByName("Maindata");  
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;  
dataRecCltn.Update("1;1.8.2017;;8");  
VcGantt1.EndLoading();
```

7.21 VcDataTable



A data table comprises **data records**, including their data fields and their contents, and it comprises the descriptions of the record fields, which are called **data table fields**. Data records and data table fields can be processed and iterated over by collection objects.

Data tables on their hand can be processed by a collection object of their own.

Properties

- DataRecordCollection
- DataTableFieldCollection
- Description
- MultiplePrimaryKeysAllowed
- Name

Properties

DataRecordCollection

Read Only Property of VcDataTable

This property returns the DataRecordCollection object of the data table. The collection contains all existing data records of a table. It is empty on the start of the program.

	Data Type	Explanation
Property value	VcDataRecordCollection	DataRecordCollection object

Example Code VB.NET

```
Dim dataTable As VcDataTable

dataTable = VcGantt1.DataTableCollection.FirstDataTable()
MsgBox(dataTable.DataRecordCollection.Count)
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
MessageBox.Show(dataTable.DataRecordCollection.Count.ToString());
```

DataTableFieldCollection**Read Only Property of VcDataTable**

This property returns the **DataTableFieldCollection** object of the data table. The collection contains the definitions of the fields of a data record of the table. On the start of the program, it holds the data fields that were defined at design time. More data fields can be added at run time by the method **Add** of the object **DataTableFieldCollection**. The definition of data table fields needs to have been terminated before data records can be filled in the table.

	Data Type	Explanation
Property value	VcDataTableFieldCollection	DataTableFieldCollection object

Example Code VB.NET

```
Dim dataTable As VcDataTable

dataTable = VcGantt1.DataTableCollection.DataTableByIndex(0)
MsgBox(dataTable.DataTableFieldCollection.Count)
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.DataTableByIndex(0);
MessageBox.Show(dataTable.DataTableFieldCollection.Count.ToString());
```

Description**Property of VcDataTable**

This property lets you set or retrieve the description of the data table. Names of objects, for example of the table, that contain some information on the object, often are long and cannot be displayed fully in previews; so their benefit is limited. To use the opportunity of short names without having to abandon the information of a long name, you can store additional information to this field. Its contents will be displayed in the data table dialog.

	Data Type	Explanation
Property value	System.String	Description of the data table Default value: Empty string

Example Code VB.NET

```
Dim dataTable As VcDataTable

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataTable.Description = "This table contains data for nodes"
```

Example Code C#

```
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
dataTable.Description = "This table contains data for nodes";
```

MultiplePrimaryKeysAllowed**Property of VcDataTable**

With this property you can set or retrieve whether the use of composite primary keys is possible.

	Data Type	Explanation
Property value	System.Boolean	Use of composite primary keys allowed (true)/not allowed (false) Default value: False

Name**Property of VcDataTable**

This property lets you set or retrieve the name of the data table. The name of a data table has to set by obligation; beside, it has to be unique. An empty character string is not allowed. Upper and lower case characters are accepted as different. By the method **DataTableByName** of the object **DataTableCollection** you can retrieve a reference to the data table object.

	Data Type	Explanation
Property value	System.String	Name of the data table Default value: Empty string

Example Code VB.NET

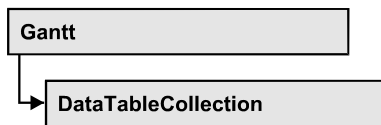
```
Dim dataTable As VcDataTable

dataTable = VcGantt1.DataTableCollection.DataTableByIndex(0)
MessageBox(dataTable.Name)
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.DataTableByIndex(0);
MessageBox.Show(dataTable.Name);
```

7.22 VcDataTableCollection



An object of the type `VcDataTableCollection` holds a collection of tables. The property **Count** retrieves the number of tables present in the collection; the Enumerator object and the methods **FirstDataTable** and **NextDataTable** allow to access tables by iteration while by **DataTableByName** and **DataTableByindex** single tables can be accessed. **Add** and **Copy** are basic administrating methods, and **Update** makes the recent modifications of the data structures known to the XGantt object.

Properties

- Count

Methods

- Add
- Copy
- DataTableByIndex
- DataTableByName
- FirstDataTable
- GetEnumerator
- NextDataTable
- Update

Properties

Count

Read Only Property of VcDataTableCollection

This property lets you retrieve the number of data tables in the `DataTableCollection` object.

	Data Type	Explanation
Property value	System.Int32	Number of data tables in the collection object

Example Code VB.NET

```
Dim dataTableCltn As VcDataTableCollection

dataTableCltn = VcGantt1.DataTableCollection
MsgBox(dataTableCltn.Count.ToString())
```

Example Code C#

```
VcDataTableCollection dataTableCltn = vcGantt1.DataTableCollection;
MessageBox.Show(dataTableCltn.Count.ToString());
```

Methods

Add

Method of VcDataTableCollection

By this method you can create a data table as a member of the **DataTableCollection**. If the name was not used before, an object of the type **VcDataTable** will be returned; otherwise "Nothing" (in Visual Basic) or "0" (in other languages) will be returned. Only if the property **ExtendedDataTables** is set to **True**, tables can be added. 90 data tables can be created at maximum.

	Data Type	Explanation
Parameter:		
⇒ dataTableName	System.String	Name of the new data table
Return value	VcDataTable	Data table generated

Example Code VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcGantt1.DataTableCollection
dataTable = dataTableCltn.Add("Resources")
dataTableCltn.Update()
```

Example Code C#

```
VcDataTableCollection dataTableCltn = vcGantt1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.Add("Resources");
dataTableCltn.Update();
```

Copy

Method of VcDataTableCollection

This method lets you copy a data table. Probably existing data records are not copied, just the definition fields. Only if the property **ExtendedDataTables**

was set to **true**, data tables can be copied. If the data table could be copied, a new object of the type **VcDataTable** will be returned; otherwise **Nothing** in Visual Basic or **0** in other languages. The table names are case sensitive.

	Data Type	Explanation
Parameter:		
⇒ dataTableName	System.String	Name of the data table to be copied (source table)
⇒ newDataTableName	System.String	Name of the data table to be generated (target table)
Return value	VcDataTable	Data table object generated

Example Code VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcGantt1.DataTableCollection
dataTable = dataTableCltn.Copy("Resources", "NewResources")
dataTableCltn.Update()
```

Example Code C#

```
VcDataTableCollection dataTableCltn = vcGantt1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.Copy("Resources", "NewResources");
dataTableCltn.Update();
```

DataTableByIndex

Method of VcDataTableCollection

This method lets you access a data table by its index. The index of the first table is 0. If a data table of the specified index does not exist, a **none** object will be returned (**Nothing** in Visual Basic or **0** in other languages).

	Data Type	Explanation
Parameter:		
⇒ index	System.Int16	Index of the data table
Return value	VcDataTable	Data table object returned

Example Code VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcGantt1.DataTableCollection
dataTable = dataTableCltn.DataTableByIndex(2)
MsgBox(dataTable.Name)
```


Example Code C#

```
VcDataTableCollection dataTableCltn = vcGantt1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.DataTableByIndex(2);
MessageBox.Show(dataTable.Name);
```

DataTableByName**Method of VcDataTableCollection**

This method lets you access a data table by its name. If a data table of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic or **0** in other languages).

	Data Type	Explanation
Parameter:		
⇒ dataTableName	System.String	Name of the data table
Return value	VcDataTable	Data table object returned

Example Code VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = vcGantt1.DataTableCollection
dataTable = dataTableCltn.DataTableByName("Resources")
MsgBox(dataTable.Description)
```

Example Code C#

```
VcDataTableCollection dataTableCltn = vcGantt1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.DataTableByName("Resources");
MessageBox.Show(dataTable.Description);
```

FirstDataTable**Method of VcDataTableCollection**

This method can be used to access the initial value, i.e. the first data table of a data table collection, and to continue in a forward iteration loop by the method **NextDataTable** for the data tables following. If there is no data table in the data table collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcDataTable	First data table

Example Code VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcGantt1.DataTableCollection
dataTable = dataTableCltn.FirstDataTable
```

Example Code C#

```
VcDataTableCollection dataTableCltn = vcGantt1.DataTableCollection;
VcDataTable dataTable= dataTableCltn.FirstDataTable();
```

GetEnumerator

Method of VcDataTableCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the data tables included.

	Data Type	Explanation
Return value	VcObject	Enumerator object

Example Code VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcGantt1.DataTableCollection
For Each dataTable In dataTableCltn
    ListBox1.Items.Add(dataTable.Name)
Next
```

Example Code C#

```
VcDataTableCollection dataTableCltn = vcGantt1.DataTableCollection;
foreach (VcDataTable dataTable in dataTableCltn)
    listBox1.Items.Add(dataTable.Name);
```

NextDataTable

Method of VcDataTableCollection

This method can be used in a forward iteration loop to retrieve subsequent data tables from a data table collection after initializing the loop by the method **FirstDataTable**. If there is no data table left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcDataTable	Succeeding data table

Example Code VB.NET

```

Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable
Dim i As Integer

dataTableCltn = VcGantt1.DataTableCollection
dataTable = dataTableCltn.FirstDataTable
For i = 1 To dataTableCltn.Count
    ListBox1.Items.Add(dataTable.Name)
    dataTable = dataTableCltn.NextDataTable
Next

```

Example Code C#

```

VcDataTableCollection dataTableCltn = vcGantt1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.FirstDataTable();
for (int i=0; i<dataTableCltn.Count; i++)
{
    listBox1.Items.Add(dataTable.Name);
    dataTable = dataTableCltn.NextDataTable();
}

```

Update

Method of VcDataTableCollection

This method lets you update recent modifications of the data structures. It makes the modifications on data table definitions and on data table fields become operative in the VARCHART component and avoids individual updates after several modifications.

	Data Type	Explanation
Return value	System.Boolean	Update successful (true) / not successful (false)

Example Code VB.NET

```

Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcGantt1.DataTableCollection
dataTable = dataTableCltn.Add("Resources")
dataTable.DataTableFieldCollection.Add("Id")
dataTableCltn.Update()

```

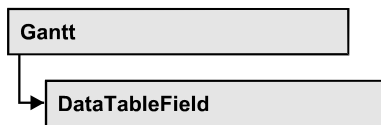
Example Code C#

```

VcDataTableCollection dataTableCltn = vcGantt1.DataTableCollection;
VcDataTable dataTable = dataTableCltn.Add("Resources");
dataTable.DataTableFieldCollection.Add("Id");
dataTableCltn.Update();

```

7.23 VcDataTableField



An object of the type **VcDataTableField** defines the properties of a data field in a data record. Part of the definition of a data table field are its name, its data type and whether it represents the primary key, by which a data record can be uniquely identified. For example, by referring to the primary key, other data tables can relate to a data table. To create a relation, a table needs to specify the primary key of a different table by the property **RelationshipFieldIndex**.

The DataTableField objects of a data table are administered by the object **DataTableFieldCollection**.

Properties

- DataTableName
- DateFormat
- Editable
- Hidden
- Index
- Name
- PrimaryKey
- RelationshipFieldIndex
- Type

Properties

DataTableName

Read Only Property of VcDataTableField

This property lets you retrieve the name of the associated data table.

	Data Type	Explanation
Property value	System.String	Name of the data table

Example Code VB.NET

```
Dim dataTable As VcDataTable

dataTable = VcGantt1.DataTableCollection.FirstDataTable()
MsgBox(dataTable.DataTableFieldCollection.FirstDataTableField.DataTableName)
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
MessageBox.Show(dataTable.DataTableFieldCollection.FirstDataTableField().DataTableName);
```

DateFormat**Read Only Property of VcDataTableField**

This property lets you set or retrieve the date format of the record field that is specified by the property **RelationshipFieldIndex**. The date format is used when reading or storing CSV files and when the format type **String** is used when adding a data record by the method **Add**. This property only works if the data type of the field was set to **vcDataTableFieldDateTime**.

Note:Remember to set the property **Type** before setting the property **DateFormat**.

	Data Type	Explanation
Property value	System.String	Date format {DMYhms;:./}

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcGantt1.DataTableCollection.DataTableByName("Operation")
dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start")
dataTableField.Type = VcDataTableFieldType.vcDataTableFieldDateTimeType
'DateFormat = "01.12.2014"
dataTableField.DateFormat = "DD.MM.YYYY"
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Operation");
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start");
dataTableField.Type = VcDataTableFieldType.vcDataTableFieldDateTimeType;
//DateFormat = "01.12.2014"
dataTableField.DateFormat = "DD.MM.YYYY";
vcGantt1.DataTableCollection.Update();
```

Editable

Property of VcDataTableField

This property lets you set or retrieve whether the record field should be editable at run time in the chart table and in the dialog **EditNode**.

	Data Type	Explanation
Property value	System.Boolean	Field editable (True) / not editable (False) Default value: True

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcGantt1.DataTableCollection.DataTableByName("Operation")
dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start")
dataTableField.Editable = False
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataTable dataTable =
VcGantt1.DataTableCollection.DataTableByName("Operation");
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start");
dataTableField.Editable = false;
VcGantt1.DataTableCollection.Update();
```

Hidden

Property of VcDataTableField

This property lets you set or retrieve whether the data field should be hidden at run time in the dialogs **EditNode** and **EditLink**.

	Data Type	Explanation
Property value	System.Boolean	Field hidden (True) / not hidden (False) Default value: False

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcGantt1.DataTableCollection.DataTableByName("Operation")
dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start")
dataTableField.Hidden = True
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Operation");
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start");
dataTableField.Hidden = true;
vcGantt1.DataTableCollection.Update();
```

Index**Read Only Property of VcDataTableField**

This property lets you retrieve the index of the data table field in the associated data table.

	Data Type	Explanation
Property value	System.Int16	Index of the data table field

Name**Property of VcDataTableField**

This property lets you set or retrieve the name of the record field. The name is indicated in runtime dialogs such as the **EditNode** dialog. Accessing a field by the API although requires its index that the field has within the **Data-TableFieldCollection** object.

	Data Type	Explanation
Property value	System.String	Name of the field Default value: Empty string

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = vcGantt1.DataTableCollection.DataTableByName("Operation")
dataTableField = dataTable.DataTableFieldCollection.Add("Start")
vcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Operation");
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.Add("Start");
vcGantt1.DataTableCollection.Update();
```

PrimaryKey

Property of VcDataTableField

This property lets you set or retrieve whether this field contains the primary key, which is used for the unique identification of a data record. In a data table, only one of the fields that were defined can be the primary key. Within the same table, assigning the primary key function to a field automatically cancels the previous assignment. A primary key is required in a table if records of a different table are to depend on the records of the former one.

	Data Type	Explanation
Property value	System.Boolean	The field serves (True) / does not serve (False) as a primary key. Default value: False

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField
Dim isPrimaryKey As Boolean

dataTable = VcGantt1.DataTableCollection.DataTableByName("Operation")
dataTableField = dataTable.DataTableFieldCollection.DataTableFieldByName("Id")
dataTableField.PrimaryKey = True
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Operation");
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Id");
dataTableField.PrimaryKey = true;
vcGantt1.DataTableCollection.Update();
```

RelationshipFieldIndex

Property of VcDataTableField

This property lets you combine a data field and its data description. For this, please set the index of the data record field to which the settings of this data table field shall refer.

	Data Type	Explanation
Property value	System.Int32	Index of the record field to which the data definition of the data table field refers. Default value: -1

Example Code VB.NET

```

Dim dataTableTask As VcDataTable
Dim dataTaskFieldId As VcDataTableField
Dim dataTaskFieldName As VcDataTableField
Dim dataTableOperation As VcDataTable
Dim dataOperationFieldId As VcDataTableField
Dim dataOperationFieldName As VcDataTableField
Dim dataOperationFieldTaskId As VcDataTableField

'Create table Task
dataTableTask = VcGantt1.DataTableCollection.Add("Task")
dataTaskFieldId = dataTableTask.DataTableFieldCollection.Add("Id")
dataTaskFieldId.PrimaryKey = True
dataTaskFieldName = dataTableTask.DataTableFieldCollection.Add("Name")
dataTaskFieldName.Type = VcDataTableFieldType.vcDataTableFieldStringType

'Create table Operation
dataTableOperation = VcGantt1.DataTableCollection.Add("Operation")
dataOperationFieldId = dataTableOperation.DataTableFieldCollection.Add("Id")
dataOperationFieldId.PrimaryKey = True
dataOperationFieldName = dataTableOperation.DataTableFieldCollection.Add("Name")
dataOperationFieldName.Type = VcDataTableFieldType.vcDataTableFieldStringType
dataOperationFieldTaskId =
dataTableOperation.DataTableFieldCollection.Add("TaskId")
dataOperationFieldTaskId.Type = VcDataTableFieldType.vcDataTableFieldIntegerType

'Node tables Task and Operations
dataOperationFieldTaskId.RelationshipFieldIndex =
VcGantt1.DetectFieldIndex("Task", "Id")
VcGantt1.DataTableCollection.Update()

```

Example Code C#

```

//Create table Task
VcDataTable dataTableTask = vcGantt1.DataTableCollection.Add("Task");
VcDataTableField dataTaskFieldId =
dataTableTask.DataTableFieldCollection.Add("Id");
dataTaskFieldId.PrimaryKey = true;
VcDataTableField dataTaskFieldName =
dataTableTask.DataTableFieldCollection.Add("Name");
dataTaskFieldName.Type = VcDataDefinitionFieldType.vcDefFieldStringType;

//Create table Operation
VcDataTable dataTableOperation = vcGantt1.DataTableCollection.Add("Operation");
VcDataTableField dataOperationFieldId =
dataTableOperation.DataTableFieldCollection.Add("Id");
dataOperationFieldId.PrimaryKey = true;
VcDataTableField dataOperationFieldName =
dataTableOperation.DataTableFieldCollection.Add("Name");
dataOperationFieldName.Type = VcDataDefinitionFieldType.vcDefFieldStringType;
VcDataTableField dataOperationFieldTaskId =
dataTableOperation.DataTableFieldCollection.Add("TaskId");
dataOperationFieldTaskId.Type = VcDataDefinitionFieldType.vcDefFieldIntegerType;

//Node tables Task and Operation
dataOperationFieldTaskId.RelationshipFieldIndex =
vcGantt1.DetectFieldIndex("Task","Id");
vcGantt1.DataTableCollection.Update();

```

Type**Property of VcDataTableField**

This property lets you set or retrieve the data type of the field.

Note: Setting the property **Type** may change the property **DateFormat**. By setting this property to **vcDataTableAlphanumeric** or to **vcDataTableFieldInteger** the date format probably set will change to "".

	Data Type	Explanation
Property value	VcDataTableFieldType	Data type of the field, can contain 512 characters maximum Default value: vcDataTableFieldIntegerType
	Possible Values: .vcDataFieldAlphanumericType 1 .vcDataFieldDateTimeType 3 .vcDataTableFieldIntegerType 2	Data type alphanumeric Data type date Data type integer (32 bits)

Example Code VB.NET

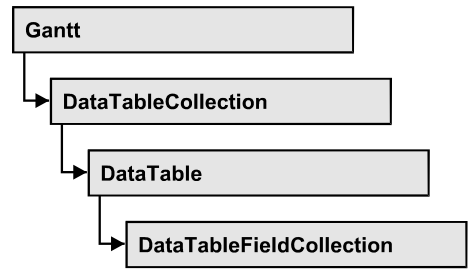
```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

VcGantt1.DataTableCollection.DataTableByName("Operation")
dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start")
dataTableField.Type = VcDataTableFieldType.vcDataTableFieldDateTimeType
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Operation");
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Start");
dataTableField.Type = VcDataTableFieldType.vcDataTableFieldDateTimeType;
vcGantt1.DataTableCollection.Update();
```

7.24 VcDataTableFieldCollection



An object of the type VcDataTableFieldCollection automatically contains all data fields of a data table. The property **Count** retrieves the number of fields present in the collection; the Enumerator object and the methods **FirstDataTableField** and **NextDataTableField** allow to access data fields by iteration while by **DataTableFieldByName** and **DataTableFieldByIndex** single data fields can be accessed. **Add** and **Copy** represent basic administering methods.

Properties

- Count

Methods

- Add
- Copy
- DataTableFieldByIndex
- DataTableFieldByName
- FirstDataTableField
- GetEnumerator
- NextDataTableField

Properties

Count

Read Only Property of VcDataTableFieldCollection

This property lets you retrieve the number of data table fields in the DataTableFieldCollection object.

	Data Type	Explanation
Property value	System.Int32	Number of data table fields in the collection object

Example Code VB.NET

```
Dim dataTable As VcDataTable

dataTable = VcGantt1.DataTableCollection.FirstDataTable()
MsgBox(dataTable.DataTableFieldCollection.Count.ToString())
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
MessageBox.Show(dataTable.DataTableFieldCollection.Count.ToString());
```

Methods

Add

Method of VcDataTableFieldCollection

By this method you can create a data table field as a member of the DataTableFieldCollection. If the name was not used before, the new data field will be returned; otherwise "Nothing" (Visual Basic) or "0" (other languages) will be returned. 9,999 fields can be created at maximum.

	Data Type	Explanation
Parameter: ⇒ dataTableFieldName	System.String	Name of the data table field to be generated
Return value	VcDataTableField	Data table field generated

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcGantt1.DataTableCollection.FirstDataTable()
dataTableField = dataTable.DataTableFieldCollection.Add("Priority")
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.Add("Priority");
vcGantt1.DataTableCollection.Update();
```

Copy

Method of VcDataTableFieldCollection

This method lets you copy a data table field. The field is identified by its name.

	Data Type	Explanation
Parameter:		
⇒ dataTableFieldName	System.String	Name of the data table field to be copied (source field)
⇒ newDataTableFieldName	System.String	Name of the data table field to be generated (target field)
Return value	VcDataTableField	Data table field generated

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcGantt1.DataTableCollection.FirstDataTable()
dataTableField = dataTable.DataTableFieldCollection.Copy("Name", "NewName")
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.Copy("Name", "NewName");
vcGantt1.DataTableCollection.Update();
```

DataTableFieldByIndex**Method of VcDataTableFieldCollection**

This method lets you access a data table field by its index. If a data field does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter:		
⇒ index	System.Int16	Index of the data table field
Return value	VcDataTableField	Data table field returned

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcGantt1.DataTableCollection.FirstDataTable()
dataTableField = dataTable.DataTableFieldCollection.DataTableFieldByIndex(1)
MsgBox(dataTableField.Name)
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByIndex(1);
MessageBox.Show(dataTableField.Name);
```

DataTableFieldByName

Method of VcDataTableFieldCollection

This method lets you access a data table field by its name. If a field of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ dataTableFieldName	System.String	Name of the data table field
Return value	VcDataTableField	Data table field returned

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcGantt1.DataTableCollection.FirstDataTable()
dataTableField = dataTable.DataTableFieldCollection.DataTableFieldByName("Name")
dataTableField.Editable = False
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.DataTableFieldByName("Name");
dataTableField.Editable = false;
vcGantt1.DataTableCollection.Update();
```

FirstDataTableField

Method of VcDataTableFieldCollection

This method can be used to access the initial value, i.e. the first data table field of a data table field collection, and to continue in a forward iteration loop by the method **NextDataTableField** for the fields following. If there is no field in the data table field collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcDataTableField	First data table field

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcGantt1.DataTableCollection.FirstDataTable()
dataTableField = dataTable.DataTableFieldCollection.FirstDataTableField()
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
VcDataTableField dataTableField =
dataTable.DataTableFieldCollection.FirstDataTableField();
```

GetEnumerator**Method of VcDataTableFieldCollection**

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the data table fields included.

	Data Type	Explanation
Return value	VcObject	Enumerator object

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataTableField As VcDataTableField

dataTable = VcGantt1.DataTableCollection.FirstDataTable()
For Each dataTableField In dataTable.DataTableFieldCollection
    ListBox1.Items.Add(dataTableField.Name)
Next
```

Example Code C#

```
VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
foreach (VcDataTableField dataTableField in dataTable.DataTableFieldCollection)
    listBox1.Items.Add(dataTableField.Name);
```

NextDataTableField**Method of VcDataTableFieldCollection**

This method can be used in a forward iteration loop to retrieve subsequent data table fields from a data table field collection after initializing the loop by the method **FirstDataTableField**. If there is no field left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcDataTable	Succeeding data table field

Example Code VB.NET

```

Dim dataTable As VcDataTable
Dim dataTableFieldCltn As VcDataTableFieldCollection
Dim dataTableField As VcDataTableField
Dim i As Integer

dataTable = VcGantt1.DataTableCollection.FirstDataTable()
dataTableFieldCltn = dataTable.DataTableFieldCollection
dataTableField = dataTableFieldCltn.FirstDataTableField
For i = 1 To dataTableFieldCltn.Count
    ListBox1.Items.Add(dataTableField.Name)
    dataTableField = dataTableFieldCltn.NextDataTableField()
Next

```

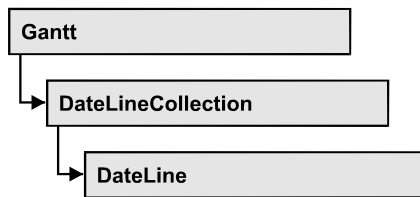
Example Code C#

```

VcDataTable dataTable = vcGantt1.DataTableCollection.FirstDataTable();
VcDataTableFieldCollection dataTableFieldCltn =
dataTable.DataTableFieldCollection;
VcDataTableField dataTableField = dataTableFieldCltn.FirstDataTableField();
for (int i=0; i<dataTableFieldCltn.Count; i++)
{
    listBox1.Items.Add(dataTableField.Name);
    dataTableField = dataTableFieldCltn.NextDataTableField();
}

```


7.25 VcDateLine



An object of the type VcDateLine is a time-orientated vertical line in a Gantt diagram that marks a date.

Properties

- AlwaysCurrentDate
- Date
- DateDataFieldIndex
- Font
- FontColor
- Identifiable
- LabelPosition
- LineColor
- LineThickness
- LineType
- Movable
- Name
- Priority
- SnapTarget
- Specification
- Text
- TurningAnnotationEnabled
- UpdateBehaviorName
- Visible
- VisibleDataFieldIndex
- VisibleMapName

Methods

- PutInOrderAfter

Properties

AlwaysCurrentDate

Read Only Property of VcDateLine

This property lets you set or retrieve whether a date line always displays the current date and time at the time of the start of VARCHART control. This property can be set in the **Specify Date Lines** dialog.

	Data Type	Explanation
Property value	System.Boolean	Property active/not active Default value: False

Example Code VB.NET

```
Dim dateLine As VcDateLine
Dim dateLineTimer As Timer

dateLine = VcGantt1.DateLineCollection.DateLineByName("DateLine1")
If dateLine.AlwaysAtCurrentDate = True Then
    dateLineTimer.Enabled = True
End If
```

Example Code C#

```
VcDateLine dateLine = vcGantt1.DateLineCollection.DateLineByName("DateLine1");
Timer dateLineTimer;

if (dateLine.AlwaysAtCurrentDate)
    dateLineTimer.Enabled = true;
```

Date

Property of VcDateLine

This property lets you set or retrieve the position of a date line. Please note: date and time must be separated by a blank. This property also can be set in the **Specify Date Lines** dialog.

	Data Type	Explanation
Property value	System.DateTime	Date {1.1.1970...31.12.2035} Default value: none or current date

Example Code VB.NET

```
Dim dateLine As VcDateLine

dateLine = VcGantt1.DateLineCollection.DateLineByName("DateLine1")
dateLine.Date = "30.09.14 12:00:00"
```

Example Code C#

```
VcDateLine dateLine = vcGantt1.DateLineCollection.DateLineByName("DateLine1");
dateLine.Date = Convert.ToDateTime("30.09.14 12:00:00");
```

DateDataFieldIndex**Property of VcDateLine**

This property lets you set or retrieve the index of the data field containing the date of the individual date line.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which contains the date

Font**Property of VcDateLine**

This property lets you set or retrieve all font attributes of the date line and can also be set in the **Edit Date Line** dialog.

	Data Type	Explanation
Property value	System.Drawing.Font	Font attributes of the date line texts

FontColor**Property of VcDateLine**

This property lets you set or retrieve the font color of the date line and can also be set in the **Edit Date Line** dialog.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values

Identifiable

Read Only Property of VcDateLine

This property lets you set or retrieve whether or not a date line grid can be identified. If this property was set to **True**, the date line can be identified by the VcGantt method **IdentifyObjectAt**.

This property can also be set in the **Specify Date lines** dialog.

	Data Type	Explanation
Property value	System.Boolean	Date line can / cannot be identified Default value: False

LabelPosition

Read Only Property of VcDateLine

This property lets you specify or retrieve the position at which the annotation of the date line shall be displayed and can also be set in the **Edit Date Line** dialog.

	Data Type	Explanation

LineColor

Property of VcDateLine

This property lets you set or retrieve the line color of a date line. This property also can be set in the **Edit Date Line** dialog.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255} Default value: 255. Visual Basic: RGB (255, 0, 0)

Example Code VB.NET

```
Dim dateLine As VcDateLine

dateLine = VcGantt1.DateLineCollection.DateLineByName("DateLine1")
dateLine.LineColor = Color.Blue
```

Example Code C#

```
VcDateLine dateLine = vcGantt1.DateLineCollection.DateLineByName("DateLine1");
dateLine.LineColor = Color.LightSteelBlue;
```

LineThickness**Property of VcDateLine**

This property lets you set or retrieve the line thickness of a date line.

If you set this property to values between 1 and 4, an absolute line thickness is defined in pixels. Irrespective of the zoom factor a line will always show the same line thickness in pixels. When printing though, the line thickness is adapted for the sake of legibility and becomes dependent of the zoom factor:

Value	Points	mm
1	1/2 point	0.09 mm
2	1 point	0.18 mm
3	3/2 points	0.26 mm
4	2 points	0.35 mm

A point equals 1/72 inch and represents the unit of the font size.

If you set this property to values between 5 and 1,000, the line thickness is defined in 1/100 mm, so the lines will be displayed in a true thickness in pixels that depends on the zoom factor.

This property also can be set in the **Edit Date Line** dialog.

	Data Type	Explanation
Property value	System.Int16	Line thickness LineType {1...4}: line thickness in pixels LineType {5...1000}: line thickness in 1/100 mm Default value: As defined in the dialog

Example Code VB.NET

```
Dim dateLine As VcDateLine

dateLine = VcGantt1.DateLineCollection.DateLineByName("DateLine1")
dateLine.LineType = VcLineType.vcSolid
dateLine.LineThickness = 3
```

Example Code C#

```
VcDateLine dateLine = vcGantt1.DateLineCollection.DateLineByName("DateLine1");
dateLine.LineType = VcLineType.vcSolid;
dateLine.LineThickness = 3;
```

LineType**Property of VcDateLine**

This property lets you set or retrieve the line type of a date line. This property also can be set in the **Edit Date Line** dialog.

	Data Type	Explanation
Property value	VcLineType	Line type Default value: vcSolid
	Possible Values: .vcDashed 4 .vcDashed 4 .vcDashedDotted 5 .vcDashedDotted 5 .vcDotted 3 .vcDotted 3 .vcLineType0 100 .vcLineType1 101 .vcLineType10 110 .vcLineType11 111 .vcLineType12 112 .vcLineType13 113 .vcLineType14 114 .vcLineType15 115 .vcLineType16 116 .vcLineType17 117 .vcLineType18 118 .vcLineType2 102 .vcLineType3 103 .vcLineType4 104 .vcLineType5 105 .vcLineType6 106 .vcLineType7 107 .vcLineType8 108	Line dashed Line dashed Line dashed-dotted Line dashed-dotted Line dotted Line dotted Line Type 0 <hr/> Line Type 1 <hr/> Line Type 10 <hr/> Line Type 11 <hr/> Line Type 12 <hr/> Line Type 13 <hr/> Line Type 14 <hr/> Line Type 15 <hr/> Line Type 16 <hr/> Line Type 17 <hr/> Line Type 18 <hr/> Line Type 2 <hr/> Line Type 3 <hr/> Line Type 4 <hr/> Line Type 5 <hr/> Line Type 6 <hr/> Line Type 7 <hr/> Line Type 8 <hr/>

.vcLineType9 109	Line Type 9
.vcNone 1	No line type assigned
.vcNone 1	No line type
.vcSolid 2	Line solid
.vcSolid 2	Line solid

Example Code VB.NET

```
Dim dateLine As VcDateLine

dateLine = VcGantt1.DateLineCollection.DateLineByName("DateLine1")
dateLine.LineType = VcLineType.vcSolid
```

Example Code C#

```
VcDateLine dateLine = vcGantt1.DateLineCollection.DateLineByName("DateLine1");
dateLine.LineType = VcLineType.vcSolid;
```

Movable

Property of VcDateLine

This property lets you set or retrieve whether a date line can be moved interactively. This property also can be set in the **Specify Date Lines** dialog.

	Data Type	Explanation
Property value	System.Boolean	Movable (True)/ not Movable (False) Default value: True

Example Code VB.NET

```
Dim dateLine As VcDateLine

dateLine = VcGantt1.DateLineCollection.DateLineByName("DateLine1")
dateLine.Movable = False
```

Example Code C#

```
VcDateLine dateLine = vcGantt1.DateLineCollection.DateLineByName("DateLine1");
dateLine.Movable = false;
```

Name

Read Only Property of VcDateLine

This property lets you retrieve the name of a date line.

	Data Type	Explanation
Property value	System.String	Name of the date line

Example Code VB.NET

```

Dim datelineCltn As VcDateLineCollection
Dim dateline As VcDateLine

datelineCltn = VcGantt1.DateLineCollection
For Each dateline In datelineCltn
    ListBox1.Items.Add(dateline.Name)
Next

```

Example Code C#

```

VcDateLineCollection datelineCltn = vcGantt1.DateLineCollection;

foreach (VcDateLine dateline in datelineCltn)
{
    ListBox.Items.Add(dateline.Name);
}

```

Priority

Property of VcDateLine

This property lets you specify or retrieve the priority of a date line. If two objects are located at the same position in the diagram, the object of higher priority is displayed in front of the objects of lower priority. By default, grids are of the lowest priority. Nodes are assigned the value 0 and thus the highest priority of all objects. By default, date lines are displayed behind nodes, but in front of calendar grids and date line grids. If you want a date line to be displayed in front of the nodes, you must set its priority to a positive value. This property also can be set in the **Specify Date Lines** dialog.

	Data Type	Explanation
Property value	System.Int16	Priority value Default value: 0

Example Code VB.NET

```

Dim dateLine As VcDateLine

dateLine = VcGantt1.DateLineCollection.DateLineByName("DateLine1")
dateLine.Priority = 10

```

Example Code C#

```

VcDateLine dateLine = vcGantt1.DateLineCollection.DateLineByName("DateLine1");
dateLine.Priority = 10;

```

SnapTarget

Read Only Property of VcDateLine

This property lets you set or retrieve whether this date line has a snap target at the date.

	Data Type	Explanation
Property value	System.Boolean	Snap target is/is not defined at the date of this date line

Specification

Read Only Property of VcDateLine

This property lets you retrieve the specification of a date line. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored without problems to text files or data bases. This allows for persistency. A specification can be used to create a date line by the method **VcDateLineCollection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the date line

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection
Dim dateLine As VcDateLine

dateLineCltn = VcGantt1.DateLineCollection
dateLine = dateLineCltn.FirstDateLine
MsgBox(dateLine.Specification)
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
VcDateLine dateLine = dateLineCltn.FirstDateLine();
MessageBox.Show(dateLine.Specification);
```

Text

Property of VcDateLine

This property lets you set or retrieve an annotation text for the date line. This property also can be set in the **Specify Date Lines** dialog.

	Data Type	Explanation
Property value	System.String	Annotation

Example Code VB.NET

```
Dim dateLine As VcDateLine

dateLine = VcGantt1.DateLineCollection.DateLineByName("DateLine1")
dateLine.Text = "Stichtag"
```

Example Code C#

```
VcDateLine dateLine = vcGantt1.DateLineCollection.DateLineByName("DateLine1");
dateLine.Text = "Stichtag";
```

TurningAnnotationEnabled**Property of VcDateLine**

This property lets you specify or retrieve whether the annotation of the date line is turned by 90 degrees. It can also be set in the **Edit Date Line** dialog.

	Data Type	Explanation
Property value	System.Boolean	Annotation of date line is/is not turned by 90 degrees

UpdateBehaviorName**Property of VcDateLine**

This property lets you set or retrieve the name of the UpdateBehavior.

	Data Type	Explanation
Property value	System.String	Name of the UpdateBehavior

Visible**Property of VcDateLine**

This property lets you set or retrieve the visibility of a date line. This property also can be set in the **Specify Date Lines** dialog.

	Data Type	Explanation
Property value	System.Boolean	Date line visible/invisible Default value: True

Example Code VB.NET

```
Dim dateLine As VcDateLine

dateLine = VcGantt1.DateLineCollection.DateLineByName("DateLine1")
dateLine.Visible = False
```

Example Code C#

```
VcDateLine dateLine = vcGantt1.DateLineCollection.DateLineByName("DateLine1");
dateLine.Visible = false;
```

VisibleDataFieldIndex

Property of VcDateLine

This property lets you set or retrieve the index of the data field to assign a visibility mode to the individual date line. The property can also be set in the **Specify Date Lines** dialog.

	Data Type	Explanation
Property value	System.Int16	Index of the data field which contains the visibility mode

VisibleMapName

Property of VcDateLine

This property lets you set or retrieve the name of a map (type vcTextMap) to set the visibility mode. If set to "", no map will be used. If a map name and additionally a data field index is specified by the property **VisibilityDataFieldIndex**, the visibility mode is selected by the map. This property also can be set in the **Specify Date lines** dialog. If no data field entry from the map applies, the visibility will adopt the value set in the dialog.

	Data Type	Explanation
Property value	System.String	Name of the map that contains the visibility mode

Methods

PutInOrderAfter

Method of VcDateLine

This method lets you set the date line behind a date line specified by name, within the DateLineCollection. If you set the name to "", the date line will be put in the first position. The order of the date lines within the collection determines the order by which they are displayed.

	Data Type	Explanation
Parameter: ↔ refName	System.String	Name of the date line behind which the current date line is to be put.

Return value	Void	
--------------	------	--

Example Code VB.NET

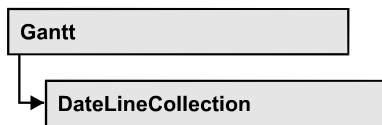
```
Dim datLinCltn As VcDateLineCollection
Dim datLin1 As VcDateLine
Dim datLin2 As VcDateLine

datLinCltn = VcGantt1.DateLineCollection()
datLin1 = datLinCltn.Add("datLin1")
datLin2 = datLinCltn.Add("datLin2")
datLin1.PutInOrderAfter("datLin2")
datLinCltn.Update()
```

Example Code C#

```
VcDateLineCollection datLinCltn = vcGantt1.DateLineCollection;
VcDateLine datLin1 = datLinCltn.Add("datLin1");
VcDateLine datLin2 = datLinCltn.Add("datLin2");
datLin1.PutInOrderAfter("datLin2");
datLinCltn.Update();
```

7.26 VcDateLineCollection



An object of the type **VcDateLineCollection** automatically contains all available date lines. You can access all objects in an iterative loop by **For Each dateLine In dateLineCollection** or by the methods **First...** and **Next...**. You can access a single date line using the methods **DateLineByName** and **DateLineByIndex**. The number of date lines in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the date lines in the corresponding way.

Properties

- Count

Methods

- Add
- AddBySpecification
- Copy
- DateLineByIndex
- DateLineByName
- FirstDateLine
- GetEnumerator
- NextDateLine
- Remove
- Update

Properties

Count

Read Only Property of VcDateLineCollection

This property lets you retrieve the number of date lines contained in the date line collection.

	Data Type	Explanation
Parameter: ⇒ Rückgabewert	System.Int32	Number of data lines
Property value	System.Int32	Number of date lines

Example Code VB.NET

```
Dim numberOfDateLine As Integer

numberOfDateLine = VcGantt1.DateLineCollection.Count
```

Example Code C#

```
int numberOfDateLines = vcGantt1.DateLineCollection.Count;
```

Methods

Add

Method of VcDateLineCollection

By this method you can create a date line as a member of the DateLineCollection. If the name was not used before, the new date line object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned. To make the new date line visible in the diagram, the date line collection needs to be updated by the **Update** call.

	Data Type	Explanation
Parameter: ⇒ dateLineName	System.String	name of date line
Return value	VcDateLine	New date line object

Example Code VB.NET

```
newDateLine = VcGantt1.DateLineCollection.Add("dateLine1")
```

Example Code C#

```
newDateLine = vcGantt1.DateLineCollection.Add("dateLine1");
```

AddBySpecification

Method of VcDateLineCollection

By this method you can create a date line by a date line specification. This way of creating allows date line objects to become persistent. The

specification of a data line can be saved and re-loaded (see VcDateLine property **Specification**). In a subsequent session, the date line can be created again from the specification and is identified by its name. To make the new date line visible in the diagram, the date line collection needs to be updated by the **Update** call.

	Data Type	Explanation
Parameter: ⇒ specification	System.String	date line specification
Return value	VcDateLine	New date line object

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection

dateLineCltn = VcGantt1.DateLineCollection
dateLineCltn.AddBySpecification(textSpecification)
dateLineCltn.Update()
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
dateLineCltn.AddBySpecification(textSpecification);
dateLineCltn.Update();
```

Copy

Method of VcDateLineCollection

By this method you can copy a date line. If the date line that is to be copied exists, and if the name for the new date line does not yet exist, the new date line object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned. To make the copied date line visible in the diagram, the date line collection needs to be updated by the **Update** call.

	Data Type	Explanation
Parameter: ⇒ dateLineName	System.String	Name of the date line to be copied
⇒ newDateLineName	System.String	Name of the new date line
Return value	VcDateLine	date lineobject

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection

dateLineCltn = VcGantt1.DateLineCollection
dateLineCltn.Copy("DateLineOne", "NewDateLine")
dateLineCltn.Update()
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
dateLineCltn.Copy("DateLineOne", "NewDateLine");
dateLineCltn.Update();
```

DateLineByIndex**Method of VcDateLineCollection**

This method lets you access a data line by its index. If a date line does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the date line
Return value	VcDateLine	Date line object returned

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection
Dim dateLine As VcDateLine

dateLineCltn = VcGantt1.DateLineCollection
dateLine = dateLineCltn.DateLineByIndex(0)
MsgBox(dateLine.Name)
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
VcDateLine dateLine = dateLineCltn.DateLineByIndex(0);
MessageBox.Show(dateLine.Name);
```

DateLineByName**Method of VcDateLineCollection**

By this method you can retrieve a date line by its name. If a date line of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ dateLineName	System.String	Name of the date line
Return value	VcDateLine	Date line

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection
Dim dateLine As VcDateLine

dateLineCltn = VcGantt1.DateLineCollection
dateLine = dateLineCltn.DateLineByName("DateLineOne")
MsgBox(dateLine.Name)
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
VcDateLine dateLine = dateLineCltn.DateLineByName("DateLineOne");
MessageBox.Show(dateLine.Name);
```

FirstDateLine**Method of VcDateLineCollection**

This method can be used to access the initial value, i.e. the first date line of a date line collection, and to continue in a forward iteration loop by the method **NextDateLine** for the date lines following. If there is no date line in the date line collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcDateLine	First date line

Example Code VB.NET

```
Dim datelineCltn As VcDateLineCollection
Dim dateline As VcDateLine

datelineCltn = VcGantt1.DateLineCollection
dateline = datelineCltn.FirstDateLine

While Not dateline Is Nothing
    ListBox1.Items.Add(dateline.Name)
    dateline = datelineCltn.NextDateLine
End While
```

Example Code C#

```
VcDateLineCollection datelineCltn = vcGantt1.DateLineCollection;
VcDateLine dateline = datelineCltn.FirstDateLine();

while (dateline != null)
{
    ListBox.Items.Add(dateline.Name);
    dateline = datelineCltn.NextDateLine();
}
```

GetEnumerator

Method of VcDateLineCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the date line objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection
Dim dateLine As VcDateLine

dateLineCltn = VcGantt1.DateLineCollection
For Each dateLine In dateLineCltn
    ListBox1.Items.Add(dateLine.Name)
Next
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
foreach (VcDateLine dateLine in dateLineCltn)
    listBox1.Items.Add(dateLine.Name);
```

NextDateLine

Method of VcDateLineCollection

This method can be used in a forward iteration loop to retrieve subsequent date lines from a date line collection after initializing the loop by the method **FirstDateLine**. If there is no date line left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcDateLine	Subsequent date line

Example Code VB.NET

```
Dim datelineCltn As VcDateLineCollection
Dim dateline As VcDateLine

datelineCltn = VcGantt1.DateLineCollection
dateline = datelinecltn.FirstDateLine

While Not dateline Is Nothing
    ListBox1.Items.Add(dateline.Name)
    dateline = datelineCltn.NextDateLine
End While
```

Example Code C#

```
VcDateLineCollection datelineCltn = vcGantt1.DateLineCollection;
VcDateLine dateline = datelineCltn.FirstDateLine();

while (dateline != null)
{
    ListBox.Items.Add(dateline.Name);
    dateline = datelineCltn.NextDateLine();
}
```

Remove**Method of VcDateLineCollection**

This method lets you delete a date line. To make the deletion visible in the diagram, the date line collection needs to be updated by the **Update** call.

	Data Type	Explanation
Parameter: ⇒ dateLineName	System.String	date line name
Return value	System.Boolean	date line deleted (True)/not deleted (False)

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection
Dim dateLine As VcDateLine

dateLineCltn = VcGantt1.DateLineCollection
dateLine = dateLineCltn.DateLineByIndex(0)
dateLineCltn.Remove(dateLine.Name)
dateLineCltn.Update()
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
VcDateLine dateLine = dateLineCltn.DateLineByIndex(0);
dateLineCltn.Remove(dateLine.Name);
dateLineCltn.Update();
```

Update**Method of VcDateLineCollection**

This method lets you update a date line collection after having modified it.

	Data Type	Explanation
Return value	System.Boolean	update successful (True)/ not successful (False)

Example Code VB.NET

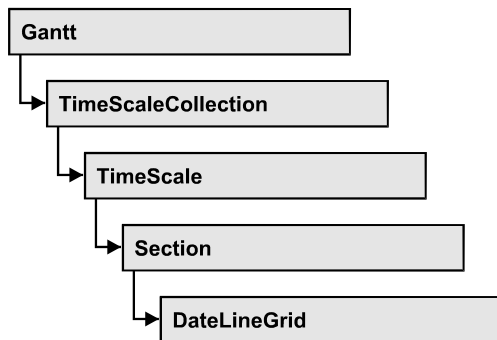
```
Dim dateLineCltn As VcDateLineCollection
Dim dateLine As VcDateLine

dateLineCltn = VcGantt1.DateLineCollection
dateLine = dateLineCltn.DateLineByIndex(0)
dateLineCltn.Remove(dateLine.Name)
dateLineCltn.Update()
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
VcDateLine dateLine = dateLineCltn.DateLineByIndex(0);
dateLineCltn.Remove(dateLine.Name);
dateLineCltn.Update();
```

7.27 VcDateLineGrid



An object of the type **VcDateLineGrid** is a predefined grid for highlighting time periods (days, weeks, months, ...) by vertical lines.

Properties

- AdjustToReferenceDate
- AnnotationAtBottom
- AnnotationAtCenter
- AnnotationAtTop
- FormatName
- HorAlignment
- LineColor
- LineColorDataFieldIndex
- LineColorMapName
- LineThickness
- LineType
- ObservedDST
- Period
- Priority
- ReferenceDate
- SnapTarget
- TurningAnnotationEnabled
- Unit
- UseReferenceDate
- Visible
- VisibleDataFieldIndex
- VisibleMapName

Properties

AdjustToReferenceDate

Property of VcDateLineGrid

The lines of a line grid by default are positioned on the beginning of a time unit, for example on 00:00 h of a day. This property lets you position the line grid on a different value of the time unit, i.e. the one defined by the reference date, for example on 13:17 of a day. The reference date you can set by the property **set/getReferenceDate**.

	Data Type	Explanation
Property value	System.Boolean	Date line grid positioned (False) / not positioned on reference date (False) Default value: False

AnnotationAtBottom

Property of VcDateLineGrid

This property lets you position the annotations of the lines in the line grid at the bottom of the Gantt graph, or retrieve whether they are there. Also see **set/getAnnotationAtCenter** and **set/getAnnotationAtTop**.

	Data Type	Explanation
Property value	System.Boolean	Date line grid annotations positioned at bottom (True) / not at bottom (False) Default value: False

AnnotationAtCenter

Property of VcDateLineGrid

This property lets you position the annotations of the lines in the line grid at the center of the Gantt graph, or retrieve whether they are there. Also see **set/getAnnotationAtBottom** and **set/getAnnotationAtTop**.

	Data Type	Explanation
Property value	System.Boolean	Date line grid annotations positioned in the center (True) / not in the center (False) Default value: False

AnnotationAtTop

Property of VcDateLineGrid

This property lets you position the annotations of the lines in the line grid at the top of the Gantt graph, or retrieve whether they are there. Also see **set/getAnnotationAtCenter** and **set/getAnnotationAtBottom**.

	Data Type	Explanation
Property value	System.Boolean	Date line grid annotations positioned at top (True) / not at top (False) Default value: False

FormatName

Property of VcDateLineGrid

This property lets you set or retrieve the name of the line format of this date line grid.

	Data Type	Explanation
Property value	System.String	Name of the line format

HorAlignment

Property of VcDateLineGrid

This property lets you set or retrieve the horizontal alignment of the line annotations.

	Data Type	Explanation
Property value	VcHorizontalAlignment	Horizontal alignment
	Possible Values: .vcHorCenterAligned - 1	horizontally centered
	.vcLeftAligned -3	left aligned

.vcRightAligned -2	right aligned
--------------------	---------------

LineColor

Property of VcDateLineGrid

This property lets you set or retrieve the color of a date line grid.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255}) Default value: 255. Visual Basic: RGB (255, 0, 0)

Example Code VB.NET

```
Dim dateLineGrid As VcDateLineGrid

dateLineGrid = VcGantt1.TimeScaleCollection.Active.Section(0).DateLineGrid(0)
dateLineGrid.LineColor = Color.Blue
```

Example Code C#

```
VcDateLineGrid dateLineGrid =
vcGantt1.TimeScaleCollection.Active.get_Section(0).get_DateLineGrid(0);
dateLineGrid.LineColor = Color.LightSteelBlue;
```

LineColorDataFieldIndex

Read Only Property of VcDateLineGrid

This property lets you set or retrieve the data field index to be used with a map specified by the property **LineColorMapName**. If you set this property to -1, no map will be used.

	Data Type	Explanation
Property value	System.Int32	Data field index

LineColorMapName

Property of VcDateLineGrid

This property lets you set or retrieve the name of a map for the line color. If set to "" or if the property **LineColorDataFieldIndex** is set to **-1**, then no map will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

LineThickness

Property of VcDateLineGrid

This property lets you set or retrieve the line thickness of the grid lines. If you set this property to values between 1 and 4, an absolute line thickness is defined in pixels. Irrespective of the zoom factor a line will always show the same line thickness in pixels. When printing though, the line thickness is adapted for the sake of legibility and becomes dependent of the zoom factor:

Value	Points	mm
1	1/2 point	0.09 mm
2	1 point	0.18 mm
3	3/2 points	0.26 mm
4	2 points	0.35 mm

A point equals 1/72 inch and represents the unit of the font size.

If you set this property to values between 5 and 1,000, the line thickness is defined in 1/100 mm, so the lines will be displayed in a true thickness in pixels that depends on the zoom factor.

This property also can be set in the **Edit Date Line** dialog.

	Data Type	Explanation
Property value	System.Int16	Line thickness LineType {1...4}: line thickness in pixels LineType {5...1000}: line thickness in 1/100 mm Default value: As defined in the dialog

Example Code VB.NET

```
Dim dateLineGrid As VcDateLineGrid

dateLineGrid = VcGantt1.TimeScaleCollection.Active.Section(0).DateLineGrid(0)
dateLineGrid.LineThickness = 2
```

Example Code C#

```
VcDateLineGrid dateLineGrid =
vcGantt1.TimeScaleCollection.Active.get_Section(0).get_DateLineGrid(0);
dateLineGrid.LineThickness = 2;
```

LineType**Property of VcDateLineGrid**

This property lets you set or retrieve the line type of a date line grid.

	Data Type	Explanation
Property value	VcLineType	Line type Default value: vcDashed
	Possible Values: .vcDashed 4 .vcDashed 4 .vcDashedDotted 5 .vcDashedDotted 5 .vcDotted 3 .vcDotted 3 .vcLineType0 100 .vcLineType1 101 .vcLineType10 110 .vcLineType11 111 .vcLineType12 112 .vcLineType13 113 .vcLineType14 114 .vcLineType15 115 .vcLineType16 116 .vcLineType17 117 .vcLineType18 118 .vcLineType2 102 .vcLineType3 103 .vcLineType4 104 .vcLineType5 105 .vcLineType6 106 .vcLineType7 107 .vcLineType8 108	Line dashed Line dashed Line dashed-dotted Line dashed-dotted Line dotted Line dotted Line Type 0 <hr/> Line Type 1 <hr/> Line Type 10 <hr/> Line Type 11 <hr/> Line Type 12 <hr/> Line Type 13 <hr/> Line Type 14 <hr/> Line Type 15 <hr/> Line Type 16 <hr/> Line Type 17 <hr/> Line Type 18 <hr/> Line Type 2 <hr/> Line Type 3 <hr/> Line Type 4 <hr/> Line Type 5 <hr/> Line Type 6 <hr/> Line Type 7 <hr/> Line Type 8 <hr/>

.vcLineType9 109	Line Type 9
.vcNone 1	No line type assigned
.vcNone 1	No line type
.vcSolid 2	Line solid
.vcSolid 2	Line solid

Example Code VB.NET

```
Dim dateLineGrid As VcDateLineGrid

dateLineGrid = VcGantt1.TimeScaleCollection.Active.Section(0).DateLineGrid(0)
dateLineGrid.LineType = VcLineType.vcSolid
```

Example Code C#

```
VcDateLineGrid dateLineGrid =
vcGantt1.TimeScaleCollection.Active.get_Section(0).get_DateLineGrid(0);
dateLineGrid.LineType = VcLineType.vcSolid;
```

ObserveDST**Property of VcDateLineGrid**

This property lets you set or retrieve whether for this line grid daylight saving time is considered or not.

	Data Type	Explanation
Property value	VcDateLineGridObserveDST	Daylight saving time is/is not considered.
	Possible Values: .vcGODDefault 9999 .vcGODNo 0 .vcGODYes 1	Default setting from .INI file is used Daylight saving time is not considered Daylight saving time is considered

Period**Property of VcDateLineGrid**

This property lets you set or retrieve after how many time units a grid line is drawn. The distance between two grid lines is given by the product of the unit (property **Unit**) and the period (property **Period**).

	Data Type	Explanation
Property value	System.Int32	Period value
		Default value: 1

Example Code VB.NET

```
Dim dateLineGrid As VcDateLineGrid

dateLineGrid = VcGantt1.TimeScaleCollection.Active.Section(0).DateLineGrid(0)
dateLineGrid.Unit = VcGridUnit.vcGridUnitDay
dateLineGrid.Period = 1
```

Example Code C#

```
VcDateLineGrid dateLineGrid =
vcGantt1.TimeScaleCollection.Active.get_Section(0).get_DateLineGrid(0);
dateLineGrid.Unit = VcGridUnit.vcGridUnitDay;
dateLineGrid.Period = 1;
```

Priority

Property of VcDateLineGrid

This property lets you set or retrieve the priority of a date line grid.

If two objects are located at the same position in the diagram, the object of higher priority is displayed in front of the objects of lower priority. By default, grids are of the lowest priority. Nodes are assigned the value 0 and thus the highest priority of all objects. By default, date line grids are displayed in front of calendar grids, but behind nodes and date lines. If you want a date line grid to be displayed in front of the nodes, you must set its priority to a positive value.

	Data Type	Explanation
Property value	System.Int32	Priority value {-1000...+1000} Default value: -20

Example Code VB.NET

```
Dim dateLineGrid As VcDateLineGrid

dateLineGrid = VcGantt1.TimeScaleCollection.Active.Section(0).DateLineGrid(0)
dateLineGrid.Priority = 10
```

Example Code C#

```
VcDateLineGrid dateLineGrid =
vcGantt1.TimeScaleCollection.Active.get_Section(0).get_DateLineGrid(0);
dateLineGrid.Period = 10;
```

ReferenceDate

Property of VcDateLineGrid

This property lets you set or retrieve the reference date. For the date line grid to actually use the reference date, the property **UseReferenceDate** needs to be set. To adjust the date line grid to the reference date, please see property **AdjustToReferenceDate**.

The reference date shifts the beginning of the grid away from the default start on Monday 0:00 h by the offset specified. For this, the difference between the default start and the reference date is the essential part; the absolute date is not. For example: if you want the grid to start on Tuesday, you can set the reference date to May 6, 2014. You will obtain the same result by setting the reference date to April, 29, 2014. It is the difference between the date given and Monday, which is 1 day. The offset does not have to be specified in days, you can also set a day time, such as 29.4.2014 8:15 h. If you are dealing with an hour grid, only minutes are of relevance at all, so in the latter example the grid offset would be 15 minutes.

	Data Type	Explanation
Property value	System.DateTime	Reference date

SnapTarget

Read Only Property of VcDateLineGrid

This property lets you set or retrieve whether this date line grid has a snap target at the date.

	Data Type	Explanation
Property value	System.Boolean	Snap target is/is not defined at the date of this date line grid

TurningAnnotationEnabled

Property of VcDateLineGrid

This property lets you set or retrieve whether the annotations at the lines of the date line grid can be turned by 90 degrees (vertically).

	Data Type	Explanation
Property value	System.Boolean	The annotations can be turned (True) / were already turned (False) Default value: True

Unit

Property of VcDateLineGrid

This property lets you set or retrieve the unit of a date line grid. The distance between two grid lines is given by the product of unit (property **Unit**) and period (property **Period**).

	Data Type	Explanation
Property value	VcGridUnit	Time unit Default value: vcGridUnitWeek
	Possible Values: .vcGridUnitDay 5 .vcGridUnitHour 6 .vcGridUnitMinute 7 .vcGridUnitMonth 3 .vcGridUnitQuarter 2 .vcGridUnitSecond 8 .vcGridUnitWeek 4 .vcGridUnitYear 1	Grid unit day Grid unit hour Grid unit minute Grid unit month Grid unit quarter Grid unit second Grid unit week Grid unit year

Example Code VB.NET

```
Dim dateLineGrid As VcDateLineGrid

dateLineGrid = VcGantt1.TimeScaleCollection.Active.Section(0).DateLineGrid(0)
dateLineGrid.Unit = VcGridUnit.vcGridUnitDay
```

Example Code C#

```
VcDateLineGrid dateLineGrid =
vcGantt1.TimeScaleCollection.Active.get_Section(0).get_DateLineGrid(0);
dateLineGrid.Period = 1;
dateLineGrid.Unit = VcGridUnit.vcGridUnitDay;
```

UseReferenceDate

Read Only Property of VcDateLineGrid

This property lets you set or retrieve whether the date line grid uses a reference date.

	Data Type	Explanation
Property value	System.Boolean	Date line grid uses (True)/does not use (False) reference date Default value: False

Visible

Property of VcDateLineGrid

This property lets you set or retrieve whether a date line grid is visible.

	Data Type	Explanation
Property value	System.Boolean	Date line grid visible/invisible Default value: True

Example Code VB.NET

```
Dim dateLineGrid As VcDateLineGrid

dateLineGrid = VcGantt1.TimeScaleCollection.Active.Section(0).DateLineGrid(0)
dateLineGrid.Visible = True
```

Example Code C#

```
VcDateLineGrid dateLineGrid =
vcGantt1.TimeScaleCollection.Active.get_Section(0).get_DateLineGrid(0);
dateLineGrid.Visible = true;
```

VisibleDataFieldIndex

Property of VcDateLineGrid

This property lets you set or retrieve the index of the data field to assign a visibility mode to the calendar grid: 1 (for "visible") or 0 (for invisible). This property also can be set in the **DateLineGrid** dialog.

	Data Type	Explanation
Property value	System.Int16	Index of the data field which contains the visibility mode

VisibleMapName

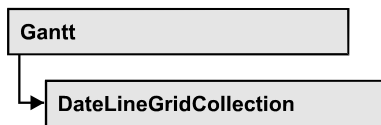
Property of VcDateLineGrid

This property lets you set or retrieve the name of a map (type vcTextMap) to set the visibility mode. If set to "", no map will be used. If a map name and

additionally a data field index is specified by the property **VisibilityData-FieldIndex**, the visibility mode is selected by the map. This property also can be set in the **CalendarGrid** dialog. If no data field entry from the map applies, the visibility value will be adopted from the dialog.

	Data Type	Explanation
Property value	System.String	Name of the map that contains the visibility mode

7.28 VcDateLineGridCollection



An object of the type VcDateLineGridCollection contains all available date line grids. You can access all objects in an iterative loop by **For Each date-LineGrid In dateLineGridCollection** or by the methods **First...** and **Next....** You can access a single date line using the methods **DateLineGridByName** and **DateLineGridByIndex**. The number of date line grids in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the date line grids in the corresponding way.

Properties

- Count

Methods

- Add
- AddBySpecification
- Copy
- DateLineGridByIndex
- DateLineGridByName
- FirstDateLineGrid
- GetEnumerator
- NextDateLineGrid
- Remove
- Update

Properties

Count

Read Only Property of VcDateLineGridCollection

This property lets you retrieve the number of date line grids in the DateLineGridCollection object.

	Data Type	Explanation
Property value	System.Int32	Number of date line grids

Example Code VB.NET

```
Dim numberOfDateLine As Integer

numberOfDateLine = VcGantt1.DateLineCollection.Count
```

Example Code C#

```
int numberOfDateLines = vcGantt1.DateLineCollection.Count;
```

Methods

Add

Method of VcDateLineGridCollection

This method lets you create a date line grid as a member of the DateLineGridCollection. If the name was not used before, the new date line grid object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ DateLineGridName	System.String	name of date line grid
Return value	VcDateLineGrid	New date line grid object

Example Code VB.NET

```
newDateLineGrid = VcGantt1.DateLineGridCollection.Add("dateLineGrid1")
```

Example Code C#

```
newDateLineGrid = vcGantt1.DateLineGridCollection.Add("dateLineGrid1");
```

AddBySpecification

Method of VcDateLineGridCollection

This method lets you create a date line grid by using a date line grid specification. This way of creating allows date line grid objects to become persistent. The specification of a date line grid can be saved and re-loaded (see VcDateLineGrid property **Specification**). In a subsequent session the

date line grid can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter: ⇒ Specification	System.String	date line grid specification
Return value	VcDateLineGrid	New date line grid object

Example Code VB.NET

```
Dim dateLineGridCltn As VcDateLineGridCollection

dateLineGridCltn = VcGantt1.DateLineGridCollection
dateLineGridCltn.AddBySpecification(textSpecification)
dateLineGridCltn.Update()
```

Example Code C#

```
VcDateLineGridCollection dateLineGridCltn = vcGantt1.DateLineGridCollection;
dateLineGridCltn.AddBySpecification(textSpecification);
dateLineGridCltn.Update();
```

Copy

Method of VcDateLineGridCollection

By this method you can copy a date line grid. If the date line grid that is to be copied exists, and if the name for the new date line grid does not yet exist, the new date line grid object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ DateLineGridName	System.String	Name of the date line grid to be copied
⇒ newDateLineGridName	System.String	Name of the new date line grid
Return value	VcDateLineGrid	date line grid object

Example Code VB.NET

```
Dim dateLineGridCltn As VcDateLineGridCollection

dateLineGridCltn = VcGantt1.DateLineGridCollection
dateLineGridCltn.Copy("DateLineGridOne", "NewDateLineGrid")
dateLineGridCltn.Update()
```

Example Code C#

```
VcDateLineGridCollection dateLineGridCltn = vcGantt1.DateLineGridCollection;
dateLineGridCltn.Copy("DateLineGridOne", "NewDateLineGrid");
dateLineGridCltn.Update();
```

DateLineGridByIndex

Method of VcDateLineGridCollection

This method lets you access a date line grid by its index. If a date line grid of the specified index does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the date line grid
Return value	VcDateLineGrid	date line grid object returned

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection
Dim dateLine As VcDateLine

dateLineCltn = VcGantt1.DateLineCollection
dateLine = dateLineCltn.DateLineByIndex(0)
MsgBox(dateLine.Name)
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
VcDateLine dateLine = dateLineCltn.DateLineByIndex(0);
MessageBox.Show(dateLine.Name);
```

DateLineGridByName

Method of VcDateLineGridCollection

This method is used to access a date line grid by its name. If a date line grid of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ DateLineGridName	System.String	Name of the date line grid
Return value	VcDateLineGrid	date line grid

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection
Dim dateLine As VcDateLine

dateLineCltn = VcGantt1.DateLineCollection
dateLine = dateLineCltn.DateLineByName("DateLineOne")
MsgBox(dateLine.Name)
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
VcDateLine dateLine = dateLineCltn.DateLineByName("DateLineOne");
MessageBox.Show(dateLine.Name);
```

FirstDateLineGrid**Method of VcDateLineGridCollection**

This method can be used to access the initial value, i.e. the first date line grid of a date line grid collection and then to continue in a forward iteration loop by the method **NextDateLineGrid** for the date line grids following. If there is no date line grid in the DateLineGridCollection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcDateLineGrid	First date line grid

Example Code VB.NET

```
Dim dateLineGridCltn As VcDateLineGridCollection

dateLineGridCltn = VcGantt1.DateLineGridCollection
dateLineGrid = dateLineGridCltn.FirstDateLineGrid
```

Example Code C#

```
VcDateLineGridCollection dateLineGridCltn = vcGantt1.DateLineGridCollection;
VcDateLineGrid dateLineGrid = dateLineGridCltn.FirstDateLineGrid();
```

GetEnumerator**Method of VcDateLineGridCollection**

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the group objects included.

	Data Type	Explanation
Return value	System.Object	Reference object

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection
Dim dateLine As VcDateLine

dateLineCltn = VcGantt1.DateLineCollection
For Each dateLine In dateLineCltn
    ListBox1.Items.Add(dateLine.Name)
Next
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
foreach (VcDateLine dateLine in dateLineCltn)
    listBox1.Items.Add(dateLine.Name);
```

NextDateLineGrid**Method of VcDateLineGridCollection**

This method can be used in a forward iteration loop to retrieve subsequent date line grids from a DateLineGridCollection after initializing the loop by the method **FirstDateLineGrid**. If there is no date line grid left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcDateLineGrid	Subsequent date line grid

Example Code VB.NET

```
Dim dateLineGridCltn As VcDateLineGridCollection
Dim dateLineGrid As VcDateLineGrid

dateLineGridCltn = VcGantt1.DateLineGridCollection
dateLineGrid = dateLineGridCltn.FirstDateLineGrid

While Not dateLineGrid Is Nothing
    ListBox1.Items.Add(dateLineGrid.Name)
    dateLineGrid = dateLineGridCltn.NextDateLineGrid
End While
```

Example Code C#

```
VcDateLineGridCollection dateLineGridCltn = vcGantt1.DateLineGridCollection;
VcDateLineGrid dateLineGrid = dateLineGridCltn.FirstDateLineGrid();

while (dateLineGrid != null)
{
    ListBox.Items.Add(dateLineGrid.Name);
    dateLineGrid = dateLineGridCltn.NextDateLineGrid();
}
```

Remove**Method of VcDateLineGridCollection**

This method lets you delete a date line grid. If the date line grid is used in another object, it cannot be deleted. Then False will be returned, otherwise True.

	Data Type	Explanation
Parameter: ⇒ DateLineGridName	System.String	date line grid name

Return value	System.Boolean	date line grid deleted (True)/not deleted (False)
---------------------	----------------	---

Example Code VB.NET

```
Dim dateLineGridCltn As VcDateLineGridCollection
Dim dateLineGrid As VcDateLineGrid

dateLineGridCltn = VcGantt1.DateLineGridCollection
dateLineGrid = dateLineGridCltn.DateLineGridByIndex(0)
dateLineGridCltn.Remove(dateLineGrid.Name)
dateLineGridCltn.Update()
```

Example Code C#

```
VcDateLineGridCollection dateLineGridCltn = vcGantt1.DateLineGridCollection;
VcDateLineGrid dateLineGrid = dateLineGridCltn.DateLineGridByIndex(0);
dateLineGridCltn.Remove(dateLineGrid.Name);
dateLineGridCltn.Update();
```

Update

Method of VcDateLineGridCollection

This method has to be used when date line grid modifications have been carried out. The method **Update** updates all objects that are concerned by the date line grid you have edited. You should call this method at the end of the code that defines the date line grids and the date line grid collection. Otherwise the update will be processed before all date line grid definitions are processed.

	Data Type	Explanation
Return value	System.Boolean	update successful (True)/ not successful (False)

Example Code VB.NET

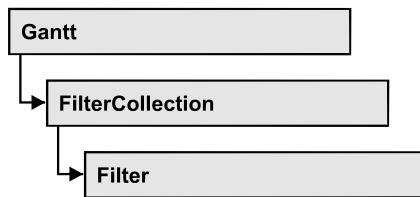
```
Dim dateLineGridCltn As VcDateLineGridCollection
Dim dateLineGrid As VcDateLineGrid

dateLineGridCltn = VcGantt1.DateLineGridCollection
dateLineGrid = dateLineGridCltn.DateLineGridByIndex(0)
dateLineGridCltn.Remove(dateLineGrid.Name)
dateLineGridCltn.Update()
```

Example Code C#

```
VcDateLineGridCollection dateLineGridCltn = vcGantt1.DateLineGridCollection;
VcDateLineGrid dateLineGrid = dateLineGridCltn.DateLineGridByIndex(0);
dateLineGridCltn.Remove(dateLineGrid.Name);
dateLineGridCltn.Update();
```

7.29 VcFilter



An object of the type VcFilter contains subconditions (VcFilterSubCondition), p.e. permitted values to be compared to the data fields of a node or a link, so that the filter conditions may or may not apply to an object. Filters are used p.e. to assign a format to an activity. Only if the filter is valid after the subconditions have been modified, the modified subconditions will become valid. Otherwise the former filter subconditions will remain valid. This can be controlled via the methods VcFilter.IsValid and VcFilterSubCondition.IsValid.

Properties

- DataDefinitionTable
- DatesWithHourAndMinute
- Name
- Specification
- StringsCaseSensitive
- SubCondition
- SubConditionCount

Methods

- AddSubCondition
- CopySubCondition
- Evaluate
- GetEnumerator
- IsValid
- RemoveSubCondition

Properties

DataDefinitionTable

Property of VcFilter

This property lets you enquire whether the filter is a filter for nodes (vcMainData) or for links (vcRelations). This property can be modified only if the filter does not contain conditions.

	Data Type	Explanation
Property value	VcDataTableType Possible Values: .vcMainData 0 .vcMaindata 0 .vcRelations 1 .vcRelations 1	Type of data definition table Definition of node data table type vcMaindata (for nodes) Definition of link data table type vcRelations (for links)

DatesWithHourAndMinute

Property of VcFilter

This property lets you set or retrieve whether the comparison of conditions that contain dates takes into account hours and minutes. This setting can only be modified if there is at least one subcondition that compares dates. Otherwise the property value is always False.

	Data Type	Explanation
Property value	System.Boolean	Hours and minutes are compared (True)/ not compared (False)

Name

Property of VcFilter

This property lets you set or retrieve the name of the filter.

	Data Type	Explanation
Property value	System.String	Name of the filter

Example Code VB.NET

```

Dim filterCltn As VcFilterCollection
Dim filter As VcFilter

filterCltn = VcGantt1.FilterCollection

For Each filter In filterCltn
    ListBox1.Items.Add(filter.Name)
Next

```

Example Code C#

```

VcFilterCollection filterCltn = vcGantt1.FilterCollection;

foreach (VcFilter filter in filterCltn)
{
    ListBox.Items.Add(filter.Name);
}

```

Specification

Read Only Property of VcFilter

This property lets you retrieve the specification of a filter. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored without problems to text files or databases. This allows for persistency. A specification can be used to create a filter by the method **VcFilterCollection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the filter

Example Code VB.NET

```

Dim filterCltn As VcFilterCollection
Dim filter As VcFilter

filterCltn = VcGantt1.FilterCollection
filter = filterCltn.FirstFilter
MsgBox(filter.Specification)

```

Example Code C#

```

VcFilterCollection boxCltn = vcGantt1.FilterCollection;
VcFilter filter = filterCltn.FirstFilter();
MessageBox.Show(filter.Specification);

```

StringsCaseSensitive

Property of VcFilter

This property lets you enquire/set whether subconditions that contain strings are case-sensitive.

	Data Type	Explanation
Property value	System.Boolean	Case-sensitive (True)/not case-sensitive (False)

SubCondition

Read Only Property of VcFilter

This property lets you access a VcFilterSubCondition object by its index.

The property SubCondition is an Indexed Property, which in C# is addressed by the method get_SubCondition (index).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the filter subcondition {0 ... VcFilter.SubConditionCount-1}
Property value	VcFilterSubCondition	Filter subcondition object

SubConditionCount

Read Only Property of VcFilter

This property lets you enquire the number of filter subconditions.

	Data Type	Explanation
Property value	System.Int16	Number of filter subconditions

Methods

AddSubCondition

Method of VcFilter

This method lets you create a new filter condition in the collection of the filter conditions. Its position is specified by the index. The corresponding VcFilterSubCondition object will be returned.

Default properties of this object:

- DataFieldIndex: -1
- Operator: vcInvalidOp
- ComparisonValueAsString: "<INVALID>"
- ConnectionOperator: vcInvalidConnOp.

	Data Type	Explanation
Parameter: ⇒ atIndex	System.Int16	Index of the new filter subcondition {0 to VcFilter.SubConditionCount and –1 for "at the end of the Collection" (identical with the value VcFilter.SubConditionCount)}
Return value	VcFilterSubCondition	Filter subcondition object

CopySubCondition

Method of VcFilter

This method lets you copy a filter subcondition by its index. The new filter subcondition will be inserted into the collection at the position specified by the index. It will be returned as a VcFilterSubCondition object.

	Data Type	Explanation
Parameter: ⇒ fromIndex ⇒ atIndex	System.Int16 System.Int16	Index of the filter subcondition to be copied Index of the new filter subcondition {0 to VcFilter.SubConditionCount and –1 for "at the end of the Collection" (identical with the value VcFilter.SubConditionCount)}
Return value	VcFilterSubCondition	Filter subcondition object

Evaluate

Method of VcFilter

This methods lets you check whether the specified filter applies for a certain data record or not. You should only pass objects that are internally linked with data records of the data tables. Those are **VcNode**, **VcLink**, **VcGroup**,

VcDataRecord. If an object is passed that is not listed, an exception will be triggered.

	Data Type	Explanation
Parameter: ⇒ dataObjectParam	Variant	Data record object
Return value	Boolean	Filter applies for data record (True)/does not apply (False)

GetEnumerator

Method of VcFilter

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the condition objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

Example Code VB.NET

```
Dim filter As VcFilter
Dim filterCond As VcFilterSubCondition

filter = VcGantt1.FilterCollection.FirstFilter

For Each filterCond In filter
    Debug.Write(filterCond.Index)
Next
```

Example Code C#

```
VcFilter filter = vcGantt1.FilterCollection.FirstFilter();
foreach(VcFilterSubCondition filterCond in filter)
{
    Console.Write(filterCond.Index);
}
```

IsValid

Method of VcFilter

This property checks whether all filter subconditions are correct. The correctness of all subconditions is the condition that changed filter subconditions become valid. Otherwise the former subconditions will remain valid.

	Data Type	Explanation
Return value	System.Boolean	Filter subconditions correct (True)/ not correct (False)

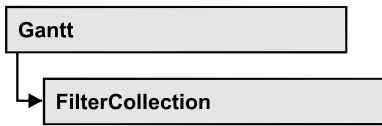
RemoveSubCondition

Method of VcFilter

This method lets you delete a filter subcondition by its index.

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the filter subcondition to be removed

7.30 VcFilterCollection



An object of the type VcFilterCollection automatically contains all available filters. You can access all objects in an iterative loop by **For Each filter In FilterCollection** or by the methods **First...** and **Next...**. You can access a single filter using the methods **FilterByName** and **FilterByIndex**. The number of filters in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the filters in the corresponding way.

Properties

- Count
- MarkedNodesFilter

Methods

- Add
- AddBySpecification
- Copy
- FilterByIndex
- FilterByName
- FirstFilter
- GetEnumerator
- NextFilter
- Remove

Properties

Count

Read Only Property of VcFilterCollection

This property lets you retrieve the number of filters in the filter collection.

	Data Type	Explanation
Property value	System.Int32	Number of filters

Example Code VB.NET

```
Dim filterCltn As VcFilterCollection
Dim numberOfFilters As Integer

filterCltn = VcGantt1.FilterCollection
numberOfFilters = filterCltn.Count
```

Example Code C#

```
VcFilterCollection filterCltn = vcGantt1.FilterCollection;
int numberOfFilters = filterCltn.Count;
```

MarkedNodesFilter**Read Only Property of VcFilterCollection**

This property lets you retrieve a constant pseudo-filter that can be used only for **ActiveNodeFilter** for filtering the nodes currently marked (sub-diagram).

	Data Type	Explanation
Property value	VcFilter	Pseudo filter

Example Code VB.NET

```
VcGantt1.ActiveNodeFilter = VcGantt1.FilterCollection.MarkedNodesFilter
```

Example Code C#

```
vcGantt1.ActiveNodeFilter = vcGantt1.FilterCollection.MarkedNodesFilter;
```

Methods**Add****Method of VcFilterCollection**

By this method you can create a filter as a member of the FilterCollection. If the name was not used before, the new filter object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

The new filter automatically refers to the data definition table vcMainData (see VcFilter.DataDefinitionTable). You can select vcRelations instead, as long as the filter does not contain any subconditions.

	Data Type	Explanation
Parameter: ⇒ newName	System.String	Filter name
Return value	VcFilter	New filter object

Example Code VB.NET

```
newFilter = VcGantt1.FilterCollection.Add("foo")
```

Example Code C#

```
newFilter = vcGantt1.FilterCollection.Add("foo");
```

AddBySpecification

Method of VcFilterCollection

This method lets you create a filter by using filter specification. This way of creating allows filter objects to become persistent. The specification of a filter can be saved and re-loaded (see VcFilter property **Specification**). In a subsequent the filter can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter: ⇒ filterSpecification	System.String	Filter specification
Return value	VcFilter	New filter object

Copy

Method of VcFilterCollection

By this method you can copy a filter. If the filter that is to be copied exists, and if the name for the new filter does not yet exist, the new filter object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ fromName	System.String	Name of the filter to be copied
⇒ newName	System.String	Name of the new filter
Return value	VcFilter	Filter object

FilterByIndex

Method of VcFilterCollection

This method lets you access a filter by its index. If a filter does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the filter
Return value	VcFilter	Filter object returned

FilterByName

Method of VcFilterCollection

By this method you can retrieve a filter by its name. If a filter of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ filterName	System.String	Filter name
Return value	VcFilter	Filter

Example Code VB.NET

```
Dim filterCltn As VcFilterCollection
Dim filter As VcFilter

filterCltn = VcGantt1.FilterCollection
filter = filterCltn.FilterByName("Department A")
```

Example Code C#

```
VcFilterCollection filterCltn = vcGantt1.FilterCollection;
VcFilter filter = filterCltn.FilterByName("Department A");
```

FirstFilter

Method of VcFilterCollection

This method can be used to access the initial value, i.e. the first filter of a filter collection, and then to continue in a forward iteration loop by the method **NextFilter** for the filters following. If there is no filter in the FilterCollection object, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcFilter	First filter

Example Code VB.NET

```
Dim filterCltn As VcFilterCollection
Dim filter As VcFilter

filterCltn = VcGantt1.FilterCollection
filter = filtercltn.FirstFilter
```

Example Code C#

```
VcFilterCollection filterCltn = vcGantt1.FilterCollection;
VcFilter filter = filterCltn.FirstFilter();
```

GetEnumerator**Method of VcFilterCollection**

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the filter objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

Example Code VB.NET

```
Dim filter As VcFilter
Dim filterCond As VcFilterSubCondition

filter = VcGantt1.FilterCollection.FirstFilter

For Each filterCond In filter
    Debug.Write(filterCond.FilterName)
Next
```

Example Code C#

```
VcFilter filter = vcGantt1.FilterCollection.FirstFilter();
foreach(VcFilterSubCondition filterCond in filter)
{
    Console.Write(filterCond.FilterName);
}
```

NextFilter**Method of VcFilterCollection**

This method can be used in a forward iteration loop to retrieve subsequent filters from a curve collection after initializing the loop by the method

FirstFilter. If there is no filter left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcFilter	Next filter

Example Code VB.NET

```
Dim filterCltn As VcFilterCollection
Dim filter As VcFilter

filterCltn = VcGantt1.FilterCollection
filter = filterCltn.FirstFilter

While Not filter Is Nothing
    ListBox1.Items.Add(filter.Name)
    filter = filterCltn.NextFilter
End While
```

Example Code C#

```
VcFilterCollection filterCltn = vcGantt1.FilterCollection;
VcFilter filter = filterCltn.FirstFilter();

while (filter != null)
{
    ListBox.Items.Add(filter.Name);
    filter = filterCltn.NextFilter();
}
```

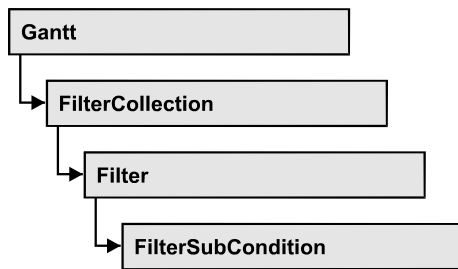
Remove

Method of VcFilterCollection

This method lets you delete a filter. If the filter is used in another object, it cannot be deleted. Then False will be returned, otherwise True.

	Data Type	Explanation
Parameter: ⇒ name	System.String	Filter name
Return value	System.Boolean	Filter deleted (True)/not deleted (False)

7.31 VcFilterSubCondition



An object of the type VcFilterSubCondition contains a single filter subcondition. It does not have a name, but only an index that specifies its position in the filter.

In the **Edit Filter** dialog each line corresponds to a subcondition. The properties specified at design time in that dialog can be modified via the API at runtime.

Properties

- ComparisonValueAsString
- ConnectionOperator
- DataFieldIndex
- FilterName
- Index
- Operator

Methods

- GetEnumerator
- IsValid

Properties

ComparisonValueAsString

Property of VcFilterSubCondition

This property lets you set or retrieve the comparison value. This string must have the below format:

- String: needs to be included by double quotation marks. Example in VB: `""Berlin""`; Example in C/C++: `"Berlin"`

- Date: included by # signs. Example: "#18/06/2015;12:34;56;#". A special date comparison value is "<TODAY>".
- Date field: included by square brackets. Example: "[ID]"
- Number: entered directly. Example: "52076"
- List: for a vc...In operator: included by {} brackets. All values included must have the same type (string, date or number). They may have one of the formats mentioned above. Example: "{"NETRONIC", [Name]}"
- Invalid (e.g. after creating a subcondition): "<INVALID>"

The type of the comparison value has to match the type of the data field and the operator type.

	Data Type	Explanation
Property value	System.String	Comparison value

ConnectionOperator

Property of VcFilterSubCondition

This property lets you set or retrieve the operator that connects the subsequent subcondition. Among the operators **vcAnd** is stronger than **vcOr**.

	Data Type	Explanation
Property value	VcConnectionOperator	Operator for the connection holding the below subcondition
	Possible Values: .vcAnd 1 .vcInvalidConnOp 0 .vcOr 2	And operator invalid operator Or operator

DataFieldIndex

Property of VcFilterSubCondition

This property lets you set or retrieve the index of the data field the content of which is to be compared. The data field type has to match the type of the comparison value and the operator.

Special values:

- -1: no data field (invalid)
- vcBarGroupLevel: variable for the group level number
- vcGroupCollapsed: entry for collapsed groups
- vcGroupNodeOrSummaryNode: entry for summary bars
- vcNodesInSeparateRows: entry for displaying all nodes in separate rows
- vcNodesOverlaid: entry for displaying nodes overlaid, if necessary
- vcRowNumber: entry to define filters for special rows
- vcSumBarLevel: variable for the level number of the summary bar

This property can also be set in the **Edit filter** dialog.

	Data Type	Explanation
Property value	System.Int32	Index of the data field to be compared

FilterName

Read Only Property of VcFilterSubCondition

This property lets you retrieve the name of the filter to which this subcondition belongs.

	Data Type	Explanation
Property value	System.String	Name of the filter

Index

Read Only Property of VcFilterSubCondition

This property lets you retrieve the index of this subcondition in the corresponding filter.

	Data Type	Explanation
Property value	System.Int16	Index of the subcondition in the filter

Operator

Property of VcFilterSubCondition

This property lets you set or retrieve the comparison operator. The operators that are available in the API correspond to the operators in the **Edit Filter** dialog. The operator type has to match the types of the data field and of the comparison value.

	Data Type	Explanation
Property value	VcOperator	Comparison operator
	Possible Values: .vcDateEarlier 27 .vcDateEarlierOrEqual 28 .vcDateEqual 25 .vcDateIn 31 .vcDateLater 29 .vcDateLaterOrEqual 30 .vcDateNotEqual 26 .vcDateNotIn 32 .vcIntEqual 9 .vcIntGreater 13 .vcIntGreaterOrEqual 14 .vcIntIn 15 .vcIntLess 11 .vcIntLessOrEqual 12 .vcIntNotEqual 10 .vcIntNotIn 16 .vcInvalidOp 0 .vcStringBeginsWith 3 .vcStringContains 5 .vcStringEqual 1 .vcStringIn 7 .vcStringNotBeginsWith 4 .vcStringNotContains 6 .vcStringNotEqual 2 .vcStringNotIn 8	Date earlier than Date earlier than or equal Date equal Date in Date later than Date later than or equal Date not equal Date not in integer equal integer greater integer greater or equal integer in integer smaller than integer smaller than or equal integer not equal integer not in invalid operator string begins with string contains string equal string contains string does not begin with string does not contain string is not equal string is not in

Methods

GetEnumerator

Method of VcFilterSubCondition

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the fields included in the filter subcondition object.

	Data Type	Explanation
Return value	VcObject	Reference object

IsValid

Method of VcFilterSubCondition

This property checks whether the filter subcondition is correct.

	Data Type	Explanation
Return value	System.Boolean	Filter subcondition correct (True)/ not correct (False)

7.32 VcGantt

Gantt

A VcGantt object is the VARCHART XGantt control. You use events to control interactions with the VcGantt object. It can be customized by a number of properties and methods to meet your demands.

Properties

- ActiveNodeFilter
- AllLayersMovingTogether
- AllLayersMovingTogetherAlways
- Arrangement
- ArrowKeyMode
- ArrowKeyStepSizeMultiplier
- BorderArea
- BoxCollection
- BoxCreationAllowed
- BoxFormatCollection
- CalendarCollection
- CalendarGridCollection
- CalendarProfileCollection
- ConsiderLinkRelationTypesOnNodeDragging
- ContextMenuForBoxesEnabled
- CtrlCXVProcessingEnabled
- DataDefinition
- DataTableCollection
- DateLineCollection
- DateLineCollection
- DateOutputFormat
- DiagramAlternatingRowBackgroundColor
- DiagramBackgroundColor
- DiagramHistogramHeightRatio
- DiagramHistogramHeightRatioEx
- DiagramVisible
- DialogFont
- DirectDataWritingModeEnabled
- DoubleOutputFormat
- Enabled
- EndDateForAutomaticScheduling

- EventsSecurityCheck
- ExtendedDataTablesEnabled
- ExtendedEditingBehavior
- FilePath
- FilterCollection
- FontAntiAliasingEnabled
- GroupCollection
- GroupingDataFieldIndex
- GroupingModificationsAllowed
- GroupLevelLayoutCollection
- GroupOptimizationOnInteractionsEnabled
- GroupSortingDataFieldIndex
- GroupSortingOrder
- HierarchyDataFieldIndex
- HierarchyLevelLayout
- HistogramCollection
- HistogramSeparationLineColor
- HorizontalMovementWhileDraggingAllowed
- InbuiltMouseCursorWhileDraggingEnabled
- InfoWindow
- InitialRowCount
- InPlaceEditingOnGroupsInDiagramEnabled
- InPlaceEditingOnGroupsInTableEnabled
- InPlaceEditingOnNodesInDiagramEnabled
- InPlaceEditingOnNodesInTableEnabled
- InteractionMode
- KeepingNodesTogetherDataFieldIndex
- LayerCollection
- LayersWithNonWorkInterval
- LeavingControlWhileDraggingAllowed
- LeftTable
- LeftTableDiagramWidthRatio
- LeftTableDiagramWidthRatioEx
- LegendView
- LineFormatCollection
- LinkAppearanceCollection
- LinkCollection
- LinkPredecessorDataFieldIndex
- LinksDataTableName
- LinkSuccessorDataFieldIndex

- LinkTypeDataFieldIndex
- MapCollection
- MinimumRowHeight
- MouseProcessingEnabled
- MoveMode
- MovingLayersAsNodeWithShiftKeyAllowed
- MultipleBoxMarkingAllowed
- NodeCalendarNameDataFieldIndex
- NodeCollection
- NodeCreationAllowed
- NodeCreationAtDroppingEnabled
- NodeCreationViaDoubleclick
- NodeCreationWithDialog
- NodeDurationDataFieldIndex
- NodeEndDateDataFieldIndex
- NodeLevelLayout
- NodeRowNumberDataFieldIndex
- NodesDataTableName
- NodeSortingDataFieldIndex
- NodeSortingOrder
- NodeStartDateDataFieldIndex
- NodesUseCalendars
- NodeToolTipTextDataFieldIndex
- NumericScaleCollection
- NumericScaleRescalingAllowed
- OLEDragViaDiagram
- OLEDragViaTable
- OverlapLayerEnabled
- OverlapLayerName
- PanningModeAllowed
- PartialLoadThreshold
- PhantomDrawingWhileDraggingEnabled
- PhantomLayerHeight
- Printer
- ResourceScheduler2
- RightTable
- RightTableDiagramWidthRatio
- RightTableDiagramWidthRatioEx
- RoundedLinkSlantsEnabled
- RowHeightReductionEnabled

- RowMargins
- Sash3DStyleEnabled
- SashThickness
- Scheduler
- ScrollEventsEnabled
- SelectedNodesMovingTogether
- SelectedRowBackgroundColor
- SelectionViaRubberRectAllowed
- ShowSnapLines
- ShowSnapMarkings
- SnapTargetNodesSelectionMode
- StartDateForAutomaticScheduling
- SubRowMargins
- SummaryBarsVisible
- TableCollection
- TableColumnWidthOptimizationAllowed
- TextEntrySupplyingEventEnabled
- TimeScaleCollection
- TimeScaleDialogEnabled
- TimeScaleEnd
- TimeScaleRescalingAllowed
- TimeScaleStart
- TimeUnit
- TimeUnitsPerStep
- ToolTipChangeDuration
- ToolTipDuration
- ToolTipPointerDuration
- ToolTipShowAfterClick
- ToolTipTextSupplyingEventEnabled
- TrackingSpaceBackgroundColor
- TrackingSpacePattern
- TrackingSpacePatternColor
- UpdateBehaviorCollection
- UseHigherDiagramHistogramHeightRatioPrecision
- UseHigherTableDiagramWidthRatioPrecision
- UseSnapTargetsInInteractions
- UseTwinLineSashPhantom
- VerticalNodeMovementAllowed
- VerticalNodeMovementViaTableAllowed
- ViewComponentsBackgroundColor

- ViewComponentsBorderColor
- WaitCursorEnabled
- WorldView
- ZoomFactor
- ZoomingPerMouseWheelAllowed

Methods

- ConvertDistance
- DeleteLinkRecord
- DeleteNodeRecord
- DetectDataTableFieldName
- DetectDataTableName
- DetectFieldIndex
- DumpConfiguration
- EndLoading
- ExportGraphicsToFileEx
- FitChartIntoView
- FitHistogramsIntoView
- FitRangeIntoView
- GetAValueFromARGB
- GetBValueFromARGB
- GetCurrentComponentStart
- GetCurrentViewDates
- GetDate
- GetDateAsString
- GetGValueFromARGB
- GetLinkByID
- GetLinkByNodeIDs
- GetNodeByID
- GetRValueFromARGB
- GetViewComponentSize
- GroupNodes
- IdentifyField
- IdentifyLayerAt
- IdentifyObject
- IdentifyObjectAt
- ImportConfiguration
- InitializeForWebService
- InsertLinkRecord
- InsertNodeRecord

- Load
- MakeARGB
- OptimizeTimeScaleStartEnd
- PrintEx
- PrintToFile
- RecalculateAllStructureCodes
- Reset
- SaveAsEx
- ScheduleProject
- ScrollComponentStartTo
- ScrollToDate
- ScrollToGroupLine
- ScrollToNode
- ScrollToNodeLine
- SetImageResource
- ShowAboutDialog
- ShowEditGroupDialog
- ShowExportGraphicsDialog
- ShowLinkEditDialog
- ShowNodeEditDialog
- ShowPageSetupDialog
- ShowPrintDialog
- ShowPrinterSetupDialog
- ShowPrintPreviewDialog
- SortGroups
- SortNodes
- SuspendUpdate
- UpdateLinkRecord
- UpdateNodeRecord
- UpdateRowNumberFields
- Zoom

Events

- KeyDown
- KeyPress
- KeyUp
- VcBoxCreated
- VcBoxCreating
- VcBoxLeftClicking
- VcBoxLeftDoubleClicking

- VcBoxModified
- VcBoxModifying
- VcBoxRightClicking
- VcCalendarGridRightClicking
- VcComponentScrolled
- VcComponentScrolling
- VcCurveLeftClicking
- VcCurveLeftDoubleClicking
- VcCurveModified
- VcCurveModifying
- VcCurveModifyingEx
- VcCurvePointDeleting
- VcCurvePointDeletingEx
- VcCurvePointInserting
- VcCurvePointInsertingEx
- VcCurveRightClicking
- VcDataModified
- VcDataRecordCreated
- VcDataRecordCreating
- VcDataRecordDeleted
- VcDataRecordDeleting
- VcDataRecordModified
- VcDataRecordModifying
- VcDataRecordNotFound
- VcDateLineModifying
- VcDateLineRightClicking
- VcDateShowing
- VcDiagramHorizontalScrolled
- VcDiagramHorizontalScrolling
- VcDiagramLeftClicking
- VcDiagramLeftDoubleClicking
- VcDiagramRightClicking
- VcDragCompleting
- VcDragOver
- VcDragStarting
- VcErrorOccurring
- VcFieldSelecting
- VcGroupDeleting
- VcGroupLeftClicking
- VcGroupLeftDoubleClicking

- VcGroupModified
- VcGroupModifying
- VcGroupRightClicking
- VcGroupsMarked
- VcGroupsMarking
- VcHelpRequested
- VcHistogramCurveNameShowingInMenu
- VcHistogramLeftClicking
- VcHistogramLeftDoubleClicking
- VcHistogramRightClicking
- VcHistogramsHeightChanged
- VcHistogramsHeightChanging
- VcHistogramsHeightChangingEx
- VcInPlaceEditorShowing
- VcInteractionEnded
- VcInteractionModeChanged
- VcInteractionModeChanging
- VcInteractionObjectChanged
- VcInteractionStarted
- VcLegendViewClosed
- VcLinkCreated
- VcLinkCreating
- VcLinkDeleted
- VcLinkDeleting
- VcLinksLeftClicking
- VcLinksLeftDoubleClicking
- VcLinksRightClicking
- VcNodeCreated
- VcNodeCreating
- VcNodeDeleted
- VcNodeDeleting
- VcNodeLeftClicking
- VcNodeLeftDoubleClicking
- VcNodeModified
- VcNodeModifiedEx
- VcNodeModifying
- VcNodeResizeStarting
- VcNodeRightClicking
- VcNodesMarked
- VcNodesMarking

- VcNumericScaleLeftClicking
- VcNumericScaleLeftDoubleClicking
- VcNumericScaleRescaling
- VcNumericScaleRightClicking
- VcObjectDrawing
- VcObjectDrawn
- VcResourceSchedulingProgressing
- VcResourceSchedulingWarning
- VcSashButtonClicked
- VcStatusLineTextShowing
- VcTableCaptionLeftClicking
- VcTableCaptionLeftDoubleClicking
- VcTableCaptionRightClicking
- VcTableColumnWidthChanged
- VcTableColumnWidthChanging
- VcTableColumnWidthOptimizing
- VcTableWidthChanging
- VcTableWidthChangingEx
- VcTextEntrySupplying
- VcTimeScaleEndModified
- VcTimeScaleLeftClicking
- VcTimeScaleLeftDoubleClicking
- VcTimeScaleModified
- VcTimeScaleRightClicking
- VcTimeScaleSectionRescaled
- VcTimeScaleSectionRescaledEx
- VcTimeScaleSectionRescaling
- VcTimeScaleSectionRescalingEx
- VcTimeScaleSectionStartModifying
- VcTimeScaleStartModified
- VcToolTipTextSupplying
- VcViewComponentsSizeModified
- VcWorldViewClosed
- VcZoomFactorModified

Properties

ActiveNodeFilter

Property of VcGantt

This property lets you set or retrieve a filter that selects the nodes to be displayed.

	Data Type	Explanation
Property value	VcFilter	Filter object Default value: Nothing

Example Code VB.NET

```
VcGantt1.ActiveNodeFilter = VcGantt1.FilterCollection.FilterByName("Milestone")
```

Example Code C#

```
vcGantt1.ActiveNodeFilter = vcGantt1.FilterCollection.FilterByName("Milestone");
```

AllLayersMovingTogether

Property of VcGantt

This property lets you set or retrieve whether a marked node can be interactively moved as a whole (True). Otherwise single layers can be moved only (False). This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Boolean	A marked node can be interactively moved as a whole (True)/only single layers can be moved (False)

Example Code VB.NET

```
VcGantt1.AllLayersMovingTogether = True
```

Example Code C#

```
vcGantt1.AllLayersMovingTogether = true;
```

AllLayersMovingTogetherAlways

Property of VcGantt

This property lets you set or retrieve whether all layers of a node can be moved at the same time without having to be marked before. This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Boolean	Dragging of nodes as a whole without marking them before is switched on (true) or off (false) Default value: false

Example Code VB.NET

```
VcGantt1.AllLayersMovingTogetherAlways = True
```

Example Code C#

```
vcGantt1.AllLayersMovingTogetherAlways = true;
```

Arrangement

Property of VcGantt

By this property you can set or retrieve whether the activities are arranged in a hierarchy or in groups. You can also set this property on the **Sorting** property page, by ticking the check box **Hierarchy**. This property is only effective if the property **HierarchyDataFieldIndex** or **GroupDataFieldIndex** was set, respectively.

	Data Type	Explanation
Property value	VcArrangementType	Arrangement of activities groupwise or hierarchical Default value: vcArrangementTypeGroupwise
	Possible Values: .vcArrangementTypeGroupwise 1 .vcArrangementTypeHierarchical 2	Groupwise Arrangement of activities Hierarchical Arrangement of activities

Example Code VB.NET

```
VcGantt1.GroupLevelLayoutCollection.FirstGroupLevelLayout().GroupDataFieldIndex
= VcGantt1.DetectFieldIndex("Maindata", "Department")
VcGantt1.Arrangement = VcArrangementType.vcArrangementTypeGroupwise
VcGantt1.GroupNodes(True)

// alternativ:

VcGantt1.HierarchyDataFieldIndex = VcGantt1.DetectFieldIndex("Maindata",
"StructureCode")
VcGantt1.Arrangement = VcArrangementType.vcArrangementTypeHierarchical
VcGantt1.GroupNodes(True)
```

Example Code C#

```
vcGantt1.GroupLevelLayoutCollection.FirstGroupLevelLayout().GroupDataFieldIndex
= vcGantt1.DetectFieldIndex("Maindata", "Department");
vcGantt1.Arrangement = VcArrangementType.vcArrangementTypeGroupwise;
vcGantt1.GroupNodes(true);

// alternativ:

vcGantt1.HierarchyDataFieldIndex = vcGantt1.DetectFieldIndex("Maindata",
"StructureCode");
vcGantt1.Arrangement = VcArrangementType.vcArrangementTypeHierarchical;
vcGantt1.GroupNodes(true);
```

ArrowKeyMode

Property of VcGantt

By this property you can set the mode of the <left> and <right> arrow keys. Usually, the arrow keys are reserved for various interactions, such as scrolling the diagram, moving a marked field within a node or within the table. These navigating functions you can change by this property into modifying functions, so the user can move, enlarge or reduce the size of a node by them. A window displaying information on the position will remain on the screen for a few more seconds after the interaction finished to let the user read its content.

If the node being moved arrives at a border of the view, the diagram will automatically start scrolling (autoscroll).

By simply striking the arrow keys, a node will move; if the user in addition presses the <Shift> key, he or she can change the size of the node.

Assigning modifying functions to the arrow keys is very useful in low-resolution charts, since moving or resizing a node by mouse may be imprecise. Positioning a node by arrow keys is more precise, because each single step of the motion is indicated in the information window, representing a much higher resolution than is offered by the time scale. The step size is

controlled by the properties **VcGantt.TimeUnit**, **VcGantt.TimeUnitsPerStep** and **VcGantt.ArrowKeyStepSizeMultiplier**.

	Data Type	Explanation
Property value	VcArrowKeyMode	Mode of the <left> and <right> arrow keys
	Possible Values:	
	.vcnodeJumpToSnapTarget 512	Pressing <CTRL> + arrow keys <left> or <right> a marked node is moved to previous/next snap target.
	.vcResizeOrMoveNode 384	The arrow keys <left> and <right> are in the mode to modify nodes
	.vcStandard 127	The arrow keys <left> and <right> are in their default mode

Example Code VB.NET

```
'Assigning the function to an option button
Private Sub OptionEditNode_Click()
    If OptionStandard.Value = True Then
        VcGantt1.ArrowKeyMode = vcStandard
    Else
        VcGantt1.ArrowKeyMode = vcResizeOrMoveNode
    End If
End Sub
```

Example Code C#

```
//Assigning the function to an option button
private void OptionEditNode_Click(object sender, EventArgs e)
{
    if (OptionEditNode.Checked)
        vcGantt1.ArrowKeyMode = VcArrowKeyMode.vcStandard;
    else
        vcGantt1.ArrowKeyMode = VcArrowKeyMode.vcResizeOrMoveNode;
}
```

ArrowKeyStepSizeMultiplier

Property of VcGantt

This property lets you set or retrieve the value of the arrow keys step size multiplier. When moving the cursor by mouse or by arrow keys (see property **VcGantt.ArrowKeyMode**), the properties **VcGantt.TimeUnit** and **VcGantt.TimeUnitsPerStep** will determine the step size, multiplying their values. If for example the time unit was set to a day and the units per step were set to 2, the step size will be 2 days. Since by the mouse farther motion can be obtained simply by continued dragging, but keys do not offer anything comparable, this additional multiplier exists for the arrow keys. The user can activate it by pressing the <Ctrl> key in addition to the arrow key. If you set the value of the multiplier to 10, the step size in the above example will be 20 days per key stroke.

714 API Reference: VcGantt

	Data Type	Explanation
Property value	System.Int16	Value of the multiplier

Example Code VB.NET

```
'Reducing the time scale resolution and enlarging the step size
Private Sub CommandExtendScale_Click()
    'Filling up the available space for the Gantt graph by extending the time
scale
    VcGantt1.TimeScaleEnd = DateSerial(2017, 1, 1)
    ' Reducing the resolution of the time scale by the factor 10
    VcGantt1.TimeScaleCollection.Active.Section(0).UnitWidth =
VcGantt1.TimeScaleCollection.Active.Section(0).UnitWidth / 10
    ' Increasing the multiplier for the arrow keys to move in larger steps
    VcGantt1.ArrowKeyStepSizeMultiplier = 25
End Sub
```

Example Code C#

```
'Reducing the time scale resolution and enlarging the step size
private void CommandExtendScale_Click(object sender, EventArgs e)
{
    //Filling up the available space for the Gantt graph by extending the time
scale
    vcGantt1.TimeScaleEnd = Convert.ToDateTime("01.01.2017");
    //Reducing the resolution of the time scale by the factor 10
    vcGantt1.TimeScaleCollection.Active.get_Section(0).UnitWidth =
vcGantt1.TimeScaleCollection.Active.get_Section(0).UnitWidth / 10;
    //Increasing the multiplier for the arrow keys to move in larger steps
    vcGantt1.ArrowKeyStepSizeMultiplier = 25;
}
```

BorderArea

Read Only Property of VcGantt

This property gives access to the BorderArea object, i. e. the title and legend area.

	Data Type	Explanation
Property value	VcBorderArea	Title and legend area

Example Code VB.NET

```
Dim borderArea As VcBorderArea
borderArea = VcGantt1.BorderArea
```

Example Code C#

```
VcBorderArea borderArea = vcGantt1.BorderArea;
```

BoxCollection

Read Only Property of VcGantt

This property gives access to the BoxCollection object that contains all boxes available.

	Data Type	Explanation
Property value	VcBoxCollection	BoxCollection object

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
boxCltn = VcGantt1.BoxCollection
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
```

BoxCreationAllowed

Property of VcGantt

This property permits (True) or prohibits (False) the user to create new boxes. If this property is set to **False**, the user cannot activate the **Mode: Create box** and it is not possible to set the **InteractionMode** to **VcCreateBox**. This property also can be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Property active (True)/ not active (False)

Example Code VB.NET

```
VcGantt1.BoxCreationAllowed = False
```

Example Code C#

```
vcGantt1.BoxCreationAllowed = false;
```

BoxFormatCollection

Read Only Property of VcGantt

This property gives access to the BoxFormatCollection object that contains all box formats available to the table.

	Data Type	Explanation
Property value	VcBoxFormatCollection	BoxFormatCollection object

Example Code VB.NET

```
Dim boxFormatCltn As VcBoxFormatCollection
Dim boxFormat As VcBoxFormat

boxFormatCltn = VcGantt1.BoxFormatCollection
For Each boxFormat In boxFormatCltn
    ListBox1.Items.Add(boxFormat.Name)
Next
```

Example Code C#

```
VcBoxFormatCollection boxFormatCltn = vcGantt1.BoxFormatCollection;
foreach (VcBoxFormat boxFormat in boxFormatCltn)
{
    listBox1.Items.Add(boxFormat.Name);
}
```

CalendarCollection

Read Only Property of VcGantt

This property gives access to the calendar collection object that contains all calendars available.

	Data Type	Explanation
Property value	VcCalendarCollection	CalendarCollection object

Example Code VB.NET

```
Dim calendarCltn As VcCalendarCollection
calendarCltn = VcGantt1.CalendarCollection
```

Example Code C#

```
VcCalendarCollection calendarCltn = vcGantt1.CalendarCollection;
```

CalendarGridCollection

Read Only Property of VcGantt

This property gives access to the calendar grid collection object and thus to the calendar grids used.

	Data Type	Explanation
Property value	VcCalendarGridCollection	CalendarGridCollection object

Example Code VB.NET

```
Dim calendarGridCltn As VcCalendarGridCollection
calendarGridCltn = VcGantt1.CalendarGridCollection
```

Example Code C#

```
VcCalendarGridCollection calendarGridCltn = vcGantt1.CalendarGridCollection;
```

CalendarProfileCollection

Read Only Property of VcGantt

This property gives access to the CalendarProfileCollection object that contains all calendar profiles available.

	Data Type	Explanation
Property value	VcCalendarProfileCollection	CalendarProfileCollection object

ConsiderLinkRelationTypesOnNodeDragging

Property of VcGantt

When this property is set to **True**, the phantom lines representing the links will be displayed indicating their type if dragged, and if links are switched on at all. The phantom lines will not start off from the center of the node, but from the left and right side of the node.

This property can also be set on the **General** property page.

	Data Type	Explanation

Example Code VB.NET

```
VcGantt1.ConsiderLinkRelationTypesOnNodeDragging = True
```

ContextMenuForBoxesEnabled

Property of VcGantt

By this property you can set or retrieve whether the context menu for boxes is enabled. This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Context menu for box is/is not enabled

Example Code VB.NET

```
VcGantt1.ContextMenuForBoxesEnabled = True
```

Example Code C#

```
vcGantt1.ContextMenuForBoxesEnabled = true;
```

CtrlCXVProcessingEnabled

Property of VcGantt

This property automatically translates the key combinations <Ctrl>+<C>, <Ctrl>+<X> and <Ctrl>+<V> into the clipboard commands **CopyNodesToClipboard**, **CutNodesToClipboard** and **PasteNodesFromClipboard**, respectively. You can suppress this feature by setting the property to **False**, in order to avoid conflicts with menu commands in Visual Basic. This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Key combinations will/will not be translated into clipboard commands Default value: True

Example Code VB.NET

```
VcGantt1.CtrlCXVProcessingEnabled = False
```

Example Code C#

```
vcGantt1.CtrlCXVProcessingEnabled = false;
```

DataDefinition

Read Only Property of VcGantt

This property gives access to the current data definition object, in order to e.g. enquire field names or field types. The data definition of VcGantt has got two data definition tables: vcMaindata and vcRelations.

	Data Type	Explanation
Property value	VcDataDefinition	Data definition

Example Code VB.NET

```
Dim dataDefinition As VcDataDefinition  
dataDefinition = VcGantt1.DataDefinition
```

Example Code C#

```
VcDataDefinition dataDefinition = vcGantt1.DataDefinition;
```

DataTableCollection

Property of VcGantt

This property gives access to the data table collection that contains the existing data tables.

	Data Type	Explanation
Property value	VcDataTableCollection	Data table collection object returned

Example Code VB.NET

```
Dim dataTableCltn As VcDataTableCollection
Dim dataTable As VcDataTable

dataTableCltn = VcGantt1.DataTableCollection
For Each dataTable In dataTableCltn
    ListBox1.Items.Add(dataTable.Name)
Next
```

Example Code C#

```
VcDataTableCollection dataTableCltn = vcGantt1.DataTableCollection;
foreach(VcDataTable dataTable in dataTableCltn)
    listBox1.Items.Add(dataTable.Name);
```

DateLineCollection

Read Only Property of VcGantt

This property gives access to the DateLineCollection object which contains all date lines available.

	Data Type	Explanation
Property value	VcDateLineCollection	DateLineCollection object

Example Code VB.NET

```
Dim dateLineCltn As VcDateLineCollection
dateLineCltn = VcGantt1.DateLineCollection
```

Example Code C#

```
VcDateLineCollection dateLineCltn = vcGantt1.DateLineCollection;
```

DateLineCollection

Read Only Property of VcGantt

This property gives access to the DateLineCollection object which contains all date lines available.

	Data Type	Explanation
Property value	VcDateLineCollection	DateLineCollection object

DateOutputFormat

Property of VcGantt

This property lets you set or retrieve the date output format. To compose the date you can use the below codes:

- D: first letter of the day of the week (not adjustable)
- TD: Day of the Week (adjustable by using the event **VcTextEntrySupplying**)
- DD: two-digit figure for the day of the month: 01-31
- DDD: first three letters of the day of the week (not adjustable)
- M: first letter of the name of the month (not adjustable)
- TM: name of the month (adjustable by using the event **VcTextEntrySupplying**)
- MM: two-digit figure for the month: 01-12
- MMM: first three letters of the name of the month (not adjustable)
- YY: two-digit figure for the year
- YYYY: four-digit figure for the year
- WW: two-digit figure for the number of the calendar week: 01-53
- TW: text for "calendar week" (adjustable by using the event **VcTextEntrySupplying**)
- Q: one-digit figure for the quarter: 1-4
- TQ: name of quarter (adjustable by using the event **VcTextEntrySupplying**)
- hh two-digit figure for the hour in 24 hours format: 00-23
- HH: two-digit figure for the hour in 12 hours format: 01-12
- Th: Text of "o' clock" (adjustable by using the event **VcTextEntrySupplying**)
- TH: "am" or "pm" (adjustable by using the event **VcTextEntrySupplying**)

mm	two-digit figure for the minute: 00-59
ss:	two-digit figure for the second: 00-59
TS:	short date format, as defined in the regional settings of the windows control panel
TL:	long date format, as defined in the regional settings of the windows control panel
TT:	time format, as defined in the regional settings of the windows control panel

Note: Characters which are not to be interpreted as part of the date should be preceded by a backslash '\'. '\\' for instance results in '\'. The special characters: ':, /, -' and **blank** don't need '\' as prefix.

This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.String	Date {DMYhms:;}

Example Code VB.NET

```
VcGantt1.DateOutputFormat = "DD.MM.YY"
```

Example Code C#

```
vcGantt1.DateOutputFormat = "DD.MM.YY";
```

DiagramAlternatingRowBackgroundColor

Read Only Property of VcGantt

This property lets you set or retrieve a second background color to the diagram, which forms a linewise alternating pattern with the color set by the property **DiagramBackgroundColor**. This property also can be set on the **Layout** property page.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {(0...255),(0...255),(0...255)} Default value: System.Drawing.Color.White

Example Code VB.NET

```
VcGantt1.DiagramAlternatingRowBackgroundColor = System.Drawing.Color.Blue
```

Example Code C#

```
vcGantt1.DiagramAlternatingRowBackgroundColor = System.Drawing.Color.Blue;
```

DiagramBackgroundColor

Property of VcGantt

This property lets you set or retrieve the diagram background color. If you combine this property with the property **DiagramAlternatingRowBackColor** you can generate a color pattern that alternates linewise. This property also can be set on the **Layout** property page.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {{0...255},{0...255},{0...255}} Default value: SystemDrawing.Color.White

Example Code VB.NET

```
VcGantt1.DiagramBackgroundColor = System.Drawing.Color.Blue
```

Example Code C#

```
vcGantt1.DiagramBackgroundColor = System.Drawing.Color.Blue;
```

DiagramHistogramHeightRatio

Property of VcGantt

By this property you can set or retrieve ratio (in %) of the height of the diagram area (without histogram) to the height of the histogram at the start of the program. If the ratio is -1 or 0, the histogram will be displayed completely at the start. This property also can be set on the **Layout** property page.

	Data Type	Explanation
Property value	Integer {-1, 0, 1, ..., 1000}	Ratio between diagram height and histogram height

Example Code VB.NET

```
Dim ratio As Integer
ratio = VcGantt1.DiagramHistogramHeightRatio
```

Example Code C#

```
int ratio = vcGantt1.DiagramHistogramHeightRatio;
```

DiagramHistogramHeightRatioEx

Property of VcGantt

This property lets you set or retrieve the ratio between the total height of the diagram (in %) and the height of the histogram. In contrast to the **DiagramHistogramHeightRatio** property this property returns a "Double" value, thus achieving a higher level of accuracy. The use of this property has to be enabled by the **UseHigherDiagramHistogramHeightRatioPrecision** property or by activating the corresponding option on the **General** property page.

	Data Type	Explanation
Property value	System.Double	Height ratio

Example Code VB.NET

```
VcGantt1.DiagramHistogramHeightRatioEx = 40
```

Example Code C#

```
vcGantt1.DiagramHistogramHeightRatioEx = 40;
```

DiagramVisible

Property of VcGantt

This property lets you set or retrieve whether the diagram section (table and Gantt graph) should be visible. This property also can be set on the **Layout** property page.

	Data Type	Explanation
Property value	System.Boolean	Diagram section visible (True) / invisible (False) Default value: False

Example Code VB.NET

```
VcGantt1.DiagramVisible = False
```

Example Code C#

```
vcGantt1.DiagramVisible = true;
```

DialogFont

Property of VcGantt

This property specifies/retrieves the font name and font size in the dialogs of the VARCHART XGantt control that appear at run time. The object expected

is a font object of your programming environment, e.g. in Visual Basic an object of the class **Stdfont**.

	Data Type	Explanation
Property value	System.DrawingFont	Font attributes

Example Code VB.NET

```
Dim newFont As Font
newFont = New Font("Verdana", 14)
VcGantt1.DialogFont = newFont
```

Example Code C#

```
Font newFont = new Font("Verdana", 14);
vcGantt1.DialogFont = newFont;
```

DirectDataWritingModeEnabled

Property of VcGantt

If this property is set to "True", data modifications that are carried out by using **VcNode/VcLink/VcDataRecord/.set_DataField** or **.AllData** are directly stored to the data pool WITHOUT being evaluated (e.g. by filter analysis, mapping etc.).

Thus a better performance is achieved.

	Data Type	Explanation
Property value	System.Boolean	Data modifications without analysis are (True)/are not (False) carried out Default value: False

DoubleOutputFormat

Property of VcGantt

This property lets you set or retrieve the output format of numbers as a double value in a Gantt diagram. The format is presented by the below characters:

- Text
- I
- D

plus the separators **comma** and **period**. **Text** represents a character string; **I** represents the figures in front of the decimal separator and **D** represents the figures after the decimal separator. The overall sequence is **Text I D Text**, where a comma and a period can be inserted in the places desired. As an example be the number -284901,3458. By the format **I,DDDD ppm** it will be output as **-284901,3458 ppm**. By the format **\$I,III.DD** it will be output as **\$-284,901.35**.

This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.String	Character string which describes the double format, for example "\$I,III.DD".

Example Code VB.NET

```
VcGantt1.DoubleOutputFormat = "I,DDDD ppm"
```

Example Code C#

```
vcGantt1.DoubleOutputFormat = "$I,III.DD";
```

Enabled

Property of VcGantt

This property lets you disable the VARCHART XGantt control so that it will not react to mouse or keyboard commands.

	Data Type	Explanation
Property value	System.Boolean	VARCHART control enabled/disabled

Example Code VB.NET

```
VcGantt1.Enabled = False
```

Example Code C#

```
vcGantt1.Enabled = false;
```

EndDateForAutomaticScheduling

Property of VcGantt

This property lets you set or retrieve the end value for autoscheduling of the current project (**Schedule** property page).

	Data Type	Explanation
Property value	System.DateTime	End date

EventsSecurityCheck

Property of VcGantt

This property lets you activate/deactivate the event security check. You also can set this property on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Event security check on/out

Example Code VB.NET

```
VcGantt1.EventsSecurityCheck = False
```

Example Code C#

```
vcGantt1.EventsSecurityCheck = false;
```

ExtendedDataTablesEnabled

Property of VcGantt

This property allows to choose between using merely two data tables (Maindata and Relations) and the advanced use of up to 90 data tables. The latter option is recommended. This property needs to be set at the beginning of your program, before data tables and data records are created.

This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	true: only two data tables (Maindata and Relations) false: up to 99 data tables Default value: false

Example Code VB.NET

```
VcGantt1.ExtendedDataTablesEnabled = True
```

Example Code C#

```
vcGantt1.ExtendedDataTablesEnabled = true;
```

ExtendedEditingBehavior

Property of VcGantt

This property lets you set or retrieve whether at run time it is possible to use the enhanced possibilities of editing the table contents or navigating. You can also set this property on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Extended Editing enabled/disabled Default value: False

Example Code VB.NET

```
VcGantt1.ExtendedEditingBehavior = True
```

Example Code C#

```
vcGantt1.ExtendedEditingBehavior = true;
```

FilePath

Property of VcGantt

This property lets you set the file path so that graphics files will be found in the directory specified, even if only a relative file name was specified. Otherwise the file will be searched in the current directory of the application and in the installation directory of the VARCHART XGantt control.

This property should be set when the application is started during the initializing procedure of the VARCHART XGantt control. We recommend to set the file path to the path of the application or to a subdirectory of the application. The advantage of this action is that the application can be stored in any directory.

	Data Type	Explanation
Property value	System.String	File path Default value: " "

Example Code VB.NET

```
Dim exeName As String
Dim exeDir As String
```

```
exeName = System.Environment.GetCommandLineArgs(0)
exeDir = System.IO.Path.GetDirectoryName(exeName)
VcGantt1.FilePath = exeDir + "\Bitmaps"
```

Example Code C#

```
String exeName = Environment.GetCommandLineArgs()[0];
vcGantt1.FilePath = System.IO.Path.GetDirectoryName(exeName) + @"..\Bitmaps";
```

FilterCollection**Read Only Property of VcGantt**

This property gives access to the FilterCollection object that contains all filters available.

	Data Type	Explanation
Property value	VcFilterCollection	FilterCollection object

Example Code VB.NET

```
Dim filterCltn As VcFilterCollection
filterCltn = VcGantt1.FilterCollection
```

Example Code C#

```
VcFilterCollection filterCltn = vcGantt1.FilterCollection;
```

FontAntiAliasingEnabled**Read Only Property of VcGantt**

This property lets you set or retrieve whether fonts can be anti-aliased with GDI+. If the legibility of certain fonts - in particular non- latin ones - changes for the worse, the property should be set to **False**.

The anti-aliasing with GDI+ has yet another effect: regardless of the selected zoom factor, texts keep their relative dimension so that the number of characters that fits in a table field will always be the same. If the option is switched off the settings of the operating system are applied instead (the settings can be found in the **Control Panel**, dialog box **Display**, Tab **Appearance: Effects**). Thus, if the option **Smooth edges** is switched on in the **Control Panel**, the texts might still be anti-aliased, notwithstanding the settings of the **General** property page. In this case, at some zoom levels more text could be visible than at others, since the native edge smoothing does not guarantee that the same relative dimension is always kept.

This property also can be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Characters will/will not be anti-aliased Default value: true

GroupCollection

Read Only Property of VcGantt

When grouping is present in a chart, this property gives access to the GroupCollection object that contains all groups present in the chart.

	Data Type	Explanation
Property value	VcGroupCollection	GroupCollection object

Example Code VB.NET

```
Dim groupCltn As VcGroupCollection
groupCltn = VcGantt1.GroupCollection
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
```

GroupingDataFieldIndex

Property of VcGantt

This property lets you set or retrieve the field in the data definition table that is to be used as criterion for the grouping on a certain level. The groups by default will be sorted in the order of reading the first activity of the group. The sorting order can be modified by the property **GroupSortingDataFieldIndex**.

This property also can be set in the **Grouping** property page.

The property **GroupingDataFieldIndex** is an Indexed Property, which in C# is addressed by the methods set_GroupingDataFieldIndex (groupingLevel, pvn) and get_GroupingDataFieldIndex (groupingLevel).

	Data Type	Explanation
Parameter: ⇒ groupingLevel	System.Int16	Grouping level (starting by 0)
Property value	System.Int32	Field ID of the data definition table

Example Code VB.NET

```
Dim definitionTable As VcDataDefinitionTable
definitionTable =
VcGantt1.DataDefinition.DefinitionTable(VcDataTableType.vcMaindata)
VcGantt1.GroupingDataFieldIndex(0) =
definitionTable.DataDefinitionFieldByName("Code 1").ID
VcGantt1.GroupNodes(True)
```

Example Code C#

```
VcDataDefinitionTable definitionTable =
vcGantt1.DataDefinition.get_DefinitionTable(VcDataTableType.vcMaindata);
vcGantt1.set_GroupingDataFieldIndex(0, "Code 1");
vcGantt1.GroupNodes(true);
```

GroupingModificationsAllowed**Property of VcGantt**

This property lets you specify whether the user can collapse expanded groups and vice versa. The user can collapse/expand groups by double-clicking on the group heading in the table section, by clicking on the minus or plus sign next to the group heading or by the context menu for groups. This property also can be set in the **Grouping** dialog.

The property GroupingModificationsAllowed is an Indexed Property, which in C# is addressed by the methods set_GroupingModificationsAllowed (groupingLevel, pvn) and get_GroupingModificationsAllowed (groupingLevel).

	Data Type	Explanation
Parameter: ⇒ groupingLevel	System.Int16	Grouping level
Property value	System.Boolean	Modifications allowed (True)/ not allowed (False)

Example Code VB.NET

```
VcGantt1.GroupingModificationsAllowed(0) = False
```

Example Code C#

```
vcGantt1.set_GroupingModificationsAllowed(0, false);
```

GroupLevelLayoutCollection**Read Only Property of VcGantt**

This property gives access to the GroupLevelLayoutCollection object which contains all group level layouts available.

	Data Type	Explanation
Property value	VcGroupLevelLayoutCollection	GroupLevelLayoutCollection object

GroupOptimizationOnInteractionsEnabled

Property of VcGantt

If this property is set to **true**, the nodes of the target group automatically are optimized on interactions such as creating nodes, moving nodes or modifying their start or end date, if they had been in the optimized state of display before. If this property is set to **false**, on the interactions mentioned the node will be placed at the cursor, if this doesn't cause nodes to overlap. If it does, the node will be placed with other nodes in the next line, if this doesn't cause overlaps. If it does, a new line will be created below the one where the cursor is and the node will be put there.

This property can also be set at design time on the **General** property page.

Also see the method **VcGroup.ReOptimizeNodes**.

	Data Type	Explanation
Property value	System.Boolean	The Optimized re-arrangement of nodes will (True) / will not (False) be performed on interaction

GroupSortingDataFieldIndex

Property of VcGantt

This property lets you specify what field of the data definition table is to be used for sorting the groups. By using **GroupSortingDataFieldIndex**, the groups will be sorted in ascending or descending alphabetical order by this field. This property also can be set in the **Grouping** dialog.

The property GroupSortingDataFieldIndex is an Indexed Property, which in C# is addressed by the methods set_GroupSortingDataFieldIndex (groupingLevel, pvn) and get_GroupSortingDataFieldIndex (groupingLevel).

	Data Type	Explanation
Parameter: ⇒ groupingLevel	System.Int16	Grouping level
Property value	System.Int32	Field index of the data definition table

Example Code VB.NET

```
VcGantt1.GroupSortingDataFieldIndex(0) = 12
VcGantt1.GroupSortingOrder(0) = VcNodeSortingOrder.vcDescending
VcGantt1.SortGroups()
```

Example Code C#

```
vcGantt1.set_GroupSortingDataFieldIndex(0, 12);
vcGantt1.set_GroupSortingOrder(0, VcNodeSortingOrder.vcDescending);
vcGantt1.SortGroups();
```

GroupSortingOrder

Property of VcGantt

This property lets you specify the sorting order of groups (ascending or descending). By the property **GroupSortingDataFieldIndex** you can specify the field by that the groups are sorted. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	VcNodeSortingOrder	Ascending or descending order Default value: vcAscending
	Possible Values: .vcAscending 1 .vcDescending 2	ascending order Descending order

Example Code VB.NET

```
VcGantt1.GroupSortingDataFieldIndex(0) = 12
VcGantt1.GroupSortingOrder(0) = VcNodeSortingOrder.vcAscending
VcGantt1.SortGroups()
```

Example Code C#

```
vcGantt1.set_GroupSortingDataFieldIndex(0, 12);
vcGantt1.set_GroupSortingOrder(0, VcNodeSortingOrder.vcAscending);
vcGantt1.SortGroups();
```

HierarchyDataFieldIndex

Property of VcGantt

This property lets you set or retrieve the index of the data field which defines the hierarchical order of activities. This can be done even **after** having loaded data already. The modifications only become effective after having set the arrangement of activities to **hierarchical** with the property **VcGantt.Arrangement** (**vcArrangementTypeHierarchical**) and having carried out an update with the method **VcGantt.GroupNodes**.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which defines the hierarchical order of activities

Example Code VB.NET

```
VcGantt1.HierarchyDataFieldIndex = VcGantt1.DetectFieldIndex("Maindata",
"Hierarchy")
VcGantt1.Arrangement = VcArrangementType.vcArrangementTypeHierarchical
VcGantt1.GroupNodes(True)
```

Example Code C#

```
vcGantt1.HierarchyDataFieldIndex = vcGantt1.DetectFieldIndex("Maindata",
"Hierarchy");
vcGantt1.Arrangement = VcArrangementType.vcArrangementTypeHierarchical;
vcGantt1.GroupNodes(true);
```

HierarchyLevelLayout

Read Only Property of VcGantt

This property gives access to the HierarchyLevelLayout object.

	Data Type	Explanation
Property value	VcHierarchyLevelLayout	HierarchyLevelLayout object

HistogramCollection

Read Only Property of VcGantt

This property gives access to the HistogramCollection object that contains all histograms available.

	Data Type	Explanation
Property value	VcHistogramCollection	HistogramCollection object

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
histogramCltn = VcGantt1.HistogramCollection
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
```

HistogramSeparationLineColor

Property of VcGantt

This property lets you set/retrieve the color of the separation lines between histograms. This property also can be set on the **Layout** property page.

	Data Type	Explanation
Property value	System.Drawing.Color	Color value {0...255},{0...255},{0...255})

HorizontalMovementWhileDraggingAllowed

Property of VcGantt

This property lets you set or retrieve whether a node can be moved if the control is the target component of an ongoing Drag&Drop action. The property does not affect activities moved within the same Gantt chart.

	Data Type	Explanation
Property value	System.Boolean	Drag&drop action allowed (true) / not allowed (False) Default value: false

InbuiltMouseCursorWhileDraggingEnabled

Property of VcGantt

This property lets you set or retrieve to/from the target component whether the mouse cursor typical of the VARCHART component should be displayed during a drag&drop procedure . If it is not to be displayed, the drag&drop mouse cursor (arrow and a little square or prohibitory sign) will be displayed. The latter can be replaced by your own mouse cursors, if you implement an event handler for the event **GiveFeedback** of the base class **Control**.

Please also see the VcGantt properties **LeavingControlWhileDragging-Allowed**, **NodeCreationAtDroppingEnabled**, **PhantomDrawingWhile-DraggingEnabled** and the **AllowDrop** property of the base class **Control**.

	Data Type	Explanation
Property value	System.Boolean	The VARCHAR mouse cursor is displayed (true) / not displayed (false) Default value: true

InfoWindow

Read Only Property of VcGantt

This property gives access to the InfoWindow object that designates the information window of a node appearing in a Gantt chart when a node is created or modified.

	Data Type	Explanation
Property value	VcInfoWindow	InfoWindow object

InitialRowCount

Property of VcGantt

This property lets you set or retrieve the number of node rows at the program start. This property also can be set on the **Layout** property page.

	Data Type	Explanation
Property value	System.Int32	Number of node rows at the program start

InPlaceEditingOnGroupsInDiagramEnabled

Property of VcGantt

This property lets you set or retrieve whether at run time in-place editing of group data fields in the diagram should be permitted to the user. For this, the group data have to use their own data tables. You also can set this property on the **General** property page.

Note: If certain data fields are not to be editable, the **Editable** check box in the **Administrative Data Tables** dialog must not be ticked.

Also see **InPlaceEditingOnNodesInDiagramEnabled**, **InPlaceEditingOnNodesInTableEnabled** and **InPlaceEditingOnGroupsInTableEnabled**.

	Data Type	Explanation
Property value	System.Boolean	In-place editing is enabled (True) / not enabled (False) Default value: True

Example Code VB.NET

```
VcGantt1.InPlaceEditingOnGroupsInDiagramEnabled = True
```

Example Code C#

```
vcGantt1.InPlaceEditingOnGroupsInDiagramEnabled = true;
```

InPlaceEditingOnGroupsInTableEnabled

Property of VcGantt

This property lets you set or retrieve whether at run time in-place editing of group data fields in the table should be permitted to the user. For this, the group data have to use their own data tables. You also can set this property on the **General** property page.

Note: If certain data fields are not to be editable, the **Editable** check box in the **Administrate Data Tables** dialog must not be ticked.

Also see **InPlaceEditingOnNodesInDiagramEnabled**, **InPlaceEditingOnNodesInTableEnabled** and **InPlaceEditingOnGroupsInTableEnabled**.

	Data Type	Explanation
Property value	System.Boolean	In-place editing is enabled (True) / not enabled (False) Default value: True

Example Code VB.NET

```
VcGantt1.InPlaceEditingOnGroupsInDiagramEnabled = True
```

Example Code C#

```
vcGantt1.InPlaceEditingOnGroupsInDiagramEnabled = true;
```

InPlaceEditingOnNodesInDiagramEnabled

Property of VcGantt

This property lets you set or retrieve whether at run time in-place editing of node data fields in the diagram should be permitted to the user. You also can set this property on the **General** property page.

Note: If certain data fields are not to be editable, the **Editable** check box in the **Administrative Data Tables** dialog must not be ticked.

Also see **InPlaceEditingOnNodesInDiagramEnabled**, **InPlaceEditingOnNodesInTableEnabled** and **InPlaceEditingOnGroupsInTableEnabled**.

	Data Type	Explanation
Property value	System.Boolean	In-place editing is enabled (True) / not enabled (False) Default value: True

Example Code VB.NET

```
VcGantt1.InPlaceEditingOnGroupsInDiagramEnabled = True
```

Example Code C#

```
vcGantt1.InPlaceEditingOnGroupsInDiagramEnabled = true;
```

InPlaceEditingOnNodesInTableEnabled

Property of VcGantt

This property lets you set or retrieve whether at run time in-place editing of node data fields in the table should be permitted to the user. You also can set this property on the **General** property page.

Note: If certain data fields are not to be editable, the **Editable** check box in the **Administrative Data Tables** dialog must not be ticked.

Also see **InPlaceEditingOnNodesInDiagramEnabled**, **InPlaceEditingOnGroupsInTableEnabled** and **InPlaceEditingOnGroupsInDiagramEnabled**.

	Data Type	Explanation

Example Code VB.NET

```
VcGantt1.InPlaceEditingOnGroupsInDiagramEnabled = True
```

Example Code C#

```
vcGantt1.InPlaceEditingOnGroupsInDiagramEnabled = true;
```

InteractionMode

Property of VcGantt

This property activates/retrieves one of the available modes of interaction.

	Data Type	Explanation
Property value	VcInteractionMode	Modes create link, delete link, create node, delete node, pointer Default value: vcPointer
	Possible Values: .vcCreateBox 36 .vcCreateLink 4 .vcCreateNode 2 .vcDeleteLink 5 .vcDeleteNode 3 .vcPanning 6 .vcPointer 0	Box creating mode Link creating mode Node creating mode Link deleting mode Node deleting mode Panning mode Select mode

Example Code VB.NET

```
VcGantt1.InteractionMode = VcInteractionMode.vcCreateNode
```

Example Code C#

```
vcGantt1.InteractionMode = VcInteractionMode.vcCreateNode;
```

KeepingNodesTogetherDataFieldIndex

Property of VcGantt

This property lets you set or retrieve the data field that controls the node separation of groups. This property only is available if nodes are grouped and the grouping options **In one line** and **Nodes optimized** have been activated (**Grouping** dialog). Then you can select a data field that should be used for the separation. Then all nodes of a group with the same value in this data field will be displayed in one line, even if they will overlap each other.

Tip: Please note that in order to achieve a satisfactory result, the fields have to have the data type **Integer** or **Alphanumeric** and have to lie within the range of 1 - long_MAX (2147483647). If a field has the value 0 the node will not be kept together with the other nodes.

	Data Type	Explanation
Property value	System.Int16	Number of the field that should be used for the separation of nodes in groups

Example Code VB.NET

```
VcGantt1.KeepingNodesTogetherDataFieldIndex = 3
```

Example Code C#

```
vcGantt1.KeepingNodesTogetherDataFieldIndex = 3;
```

LayerCollection

Read Only Property of VcGantt

This property gives access to the LayerCollection object that contains all defined layers.

	Data Type	Explanation
Property value	VcLayerCollection	LayerCollection object

Example Code VB.NET

```
Dim layerCltn As VcLayerCollection
layerCltn = VcGantt1.LayerCollection
```

Example Code C#

```
VcLayerCollection layerCltn = vcGantt1.LayerCollection;
```

LayersWithNonWorkInterval

Property of VcGantt

This property lets you set or retrieve whether workfree intervals are to be displayed in the nodes. This property also can be set on the **Nodes** property page.

Note: **NodesUseCalendars** has to be set to True.

	Data Type	Explanation
Property value	System.Boolean	Show workfree intervals (true)/do not show workfree intervals (false).

Example Code VB.NET

```
VcGantt1.LayersWithNonWorkInterval = True
```

Example Code C#

```
vcGantt1.LayersWithNonWorkInterval = true;
```


LeavingControlWhileDraggingAllowed

Property of VcGantt

This property lets you set or retrieve, whether it should be allowed to drag nodes beyond the borders of the source component. Nodes can then be moved or copied to

- another VARCHART control
- a different control of the same application
- a different application.

Internally, the drag&drop routines implemented in the base class **Control** are triggered, which use the OLE drag&drop feature of the Windows operating system.

Please also see the VcGantt properties **NodeCreationAtDroppingEnabled**, **InbuiltMouseCursorWhileDraggingEnabled**, **PhantomDrawingWhileDraggingEnabled** and the **AllowDrop** property of the base class **Control**.

	Data Type	Explanation
Property value	System.Boolean	Leaving the component is permitted (true) / not permitted (false) Default value: false

LeftTable

Read Only Property of VcGantt

This property gives access to the Table object in order to access the formats used in order to modify its table columns and their headings.

	Data Type	Explanation
Property value	VcTable	Table

Example Code VB.NET

```
Dim table As VcTable
table = VcGantt1.LeftTable
```

Example Code C#

```
VcTable table = vcGantt1.LeftTable;
```

LeftTableDiagramWidthRatio

Property of VcGantt

This property lets you set or retrieve the ratio between the width of the left table and the width of the diagram (in %). If this property is set to -1, the table will always be displayed completely.

	Data Type	Explanation
Property value	System.Int16	Width ratio {-1, 1...100}

Example Code VB.NET

```
VcGantt1.LeftTableDiagramWidthRatio = 40
```

Example Code C#

```
vcGantt1.LeftTableDiagramWidthRatio = 40;
```

LeftTableDiagramWidthRatioEx

Property of VcGantt

This property lets you set or retrieve the ratio between the width of the left table and the width of the diagram (in %). If this property is set to -1, the table will always be displayed completely.

In contrast to the **LeftTableDiagramWidthRatio** property this property returns a "Double" value, thus achieving a higher level of accuracy. The usage of this property has to be enabled by the **UseHigherTableDiagram-WidthRatioPrecision** property or by activating the corresponding option on the **General** property page.

	Data Type	Explanation
Property value	System.Double	Width ratio

Example Code VB.NET

```
VcGantt1.LeftTableDiagramWidthRatioEx = 40
```

Example Code C#

```
vcGantt1.LeftTableDiagramWidthRatioEx = 40;
```

LegendView

Read Only Property of VcGantt

This property gives access to the LegendView object that lets you define the legend view.

	Data Type	Explanation
Property value	VcLegendView	LegendView object

Example Code VB.NET

```
Dim legendview As VcLegendView
legendview = VcGantt1.LegendView
legendview.Visible = True
```

Example Code C#

```
VcLegendView legendview = vcGantt1.LegendView;
legendview.Visible = true;
```

LineFormatCollection

Read Only Property of VcGantt

This property gives access to the LineFormatCollection object that contains all line formats available to the table.

	Data Type	Explanation
Property value	VcLineFormatCollection	LineFormatCollection object

Example Code VB.NET

```
Dim lineFormatCltn As VcLineFormatCollection
Dim lineFormat As VcLineFormat

lineFormatCltn = VcGantt1.LineFormatCollection
For Each lineFormat In lineFormatCltn
    ListBox1.Items.Add(lineFormat.Name)
Next
```

Example Code C#

```
VcLineFormatCollection lineFormatCltn = vcGantt1.LineFormatCollection;
foreach (VcLineFormat lineFormat in lineFormatCltn)
{
    listBox1.Items.Add(lineFormat.Name);
}
```

LinkAppearanceCollection

Read Only Property of VcGantt

This property gives access to the LinkAppearanceCollection object that contains all link appearance objects defined.

	Data Type	Explanation
Property value	VcLinkAppearanceCollection	LinkAppearanceCollection object

Example Code VB.NET

```
Dim linkAppearanceCltn As VcLinkAppearanceCollection
linkAppearanceCltn = VcGantt1.LinkAppearanceCollection
```

Example Code C#

```
VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
```

LinkCollection

Read Only Property of VcGantt

This property gives access to the LinkCollection object that contains all links defined.

	Data Type	Explanation
Property value	VcLinkCollection	LinkCollection object

Example Code VB.NET

```
Dim linkCltn As VcLinkCollection
linkCltn = VcGantt1.LinkCollection
```

Example Code C#

```
VcLinkCollection linkCltn = vcGantt1.LinkCollection;
```

LinkPredecessorDataFieldIndex

Property of VcGantt

This property lets you set or retrieve the data field which holds the identification of the predecessor node of the link. You can only set this property if data was not yet loaded.

This property can also be set on the **Links** property page.

The property `LinkPredecessorDataFieldIndex` is an Indexed Property, which in C# is addressed by the methods `set_LinkPredecessorDataFieldIndex` (identifierIndex, pvn) and `get_LinkPredecessorDataFieldIndex` (identifierIndex).

	Data Type	Explanation
Parameter: ⇒ identifierIndex	System.Int16	Index of predecessor node {0...2}
Property value	System.Int32	Field index of the data definition table

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecord As VcDataRecord

'create Link DataTable
dataTable = VcGantt1.DataTableCollection.Add("LinkDataTable")
VcGantt1.LinksDataTableName = dataTable.Name
dataTable.DataTableFieldCollection.Add("Id").PrimaryKey = True
dataTable.DataTableFieldCollection.Add("Predecessor")
dataTable.DataTableFieldCollection.Add("Successor")
VcGantt1.DataTableCollection.Update()

VcGantt1.LinkPredecessorDataFieldIndex(0) =
VcGantt1.DetectFieldIndex("LinkDataTable", "Id")
VcGantt1.LinkSuccessorDataFieldIndex(0) =
VcGantt1.DetectFieldIndex("LinkDataTable", "Id")

'Load Data
dataTable = VcGantt1.DataTableCollection.DataTableByName("LinkDataTable")
dataRecord = dataTable.DataRecordCollection.Add("1;1;2;")
VcGantt1.EndLoading()
```

Example Code C#

```
VcDataTable dataTable;
VcDataRecord dataRecord;

//create Link DataTable
dataTable = vcGantt1.DataTableCollection.Add("LinkDataTable");
vcGantt1.LinksDataTableName = dataTable.Name;
dataTable.DataTableFieldCollection.Add("Id").PrimaryKey = true;
dataTable.DataTableFieldCollection.Add("Predecessor");
dataTable.DataTableFieldCollection.Add("Successor");
vcGantt1.DataTableCollection.Update();

vcGantt1.set_LinkPredecessorDataFieldIndex(0,
vcGantt1.DetectFieldIndex("LinkDataTable", "Id"));
vcGantt1.set_LinkSuccessorDataFieldIndex(0,
vcGantt1.DetectFieldIndex("LinkDataTable", "Id"));

//Load Data
dataTable = vcGantt1.DataTableCollection.DataTableByName("LinkDataTable");
dataRecord = dataTable.DataRecordCollection.Add("1;1;2;");
vcGantt1.EndLoading();
```

LinksDataTableName

Property of VcGantt

This property lets you set or retrieve the name of the data table which provides the data fields for links.

To make links appear on the screen, also the properties **LinkSuccessorDataFieldIndex** and **LinkPredecessor** need to be set.

This property can also be set on the **Links** property page.

	Data Type	Explanation
Property value	System.String	Name of the data table which provides the fields for the links

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecord As VcDataRecord

'create Link DataTable
dataTable = VcGantt1.DataTableCollection.Add("LinkDataTable")
VcGantt1.LinksDataTableName = dataTable.Name
dataTable.DataTableFieldCollection.Add("Id").PrimaryKey = True
dataTable.DataTableFieldCollection.Add("Predecessor")
dataTable.DataTableFieldCollection.Add("Successor")
VcGantt1.DataTableCollection.Update()

VcGantt1.LinkPredecessorDataFieldIndex(0) =
VcGantt1.DetectFieldIndex("LinkDataTable", "Id")
VcGantt1.LinkSuccessorDataFieldIndex(0) =
VcGantt1.DetectFieldIndex("LinkDataTable", "Id")

'Load Data
dataTable = VcGantt1.DataTableCollection.DataTableByName("LinkDataTable")
dataRecord = dataTable.DataRecordCollection.Add("1;1;2;")
VcGantt1.EndLoading()
```

Example Code C#

```

VcDataTable dataTable;
VcDataRecord dataRecord;

//create Link DataTable
dataTable = vcGantt1.DataTableCollection.Add("LinkDataTable");
vcGantt1.LinksDataTableName = dataTable.Name;
dataTable.DataTableFieldCollection.Add("Id").PrimaryKey = true;
dataTable.DataTableFieldCollection.Add("Predecessor");
dataTable.DataTableFieldCollection.Add("Successor");
vcGantt1.DataTableCollection.Update();

vcGantt1.set_LinkPredecessorDataFieldIndex(0,
vcGantt1.DetectFieldIndex("LinkDataTable", "Id"));
vcGantt1.set_LinkSuccessorDataFieldIndex(0,
vcGantt1.DetectFieldIndex("LinkDataTable", "Id"));

//Load Data
dataTable = vcGantt1.DataTableCollection.DataTableByName("LinkDataTable");
dataRecord = dataTable.DataRecordCollection.Add("1;1;2;");
vcGantt1.EndLoading();

```

LinkSuccessorDataFieldIndex**Property of VcGantt**

This property lets you set or retrieve the data field which holds the successor node of a link. This is only possible as long as no data has been loaded.

This property can also be set on the **Links** property page.

The property LinkSuccessorDataFieldIndex is an Indexed Property, which in C# is addressed by the methods set_LinkSuccessorDataFieldIndex (identifierIndex, pvn) and get_LinkSuccessorDataFieldIndex (identifierIndex).

	Data Type	Explanation
Parameter: ⇒ identifierIndex	System.Int16	Index of successor node {0...2}
Property value	System.Int32	Fieldindex of the data definition table

Example Code VB.NET

```

Dim dataTable As VcDataTable
Dim dataRecord As VcDataRecord

'create Link DataTable
dataTable = VcGantt1.DataTableCollection.Add("LinkDataTable")
VcGantt1.LinksDataTableName = dataTable.Name
dataTable.DataTableFieldCollection.Add("Id").PrimaryKey = True
dataTable.DataTableFieldCollection.Add("Predecessor")
dataTable.DataTableFieldCollection.Add("Successor")
VcGantt1.DataTableCollection.Update()

VcGantt1.LinkPredecessorDataFieldIndex(0) =
VcGantt1.DetectFieldIndex("LinkDataTable", "Id")
VcGantt1.LinkSuccessorDataFieldIndex(0) =
VcGantt1.DetectFieldIndex("LinkDataTable", "Id")

'Load Data
dataTable = VcGantt1.DataTableCollection.DataTableByName("LinkDataTable")
dataRecord = dataTable.DataRecordCollection.Add("1;1;2;")
VcGantt1.EndLoading()

```

Example Code C#

```

VcDataTable dataTable;
VcDataRecord dataRecord;

//create Link DataTable
dataTable = vcGantt1.DataTableCollection.Add("LinkDataTable");
vcGantt1.LinksDataTableName = dataTable.Name;
dataTable.DataTableFieldCollection.Add("Id").PrimaryKey = true;
dataTable.DataTableFieldCollection.Add("Predecessor");
dataTable.DataTableFieldCollection.Add("Successor");
vcGantt1.DataTableCollection.Update();

vcGantt1.set_LinkPredecessorDataFieldIndex(0,
vcGantt1.DetectFieldIndex("LinkDataTable", "Id"));
vcGantt1.set_LinkSuccessorDataFieldIndex(0,
vcGantt1.DetectFieldIndex("LinkDataTable", "Id"));

//Load Data
dataTable = vcGantt1.DataTableCollection.DataTableByName("LinkDataTable");
dataRecord = dataTable.DataRecordCollection.Add("1;1;2;");
vcGantt1.EndLoading();

```

LinkTypeDataFieldIndex**Property of VcGantt**

This property lets you set or retrieve the name of the data field which contains the link type. This is only possible as long as no data was loaded.

This property can also be set on the **Links** property page.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which contains the link type

Example Code VB.NET

```
Dim dataTable As VcDataTable

'create Link DataTable
dataTable = VcGantt1.DataTableCollection.Add("LinkDataTable")
VcGantt1.LinksDataTableName = dataTable.Name
dataTable.DataTableFieldCollection.Add("Id").PrimaryKey = True
dataTable.DataTableFieldCollection.Add("Predecessor")
dataTable.DataTableFieldCollection.Add("Successor")
dataTable.DataTableFieldCollection.Add("LinkType")
VcGantt1.DataTableCollection.Update()
```

Example Code C#

```
VcDataTable dataTable;

//create Link DataTable
dataTable = vcGantt1.DataTableCollection.Add("LinkDataTable");
vcGantt1.LinksDataTableName = dataTable.Name;
dataTable.DataTableFieldCollection.Add("Id").PrimaryKey = true;
dataTable.DataTableFieldCollection.Add("Predecessor");
dataTable.DataTableFieldCollection.Add("Successor");
dataTable.DataTableFieldCollection.Add("LinkType");
vcGantt1.DataTableCollection.Update();
```

MapCollection

Read Only Property of VcGantt

This property gives access to the MapCollection object that contains a defined number of maps. The number of maps is defined by the method **VcMapCollection.SelectMaps**.

	Data Type	Explanation
Property value	VcMapCollection	MapCollection object

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
```

MinimumRowHeight

Property of VcGantt

By this property you can assign a minimum height (unit: 1/100 mm) to a row. The height chosen should correspond to the average height of an activity. This property can also be set on the **Layout** property page.

The minimum row height only becomes effective if there is no activity in the row or if existing activities do not exceed the minimum row height. In all other cases the row height automatically adapts to the space required by the activities. The values permitted range between 2 and 1000.

	Data Type	Explanation
Property value	System.Int32	Minimal row height

Example Code VB.NET

```
VcGantt1.MinimumRowHeight = 100
```

Example Code C#

```
vcGantt1.MinimumRowHeight = 100;
```

MouseProcessingEnabled

Property of VcGantt

This property allows you to process mouse events in your own way. If you want your own processing method of the .NET mouse events MouseDown/Up/Move, then set the **MouseProcessingEnabled** property to False for this time interval. Then VARCHART XGantt will ignore all mouse movements and clicks until this property is set to True again.

	Data Type	Explanation
Property value	System.Boolean	Property active (True)/ not active (False) Default value: True

MoveMode

Property of VcGantt

This property lets you set or retrieve the direction(s) that a node interactively can be moved to.

	Data Type	Explanation

Example Code VB.NET

```

Dim nodeCltn As VcNodeCollection
Dim node As VcNode

nodeCltn = VcGantt1.NodeCollection
nodeCltn.SelectNodes(VcSelectionType.vcAll)

For Each node In nodeCltn
    If node.DataField(2) < Now Then
        node.MoveMode = VcNodeMoveMode.vcNodeMoveModeNoMove
    End If
Next

```

Example Code C#

```

VcNodeCollection nodeCltn = VcGantt1.NodeCollection;
nodeCltn.SelectNodes(VcSelectionType.vcAll);
foreach (VcNode node in nodeCltn)
{
    if (DateTime.Compare(Convert.ToDateTime(node.get_DataField(2)),
DateTime.Today).Equals(true))
        node.MoveMode = VcNodeMoveMode.vcNodeMoveModeNoMove;
}

```

MovingLayersAsNodeWithShiftKeyAllowed**Read Only Property of VcGantt**

This property lets you specify/enquire whether the layers of a marked node are moved as a whole when the shift key is being pressed while dragging (True). Otherwise the layers can be moved individually only (False). This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Boolean	Moving of all layers of a node with shift enabled/disabled Default value: true

Example Code VB.NET

```
VcGantt1.MoveLayersAsNodeWithShiftKey = False
```

MultipleBoxMarkingAllowed**Property of VcGantt**

This property lets you set or retrieve whether at run time the marking of several boxes at the same time is possible or not. If the property is not activated the user has to keep the CTRL key pressed in order to mark several boxes. You can also set this property on the **General** property page

	Data Type	Explanation
Property value	System.Boolean	multiple box marking possible/not possible Default value: False

Example Code VB.NET

```
VcGantt1.MultipleBoxMarking = True
```

Example Code C#

```
vcGantt1.MultipleBoxMarking = true;
```

NodeCalendarNameDataFieldIndex

Property of VcGantt

This property lets you set or retrieve the index of the data field which holds the name of the calendar if you wish to use an individual calendar for a node. Setting this property is only possible if no data was loaded yet.

This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which contains the name of the node calendar

NodeCollection

Read Only Property of VcGantt

This property gives access to the NodeCollection object, that that contains a defined number of nodes. The number of nodes is defined by the method **VcNodeCollection.SelectMaps**

	Data Type	Explanation
Property value	VcNodeCollection	NodeCollection object

Example Code VB.NET

```
Dim nodeCltn As VcNodeCollection
nodeCltn = VcGantt1.NodeCollection
nodeCltn.SelectNodes(VcSelectionType.vcAll)
```

Example Code C#

```
VcNodeCollection nodeCltn = vcGantt1.NodeCollection;
nodeCltn.SelectNodes(VcSelectionType.vcAll);
```

NodeCreationAllowed

Property of VcGantt

This property permits (True) or prohibits (False) the user to create new nodes. If this property is set to **False**, the user cannot activate the **CreateNode** mode. This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Boolean	Generating new nodes enabled/disabled Default value: True

Example Code VB.NET

```
VcGantt1.NodeCreationAllowed = False
```

Example Code C#

```
vcGantt1.NodeCreationAllowed = false;
```

NodeCreationAtDroppingEnabled

Property of VcGantt

This property lets you set or retrieve whether after dropping a dragged object to the target component an activity should be generated automatically in the VARCHART component.

If this property is set to **false**, an event handler needs to be written for the DragDrop event to handle the dropping. The event mentioned and this property are only active if the property **AllowDrop** of the base class **Control** was set to **true**.

Please also see the VcGantt properties **LeavingControlWhileDragging-Allowed**, **InbuiltMouseCursorWhileDraggingEnabled**, **Phantom-DrawingWhileDraggingEnabled** and the **AllowDrop** property of the base class **Control**.

	Data Type	Explanation
Property value	System.Boolean	An activity is to be created (true) / not created (false) Default value: false

NodeCreationViaDoubleClick

Property of VcGantt

This property lets you enable the user to create a new node by double-clicking in the diagram area. Note: The **NodeCreationAllowed** property must be set to True. This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Boolean	Generating new nodes via double-click enabled/disabled Default value: False

Example Code VB.NET

```
VcGantt1.NodeCreationViaDoubleClick = True
```

Example Code C#

```
vcGantt1.NodeCreationViaDoubleClick = true;
```

NodeCreationWithDialog

Property of VcGantt

This property sets whether or not the **Edit Data** dialog box is to appear when a new node is created. The **NodeCreationAllowed** property must be set to **True** to enable the user to create new nodes. This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Boolean	Edit Data dialog appears/does not appear. Default value: False

Example Code VB.NET

```
VcGantt1.NodeCreationWithDialog = False
```

Example Code C#

```
vcGantt1.NodeCreationWithDialog = false;
```

NodeDurationDataFieldIndex

Property of VcGantt

This property lets you set or retrieve the index of the data field that contains the duration of an interactively created node. This is only possible as long as

no data has been loaded. This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the duration of an interactively created node

NodeEndDateDataFieldIndex

Property of VcGantt

This property lets you set or retrieve the index of the data field which holds the end date of an interactively created activity. This is only possible as long as no data has been loaded. This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the end date of an interactively created node

NodeLevelLayout

Read Only Property of VcGantt

This property gives access to the NodeLevelLayout object. This object lets you set or retrieve the properties of the hierarchical arrangement of activities.

	Data Type	Explanation
Property value	VcNodeLevelLayout	NodeLevelLayout object Default value: True

NodeRowNumberDataFieldIndex

Property of VcGantt

This property lets you set or retrieve the index of the data field which holds the row number of an activity. Setting this property is only possible if no data was loaded yet. This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the row number of an activity

Example Code VB.NET

```

Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs)
    Handles MyBase.Load
        VcGantt1.NodeRowNumberDataFieldIndex =
            VcGantt1.DetectFieldIndex("NodeDataTable", "SortNumber")
        'Load data
        LoadData()

        VcGantt1.UpdateRowNumberFields()
        VcGantt1.SaveAsEx("C:\ProjectData.txt", VcEncoding.vcUnicodeEncoding)
End Sub

```

Example Code C#

```

private void Form1_Load(object sender, System.EventArgs e)
{
    vcGantt1.NodeRowNumberDataFieldIndex =
        VcGantt1.DetectFieldIndex("NodeDataTable", "SortNumber");

    // Load data
    loadData();

    vcGantt1.UpdateRowNumberFields();
    vcGantt1.SaveAsEx(@"C:\ProjectData.txt", VcEncoding.vcUnicodeEncoding);
}

```

NodesDataTableName**Property of VcGantt**

This property lets you set or retrieve the name of the data table which provides the data fields for the nodes. This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.String	Name of the data table which provides the fields for the nodes

Example Code VB.NET

```

Dim dataTable As VcDataTable
Dim dataRecord As VcDataRecord

'create Node DataTable
dataTable = VcGantt1.DataTableCollection.Add("NodeDataTable")
VcGantt1.NodesDataTableName = dataTable.Name
dataTable.DataTableFieldCollection.Add("Id").PrimaryKey = True

```


Example Code C#

```

VcDataTable dataTable;
VcDataRecord dataRecord;

//create Node DataTable
dataTable = vcGantt1.DataTableCollection.Add("NodeDataTable");
vcGantt1.NodesDataTableName = dataTable.Name;
dataTable.DataTableFieldCollection.Add("Id").PrimaryKey = true;
//Load Data
dataTable = vcGantt1.DataTableCollection.DataTableByName("NodeDataTable");
dataRecord = dataTable.DataRecordCollection.Add("1;Node One;");
dataRecord = dataTable.DataRecordCollection.Add("2;Node Two;");
vcGantt1.EndLoading();

```

NodeSortingDataFieldIndex**Property of VcGantt**

This property lets you specify the fields that the nodes are to be sorted by. Three sorting levels exist. For each one the field index can be specified. The sorting order you can specify by the **NodeSortingOrder** property. Sorting is to be triggered by the method **SortNodes**.

This property also can be set in the **Grouping** dialog.

The property NodeSortingDataFieldIndex is an Indexed Property, which in C# can be addressed by the methods set_NodeSortingDataFieldIndex (sortLevel, pvn) and get_NodeSortingDataFieldIndex (sortLevel).

	Data Type	Explanation
Parameter: ⇒ sortLevel	System.Int16	Sorting level {0...2}
Property value	System.Int32	Field index of the data definition table

Example Code VB.NET

```

VcGantt1.NodeSortingDataFieldIndex(0) = 11
VcGantt1.NodeSortingOrder(0) = VcNodeSortingOrder.vcDescending
VcGantt1.SortNodes()

```

Example Code C#

```

vcGantt1.set_NodeSortingDataFieldIndex(0,11);
vcGantt1.set_NodeSortingOrder(0, VcNodeSortingOrder.vcDescending);
vcGantt1.SortNodes();

```

NodeSortingOrder

Property of VcGantt

This property specifies the sorting order (ascending or descending) for each of the three sorting levels. The sorting is triggered by the method **SortNodes**. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	Integer	Ascending or descending order Default value: vcAscending
	Possible Values: .vcAscending 1 .vcDescending 2	ascending order Descending order

Example Code VB.NET

```
VcGantt1.NodeSortingDataFieldIndex(0) = 11
VcGantt1.NodeSortingOrder(0) = VcNodeSortingOrder.vcDescending
VcGantt1.SortNodes()
```

Example Code C#

```
vcGantt1.set_NodeSortingDataFieldIndex(0,11);
vcGantt1.set_NodeSortingOrder(0, VcNodeSortingOrder.vcDescending);
vcGantt1.SortNodes();
```

NodeStartDateDataFieldIndex

Property of VcGantt

This property lets you set or retrieve the index of the data field which holds the start date of an interactively created activity. Setting this property is only possible if no data was loaded yet. This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the start date of an interactively created node

NodesUseCalendars

Property of VcGantt

This property specifies whether a calendar is assigned to the nodes. Due to the calendar, the beginning/end of an activity will not be placed on a workfree day when shifted. Also, when calculating durations for activities,

workfree days will be considered. A five-day-calendar is the default calendar. Beside, you can to define your own calendars. This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Boolean	Property active/not active

Example Code VB.NET

```
VcGantt1.NodesUseCalendars = False
```

Example Code C#

```
vcGantt1.NodesUseCalendars = false;
```

NodeToolTipTextDataFieldIndex

Property of VcGantt

This property lets you require/set the index of the data field of a node to store the tooltip texts for VMF files. This text appears when in the WebViewer the right mouse button is pressed.

This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Int16	Index of the node data field for tooltip texts Default value: 4

Example Code VB.NET

```
VcGantt1.NodeToolTipTextDataFieldIndex = 1
```

Example Code C#

```
vcGantt1.NodeToolTipTextDataFieldIndex = 1;
```

NumericScaleCollection

Read Only Property of VcGantt

This property gives access to the NumericScaleCollection object that contains all numeric scales available.

	Data Type	Explanation
Property value	VcNumericScaleCollection	NumericScaleCollection object

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim numericScaleCltn As VcNumericScaleCollection
histogramCltn = VcGantt1.HistogramCollection
numericScaleCltn = histogram.FirstHistogram.NumericScaleCollection
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcNumericScaleCollection numericScaleCltn =
histogramCltn.FirstHistogram().NumericScaleCollection;
```

NumericScaleRescalingAllowed**Property of VcGantt**

This property lets you set or retrieve whether the resolution of the numeric scale can be modified at run time. This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Numerical scale can be rescaled (True)/ cannot be rescaled (False)

Example Code VB.NET

```
VcGantt1.NumericScaleRescalingAllowed = True
```

Example Code C#

```
vcGantt1.NumericScaleRescalingAllowed = true;
```

OLEDragViaDiagram**Property of VcGantt**

This property lets you specify or retrieve whether OLE-DragDrop is enabled in the diagram area.

	Data Type	Explanation
Property value	System.Boolean	OLE DragDrop enabled/not enabled in diagram Default value: True

OLEDragViaTable**Property of VcGantt**

This property lets you specify or retrieve whether OLE-DragDrop is enabled in the table area.

	Data Type	Explanation
Property value	System.Boolean	OLE DragDrop enabled/not enabled in table Default value: True

OverlapLayerEnabled

Property of VcGantt

This property lets you activate the overlap layer of the diagram. Please also see the property **OverlapLayerName** and the property **UsedAsOverlap Layer** at the layer object.

	Data Type	Explanation
Property value	System.Boolean	Overlap layer on (True) / off (False) Default value: False

OverlapLayerName

Property of VcGantt

This property lets you set or retrieve by ist name the layer that is designed to occur as the overlap layer in the diagram. The overlap layer needs to be created and described by methods an properties of the layer object and needs to be marked by the layer property **UsedAsOverlap Layer**. Finally, it needs to be activated by the property **OverlapLayerEnabled** of the Gantt object.

	Data Type	Explanation
Property value	System.String	Name of the overlap layer Default value: " "

PanningModeAllowed

Property of VcGantt

This property lets you move a screen section below a handcursor.

This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Moving screen by mouse allowed (true)/not allowed (false) Default value: False

PartialLoadThreshold

Property of VcGantt

This property lets you set or retrieve a value up to which the loading of nodes will be performed by an optimized partial update and not by a complete update of the data records.

If data records are added, a default loading cycle is started that is optimized for the loading of large amounts of data: structures as grouping and sorting, the calculating of summary bars etc. are being removed and created anew completely. This is convenient when large amounts of data are loaded into an empty chart. If, however, only few records are being loaded into an existing data structure the reloading of only few nodes could take just as long as the loading of the existing nodes because the configuration of the above mentioned structures take up the main part of the performance.

The property **PartialLoadThreshold** offers an alternative: Only few data are inserted in an optimized form into an existing amount of data by a partial update. The value that is specified here sets the threshold value up to which data are being inserted by a "small" update. The recommendable value depends on various factors in the respective application and has to be tested by the user:

- Number of the existing nodes
- Complexity of the Gantt (grouping, sorting, summary bars, links, mapping etc.)

The property should mainly be used when the chart contains already many nodes and only few shall be added at runtime.

This property can be also set by the properties of the control:

OLEDragWithPhantom	True
OLEDropMode	0 - vcOLEDropNone
OverlapLayerEnabled	True
OverlapLayerName	
PartialLoadThreshold	0
PhantomLayerHeight	200
RightTableDiagramWidthRatio	-1
RoundedLinkSlantsEnabled	False
RowHeightReductionEnabled	False
RowMargins	50
ScrollEventsEnabled	True
SelectedRowBackColorAsARGB	0
ShowNonWorkInterval	False
ShowTimeScaleDialog	True

Tip: The optimization can currently only be used for the **Maindata** table. Hence the setting will be ignored if data from other tables or links are being loaded in a loading cycle.

	Data Type	Explanation
Property value	System.Int32	Number of nodes up to which loading of nodes will be performed partially Default value: 0

PhantomDrawingWhileDraggingEnabled

Property of VcGantt

This property lets you set or retrieve to/from the target component whether the default phantom should be generated. This is possible only if the data passed exist in CSV format, and if they correspond to the data definition of the target component.

During dragging, the data is in the DataObject, which is passed by the events **DragEnter** and **DragOver** of the base class **Control**.

Please also see the VcGantt properties **LeavingControlWhileDragging-Allowed**, **NodeCreationAtDroppingEnabled**, **InbuiltMouseCursorWhile-DraggingEnabled** and the **AllowDrop** property of the base class **Control**.

	Data Type	Explanation
Property value	System.Boolean	The phantom is to be displayed (true) / is not to be displayed (false) Default value: false

PhantomLayerHeight

Property of VcGantt

By this property you can set or retrieve the height of the layer phantom (in 1/100 mm) that appears when a node is created interactively.

	Data Type	Explanation
Property value	System.Int16	Height of the layer phantom

Example Code VB.NET

```
Dim phantomLayerHeight As Integer
phantomLayerHeight = VcGantt1.PhantomLayerHeight
```

Example Code C#

```
int phantomLayerHeight = vcGantt1.PhantomLayerHeight;
```

Printer

Read Only Property of VcGantt

This property gives access to the printer object. This object lets you set or retrieve the properties of the current printer.

	Data Type	Explanation
Property value	VcPrinter	Printer object

Example Code VB.NET

```
Dim printerZoomfactor As Integer
Dim printerCuttingMarks As String
printerZoomfactor = VcGantt1.Printer.ZoomFactor
printerCuttingMarks = VcGantt1.Printer.CuttingMarks
```

Example Code C#

```
int printerZoomfactor = vcGantt1.Printer.ZoomFactor;
bool printerCuttingMarks = vcGantt1.Printer.CuttingMarks;
```

ResourceScheduler2

Read Only Property of VcGantt

This property gives access to the ResourceScheduler2 object for resource scheduling.

	Data Type	Explanation
Property value	VcResourceScheduler2	ResourceScheduler2 object

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.TaskDataTableName = "Task"
VcGantt1.ResourceScheduler2.TaskDueDateFieldIndex = 1
VcGantt1.ResourceScheduler2.TaskReleaseDateFieldIndex = 2
'...
VcGantt1.ResourceScheduler2.Process()
```

Example Code C#

```
vcGantt1.ResourceScheduler2.TaskDataTableName = "Task";
vcGantt1.ResourceScheduler2.TaskDueDateFieldIndex = 1;
vcGantt1.ResourceScheduler2.TaskReleaseDateFieldIndex = 2;
//...
vcGantt1.ResourceScheduler2.Process();
```

RightTable

Read Only Property of VcGantt

This property gives access to the second table object on the right in order to access the formats used or to modify the table columns/headings.

	Data Type	Explanation
Property value	VcTable	Second table on the right

Example Code VB.NET

```
Dim rightTable As VcTable
rightTable = VcGantt1.RightTable
```

Example Code C#

```
VcTable rightTable = vcGantt1.RightTable;
```

RightTableDiagramWidthRatio

Property of VcGantt

This property lets you set or retrieve the ratio between the width of the right table and the width of the diagram (in %). If this property is set to -1, the table will always be displayed completely.

	Data Type	Explanation
Property value	System.Int16	Width ratio {-1, 1...100}

Example Code VB.NET

```
VcGantt1.RightTableDiagramWidthRatio = 40
```

Example Code C#

```
vcGantt1.RightTableDiagramWidthRatio = 40;
```

RightTableDiagramWidthRatioEx**Property of VcGantt**

This property lets you set or retrieve the ratio between the width of the right table and the width of the diagram (in %). If this property is set to -1, the table will always be displayed completely.

In contrast to the **RightTableDiagramWidthRatio** property this property returns a "Double" value, thus achieving a higher level of accuracy. The usage of this property has to be enabled by the **UseHigherTableDiagram-WidthRatioPrecision** property or by activating the corresponding option on the **General** property page.

	Data Type	Explanation
Property value	System.Double	Width ratio

Example Code VB.NET

```
VcGantt1.RightTableDiagramWidthRatioEx = 40
```

RoundedLinkSlantsEnabled**Property of VcGantt**

This property lets you set or retrieve whether the slants of links of the routing type **vcLRTOrthogonalDistinguishable** are to be displayed as quarter circles instead of straight lines. This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Slants of links are to be displayed/not displayed as quarter circles Default value: false

Example Code VB.NET

```
VcGantt1.RoundedLinkSlantsEnabled = True
```

Example Code C#

```
vcGantt1.RoundedLinkSlants.Enabled = true;
```

RowHeightReductionEnabled**Property of VcGantt**

This property controls the way of calculating the row height in the diagram. If it is set to **false**, the vertical offsets of the layers are applied by using an imaginary zero line in the vertical center of a node line. To keep the zero line always in the center of the row, it thus may happen that either the top or the bottom row margin will seem rather broad . The layers with a vertical offset of 0, however, stay always vertically centered .

If the property is set to **true**, the imaginary zero line is still used but its position is no longer necessarily in the center of the row but so that the row height is as low as possible. Thus it may happen that layers with a vertical offset of 0 are not on the same level as the vertical centered text of the corresponding table row.

This feature can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Reduction of row height allowed (true)/not allowed (false) Default value: False

Example Code VB.NET

```
VcGantt1.RowHeightReductionEnabled = True
```

Example Code C#

```
vcGantt1.RowHeightReductionEnabled = true;
```

RowMargins**Property of VcGantt**

This property lets you set or retrieve the width between the upper/ lower node margins and the upper/lower margins of the node rows. This property can also be set on the **Layout** property page.

	Data Type	Explanation
Property value	System.Int32	Distance (in 1/100 mm) between the upper/ lower node margins and the upper/lower margins of the node rows

Example Code VB.NET

```
VcGantt1.RowMargins = 100
```

Example Code C#

```
vcGantt1.RowMargins = 100
```

Sash3DStyleEnabled

Property of VcGantt

This property returns/sets whether the sash 3D style is enabled.

	Data Type	Explanation
Property value	System.Boolean	3D style of sash switched on/off Default value: True

SashThickness

Property of VcGantt

This property returns/sets the sash thickness. Value range: 3 - 20 pixels.

The property SashThickness is an Indexed Property, which in C# is addressed by the method `get_SashThickness (gridIndex)` .

	Data Type	Explanation
Property value	System.Integer	Modify sash thickness Default value: 4

Scheduler

Read Only Property of VcGantt

This property returns the VcScheduler object.

	Data Type	Explanation
Property value	VcScheduler	Returns the VcScheduler object

ScrollEventsEnabled

Property of VcGantt

This property lets you enable or disable the scroll events **VcComponentScrolled**, **VcComponentScrolling**, **VcDiagramHorizontalScrolled** and **VcDiagramHorizontalScrolling**. This feature can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Scroll events enabled (True) or disabled (False) Default value: False

Example Code VB.NET

```
VcGantt1.ScrollEventsEnabled = True
```

Example Code C#

```
vcGantt1.ScrollEventsEnabled = true;
```

SelectedNodesMovingTogether

Property of VcGantt

This property lets you set or retrieve whether the user can move the marked nodes collectively. If disabled, only single nodes (depending on whether on the **Nodes** property page the **All layers moving together** check box was ticked) or layers can be moved by the mouse, even if several nodes were marked.

This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Boolean	All marked nodes can be moved together (True)./Only single layers or nodes can be moved by the mouse, even if several nodes were marked (False).

Example Code VB.NET

```
VcGantt1.SelectedNodesMovingTogether = True
```

Example Code C#

```
vcGantt1.SelectedNodesMovingTogether = true;
```

SelectedRowBackgroundColor**Property of VcGantt**

By this property you can assign a color to a selected row. You can use an alpha value that sets the degree of transparency to the color, in order to put a colored fog on the background color of the row (see properties **DiagramBackgroundColor** and **DiagramAlternatingRowBackgroundColor**).

The color is disabled by default, since the default value is fully transparent. You can also set this property on the **Layout** property page.

	Data Type	Explanation
Property value	System.Drawing.Color	ARGB color values ({0...255},{0...255},{0...255},{0...255}) Default value: Color.FromArgb(0,0,0,0)

SelectionViaRubberRectAllowed**Property of VcGantt**

This property lets you set/retrieve whether nodes can be selected in the empty diagram area by a rubber rectangle, drawn by mouse.

This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Zooming allowed (true)/not allowed (False) Default value: False

ShowSnapLines**Property of VcGantt**

This property enables snap lines to be shown while nodes are being resized or dragged with the snap target mode switched on. These lines help to better recognize the defined snap targets.

This feature can also be switched on on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Boolean	Snap lines are/ are not shown Default value: false

ShowSnapMarkings

Property of VcGantt

This property enables snap markings to be shown at the nodes being defined as snap targets while nodes are being resized or dragged with the snap target mode switched on. These markings help to better recognize the defined snap targets.

This feature can also be switched on on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Boolean	Snap markings are/ are not shown Default value: false

SnapTargetNodesSelectionMode

Read Only Property of VcGantt

This property lets you specify whether nodes are selected automatically or manually when moving with the snap target mode switched on. The property **VcNode.SnapTargetMode** selects the nodes as possible snap targets when manual selection is switched on.

	Data Type	Explanation
Property value	VcSnapTargetNodesSelectionMode	Nodes selection mode for moving with snap targets switched on Default value: vcAutomatically

StartDateForAutomaticScheduling

Property of VcGantt

This property lets you set or retrieve the start value for autoscheduling of the current project (**Schedule** property page).

	Data Type	Explanation
Property value	System.DateTime	Start date

SubRowMargins

Read Only Property of VcGantt

This property lets you set or retrieve the vertical offset between sub rows (unit: 1/100 mm). Sub rows only come into existence if groups are displayed in an optimized way. Then nodes of the group are distributed to sub rows to prevent them from overlapping. This property can also be set on the **Layout** property page.

	Data Type	Explanation
Property value	System.Int32	width between sub rows by 1/100 mm {0...200} Default value: 50

Example Code VB.NET

```
VcGantt1.SubRowMargins = 100
```

SummaryBarsVisible

Property of VcGantt

This property lets you set or retrieve whether summary bars are visible or not.

The property SummaryBarsVisible is an Indexed Property, which in C# can be addressed by the methods set_SummaryBarsVisible (groupingLevel, pvn) and get_SummaryBarsVisible (groupingLevel).

	Data Type	Explanation
Parameter: ⇒ groupingLevel	System.Int16	(not for hierarchy) grouping level (GroupingLevel = -1: reading: all levels, writing: at least one level)
Property value	System.Boolean	Summary bars visible (True)/ invisible (False)

Example Code VB.NET

```
VcGantt1.SummaryBarsVisible(-1) = True
```


Example Code C#

```
vcGantt1.set_SummaryBarsVisible(-1, true);
```

TableCollection**Read Only Property of VcGantt**

This property gives access to the table collection object that contains all tables available.

	Data Type	Explanation
Property value	VcTableCollection	Table collection object returned

Example Code VB.NET

```
Dim tableCltn As VcTableCollection
Dim table As VcTable

tableCltn = VcGantt1.TableCollection
For Each table In tableCltn
    ListBox1.Items.Add(table.Name)
Next
```

Example Code C#

```
VcTableCollection tableCltn = vcGantt1.TableCollection;
foreach(VcTable table in tableCltn)
    listBox1.Items.Add(table.Name);
```

TableColumnWidthOptimizationAllowed**Property of VcGantt**

This property permits (True) or prohibits (False) the user to have the column width rescaled automatically. The optimization will be triggered when the user double-clicks on the separation line between the column to be optimized and the column on its right. Thereafter the event **VcTableColumnWidthOptimizing** is triggered. When the column width was modified, the event **VcTableColumnWidthChanging** will occur.

This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Optimizing enabled/disabled Default value: True

Example Code VB.NET

```
VcGantt1.TableColumnWidthOptimizationAllowed = False
```

Example Code C#

```
vcGantt1.TableColumnWidthOptimizationAllowed = false;
```

TextEntrySupplyingEventEnabled**Property of VcGantt**

This property lets you activate the **VcTextEntrySupplying** event. This event lets you modify the texts of context menus, dialog boxes, error messages, months' and days' names etc. that occur during run time, for example for translation into different languages.

This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Property active/not active

Example Code VB.NET

```
VcGantt1.TextEntrySupplyingEventEnabled = True
```

Example Code C#

```
vcGantt1.TextEntrySupplyingEventEnabled = false;
```

TimeScaleCollection**Read Only Property of VcGantt**

This property gives access to the TimescaleCollection object and thus to the time scales available.

	Data Type	Explanation
Property value	VcTimeScaleCollection	TimeScaleCollection object

Example Code VB.NET

```
Dim timeScaleCltn As VcTimeScaleCollection
timeScaleCltn = VcGantt1.TimeScaleCollection
```

Example Code C#

```
VcTimeScaleCollection timeScaleCltn = vcGantt1.TimeScaleCollection;
```

TimeScaleDialogEnabled

Property of VcGantt

This property lets you set or retrieve whether the **Edit Timescale** dialog box is to appear when the user double-clicks on the time scale. This property also can be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Property active (True)/ not active (False) Default value: true

Example Code VB.NET

```
VcGantt1.TimeScaleDialogEnabled = False
```

Example Code C#

```
vcGantt1.TimeScaleDialogEnabled = false;
```

TimeScaleEnd

Property of VcGantt

This property lets you set or retrieve the end of the timescale. When setting, the date of the end needs to be later than the date of the start (also see the **TimeScaleStart** property), otherwise the setting will be ignored by XGantt. At the same time the sequence of the statements set needs to be vice versa. We recommend to use the sequence of statements as shown in the source code example below.

Note: The end date is not included. If you specify **TimeScaleEnd** = "31.12.02" for example, the last day displayed will be the 30.12.02.

	Data Type	Explanation
Property value	System.DateTime	End date of the time scale {1.1.1980...31.12.2035}

Example Code VB.NET

```
'Timescale from 01.10.2014 to 30.11.2014
VcGantt1.TimeScaleEnd = "01.12.2014"
VcGantt1.TimeScaleStart = "01.10.2014"
VcGantt1.TimeScaleEnd = "01.12.2014"
```

Example Code C#

```
//Timescale from 01.10.2014 to 30.11.2014
vcGantt1.TimeScaleEnd = Convert.ToDateTime("01.12.14");
vcGantt1.TimeScaleStart = Convert.ToDateTime("01.10.14");
vcGantt1.TimeScaleEnd = Convert.ToDateTime("01.12.14");
```

TimeScaleRescalingAllowed**Property of VcGantt**

This property permits (True) or prohibits (False) the user to rescale the timescale. If the user is allowed to rescale the timescale, the event **VcTimeScaleSectionRescaling** will be triggered after rescaling the timescale.

This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Rescaling enabled/disabled Default value: True

Example Code VB.NET

```
VcGantt1.TableColumnWidthOptimizationAllowed = False
```

Example Code C#

```
vcGantt1.TableColumnWidthOptimizationAllowed = false;
```

TimeScaleStart**Property of VcGantt**

This property lets you set or retrieve the start of the timescale. When setting, the date of the start needs to be earlier than the date of the end (also see the **TimeScaleEnd** property), otherwise the setting will be ignored by XGantt. At the same time the sequence of the statements set needs to be vice versa. We recommend to use the sequence of statements as shown in the source code example below.

	Data Type	Explanation
Property value	System.DateTime	Start date of the time scale {1.1.1980...31.12.2035}

Example Code VB.NET

```
'Timescale from 01.10.2014 to 30.11.2014
VcGantt1.TimeScaleEnd = "01.12.2014"
VcGantt1.TimeScaleStart = "01.10.2014"
VcGantt1.TimeScaleEnd = "01.12.2014"
```

Example Code C#

```
//Timescale from 01.10.2014 to 30.11.2014
vcGantt1.TimeScaleEnd = Convert.ToDateTime("01.12.14");
vcGantt1.TimeScaleStart = Convert.ToDateTime("01.10.14");
vcGantt1.TimeScaleEnd = Convert.ToDateTime("01.12.14");
```

TimeUnit

Property of VcGantt

This property lets you set or retrieve the time unit used for the calculation of the duration (see "Layers") and for generating and modifying nodes interactively. If for example you have chosen the unit of a day, nodes can be generated or shifted by steps of days only, and the duration of nodes will also be calculated in days. This property can be also set on the **General** property page.

Note: If you want to change the time unit, you should do this before reading data because modifications set later will not be effective.

	Data Type	Explanation
Property value	VcTimeUnit	Time unit Default value: vcDay
	Possible Values: .vcDay 5 .vcHour 6 .vcMinute 7 .vcSecond 8	Time unit day Time unit hour Time unit minute Time unit second

Example Code VB.NET

```
Dim timeUnit As VcTimeUnit
timeUnit = VcGantt1.TimeUnit
```

Example Code C#

```
VcTimeUnit timeUnit = vcGantt1.TimeUnit;
```

TimeUnitsPerStep

Property of VcGantt

This property lets you specify the number of time units covered by minimum interactive shifting of a node. This property also can be set on the **General** property page (**Smallest time interval**).

	Data Type	Explanation
Property value	System.Int16	Number of time units per step Default value: 1

Example Code VB.NET

```
VcGantt1.TimeUnitsPerStep = 4
```

Example Code C#

```
vcGantt1.TimeUnitsPerStep = 4;
```

ToolTipChangeDuration

Property of VcGantt

By this property you can set the duration that elapses before a subsequent tool tip window appears when the pointer moves to a different object. Unit: milliseconds. To reset this delay time to its default value of 98 msec, please set it to -1.

	Data Type	Explanation
Property value	System.Int32	Duration in milliseconds. Maximum value: 32767 msec Default value: -1

Example Code VB.NET

```
VcGantt1.ToolTipChangeDuration = 1000
```

Example Code C#

```
vcGantt1.ToolTipChangeDuration = 1000;
```

ToolTipDuration

Property of VcGantt

By this property you can set the duration of the tool tip window to remain visible if the pointer is stationary within the bounding rectangle of an object.

Unit: milliseconds. To reset this delay time to its default value of 5,000 msec, please set it to -1.

	Data Type	Explanation
Property value	System.Int32	Duration in milliseconds. Maximum value: 32767 msec Default value: -1

Example Code VB.NET

```
VcGantt1.ToolTipDuration = 1000
```

Example Code C#

```
vcGantt1.ToolTipDuration = 1000;
```

ToolTipPointerDuration

Property of VcGantt

By this property you can set the duration during which the pointer must remain stationary within the bounding rectangle of an object before the tool tip window appears. Unit: milliseconds. To reset this delay time to its default value of 480 msec, please set it to -1.

	Data Type	Explanation
Property value	System.Int32	Duration in milliseconds Default value: -1

Example Code VB.NET

```
VcGantt1.ToolTipPointerDuration = 1000
```

Example Code C#

```
vcGantt1.ToolTipPointerDuration = 1000;
```

ToolTipShowAfterClick

Property of VcGantt

By this property you can set whether a tool tip window should disappear when its object is clicked (default behavior) or whether it should remain for the times set to it.

	Data Type	Explanation
Property value	System.Boolean	Tool tip window disappears (false) or remains (true) Default value: False

Example Code VB.NET

```
VcGantt1.ToolTipShowAfterClick = True
```

Example Code C#

```
vcGantt1.ToolTipShowAfterClick = true;
```

ToolTipTextSupplyingEventEnabled

Property of VcGantt

This property lets you activate/deactivate the event **VcToolTipTextSupplying**. This property also can be set on the **General** property page. The event **VcToolTipTextSupplying** lets you edit tooltip texts.

	Data Type	Explanation
Parameter: ⇒ Rückgabewert	System.Boolean	Property active/not active
Property value	System.Boolean	Scroll event enabled (True) or disabled (False) Default value: False

Example Code VB.NET

```
VcGantt1.ToolTipTextSupplyingEventEnabled = True
```

Example Code C#

```
vcGantt1.ToolTipTextSupplyingEventEnabled = true;
```

TrackingSpaceBackgroundColor

Property of VcGantt

This property lets you set or retrieve the tracking space background color. This property also can be set on the **Layout** property page.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255})

Example Code VB.NET

```
VcGantt1.TrackingSpaceBackgroundColor = System.Drawing.Color.Blue
```









Example Code C#


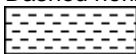
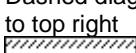
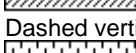
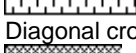
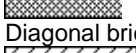
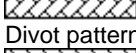
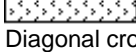

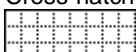
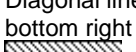
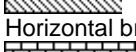
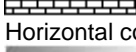

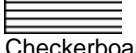


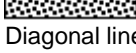

```
vcGantt1.DiagramTrackingSpaceColor = System.Drawing.Color.Blue;
```




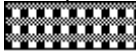



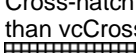
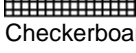



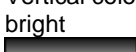
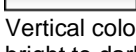




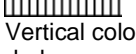
TrackingSpacePattern

Property of VcGantt

This property lets you set or retrieve the background pattern of the tracking space.

	Data Type	Explanation
Property value	VcFillPattern	Pattern type
	Possible Values: .vc05PercentPattern... vc90PercentPattern 01 - 11	Dots in foreground color on background color, the density of the foreground color increasing with the percentage 
	.vcAeroGlassPattern 44	Vertical color gradient in the color of the fill pattern 
	.vcBDiagonalPattern 5	Diagonal lines slanting from bottom left to top right 
	.vcCrossPattern 6	Cross-hatch pattern 
	.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width 
	.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 
	.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width 
	.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 

.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
.vcDashedHorizontalPattern 2026	Dashed horizontal lines 
.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 
.vcDashedVerticalPattern 2027	Dashed vertical lines 
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small 
.vcDiagonalBrickPattern 2032	Diagonal brick pattern 
.vcDivotPattern 2036	Divot pattern 
.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines 
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines 
.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right 
.vcHorizontalBrickPattern 2033	Horizontal brick pattern 
.vcHorizontalGradientPattern 52	Horizontal color gradient 
.vcHorizontalPattern 3	Horizontal lines 
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern 
.vcLargeConfettiPattern 2029	Confetti pattern, large 
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern 
.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern 
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern 
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern 

.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern 
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern 
.vcNoPattern 1276	No fill pattern
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large 
.vcPlaidPattern 2035	Plaid pattern 
.vcShinglePattern 2039	Diagonal shingle pattern 
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern 
.vcSmallConfettiPattern 2028	Confetti pattern 
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern 
.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares 
.vcSpherePattern 2041	Checkerboard of spheres 
.vcTrellisPattern 2040	Trellis pattern 
.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright 
.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark 
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright 
.vcVerticalGradientPattern 62	Vertical color gradient 
.vcVerticalPattern 2	Vertical lines 
.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark 
.vcWavePattern 2031	Horizontal waves pattern 
.vcWeavePattern 2034	Interwoven stripes pattern 

.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern
.vcWideUpwardDiagonalPattern 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of vcBDiagonalPattern
.vcZigZagPattern 2030	Horizontal zig-zag lines

TrackingSpacePatternColor

Property of VcGantt

This property lets you set or retrieve the pattern color of the tracking space. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color.

	Data Type	Explanation
Property value	System.Drawing.Color	ARGB color values {0...255},{0...255},{0...255},{0...255})

UpdateBehaviorCollection

Read Only Property of VcGantt

This property gives access to the update behavior collection object that contains all update behaviors available.

	Data Type	Explanation
Property value	VcUpdateBehaviorCollection	UpdateBehaviorCollection object

Example Code VB.NET

```
Dim updBehCltn As VcUpdateBehaviorCollection
updBehCltn = VcGantt1.UpdateBehaviorCollection
```

Example Code C#

```
VcUpdateBehaviorCollection updBehCltn = vcGantt1.UpdateBehaviorCollection;
```

UseHigherDiagramHistogramHeightRatioPrecision

Property of VcGantt

Set this property to "True" to enable the usage of the more accurate method **DiagramHistogramHeightRatioEx** or the event **VcHistogramHeightChangingEx** that return a value of the type "Double" to calculate the width ratio between diagram and histogram.

If this property is set to the default value "False", the method **DiagramHistogramHeightRatio** or the event **VcHistogramHeight** are used.

This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	More accurate methods to calculate the diagram/histogram height ratio are (True)/are not (False) used Default value: False

Example Code VB.NET

```
VcGantt1.UseHigherTableDiagramHeightRatioPrecision = False
```

Example Code C#

```
vcGantt1.UseHigherTableDiagramHeightRatioPrecision = true;
```

UseHigherTableDiagramWidthRatioPrecision

Property of VcGantt

Set this property to "True" to enable the usage of the more accurate methods **LeftTableDiagramWidthRatioEx** and **RightTableDiagramWidthRatioEx** or the event **VcTableWidthChangingEx** that all return a value of the type "Double" to calculate the width ratio between table and diagram.

If this property is set to the default value "False" then the methods **LeftTableDiagramWidthRatio** and **RightTableDiagramWidthRatio** or the event **VcTableWidthChanging** are used.

This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	More accurate methods to calculate the table(s)/diagram width ratio are (True)/are not (False) used Default value: False

UseSnapTargetsInInteractions

Read Only Property of VcGantt

This property lets you set or retrieve whether the snap targets are used on node/layer interactions.

	Data Type	Explanation
Property value	System.Boolean	Snap targets are used/not used on node/layer interactions

UseTwinLineSashPhantom

Property of VcGantt

This property returns/sets whether a single or a double phantom line appears when interactively moving the sash with **standard** update behavior switched on.

	Data Type	Explanation
Property value	System.Boolean	Double phantom line while moving sash switched on/off Default value: True

VerticalNodeMovementAllowed

Property of VcGantt

Returns/Sets whether nodes are allowed to be moved vertically in the diagram. This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Boolean	Vertical node movement in diagram enabled/disabled Default value: false

VerticalNodeMovementViaTableAllowed

Property of VcGantt

Returns/Sets whether nodes are allowed to be moved vertically in the table. This property also can be set on the **Nodes** property page.

	Data Type	Explanation
Property value	System.Boolean	Vertical node movement in table enabled/disabled Default value: false

ViewComponentsBackgroundColor

Property of VcGantt

This property lets you set or retrieve the diagram background color. If you combine this property with the property **DiagramAlternatingRowBackgroundColor** you can generate a color pattern that alternates linewise. This property also can be set on the **Layout** property page.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255}) Default value: SystemDrawing.Color.White

Example Code VB.NET

```
VcGantt1.ViewComponentsBackgroundColor = System.Drawing.Color.Blue
```

Example Code C#

```
vcGantt1.ViewComponentsBackgroundColor = System.Drawing.Color.Blue;
```

ViewComponentsBorderColor

Property of VcGantt

This property lets you set or retrieve the border color of all frames at one time. This property also can be set on the **Layout** property page.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255}) Default value: SystemDrawing.Color.White

Example Code VB.NET

```
VcGantt1.ViewBorderColor = System.Drawing.Color.Blue
```

Example Code C#

```
vcGantt1.ViewBorderColor = System.Drawing.Color.Blue;
```

WaitCursorEnabled

Property of VcGantt

This property lets you set or returns whether a wait cursor appears on time critical operations (like SheduleProject).

The property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Wait cursor is set/is not set Default value: False

WorldView

Read Only Property of VcGantt

This property gives access to the VcWorldView object, that defines the world view (complete view) of the diagram.

	Data Type	Explanation
Property value	VcWorldView	World View object

Example Code VB.NET

```
Dim worldview As VcWorldView
worldview = VcGantt1.WorldView
worldview.Visible = True
```

Example Code C#

```
VcWorldView worldview = vcGantt1.WorldView;
worldview.Visible = true;
```

ZoomFactor

Property of VcGantt

This property lets you set or retrieve the absolute zoom factor in percent (zoom factor = 100: original size, zoom factor > 100: enlargement, zoom factor < 100: reduction).

The absolute zoom factor is a rounded value and thus may display some inaccuracy.

Please see also the VcGantt methods **FitChartIntoView()** and **Zoom()**.

	Data Type	Explanation
Property value	System.Int16	absolute zoom factor (%)

Example Code VB.NET

```
VcGantt1.ZoomFactor = 150
```

Example Code C#

```
vcGantt1.ZoomFactor = 150;
```

ZoomingPerMouseWheelAllowed

Property of VcGantt

This property lets you set or retrieve whether zooming by mouse wheel should be allowed to the user. This property also can be set on the **General** property page.

	Data Type	Explanation
Property value	System.Boolean	Zooming allowed (true)/not allowed (False) Default value: False

Example Code VB.NET

```
VcGantt1.ZoomingPerMouseWheelAllowed = True
```

Example Code C#

```
VcGantt1.ZoomingPerMouseWheelAllowed = true;
```

Methods

ConvertDistance

Method of VcGantt

By this method you can convert distances from the unit of 1/100 mm into the unit of pixels, or vice versa. You can choose between x- and y-direction of the distance. The conversion takes into account the zoom factor set at a time (also see property **VcGantt.ZoomFactor**).

	Data Type	Explanation
Parameter: ⇒ conversionType	VcDistanceConversionType Possible Values: .vcXCentiMillimetersToPixels 1 .vcXPixelsToCentiMillimeters 3 .vcYCentiMillimetersToPixels 2 .vcYPixelsToCentiMillimeters 4	Conversion type Conversion of a distance in x-direction, from 1/100 millimeters to pixels. Conversion of a distance in x-direction, from pixels to 1/100 millimeters. Conversion of a distance in y-direction, from 1/100 millimeters to pixels. Conversion of a distance in y-direction, from pixels to 1/100 millimeters.
⇒ value	System.Int32	Number of source units (that are to be converted)
Return value	System.Int32	Number of target units (into which was converted)

DeleteLinkRecord

Method of VcGantt

This method lets you delete a link between two nodes. The link record will be identified by the primary keys set in the **Administrate Data Tables** dialog.

	Data Type	Explanation
Parameter: ⇒ linkRecordContent	System.Object	Content of the link record
Return value	System.Boolean	Link record was/was not deleted successfully.

Example Code VB.NET

```
VcGantt1.DeleteLinkRecord("A100;A105;;")
```

Example Code C#

```
vcGantt1.DeleteLinkRecord("A100;A105;;");
```

DeleteNodeRecord

Method of VcGantt

This method lets you delete a node. The node will be identified by the primary key in the node record. The data field that is used for the identification of nodes is set in the **Administrate Data Tables** dialog.

	Data Type	Explanation
Parameter: ⇒ nodeRecordContent	System.Object	Content of the node record
Return value	System.Boolean	Node record was/was not deleted successfully.

Example Code VB.NET

```
VcGantt1.DeleteNodeRecord("A100;;;;;")
```

Example Code C#

```
vcGantt1.DeleteNodeRecord("A100;;;;;");
```

DetectDataTableFieldName**Method of VcGantt**

This property lets you retrieve the name of a data table field by its index.

	Data Type	Explanation
Parameter: ⇒ fieldIndex	System.Int32	Index of the data table field of which the name is to be retrieved
Return value	System.String	Name of the data table field returned

Example Code VB.NET

```
'Find the name of a DataTableField
Dim fieldName As String

fieldName = VcGantt1.DetectDataTableFieldName(0)
```

Example Code C#

```
//Find the name of a DataTableField
string fieldName = vcGantt1.DetectDataTableFieldName(0);
```

DetectDataTableName**Method of VcGantt**

This property lets you retrieve the name of a data table by its index.

	Data Type	Explanation
Parameter: ⇒ fieldIndex	System.Int32	Index of the data table of which the name is to be retrieved
Return value	System.String	Name of the data table returned

Example Code VB.NET

```
'Find the name of a DataTable
Dim tableName As String

tableName = VcGantt1.DetectDataTableName(0)
```

Example Code C#

```
//Find the name of a DataTable
string tableName = vcGantt1.DetectDataTableName(0);
```

DetectFieldIndex**Method of VcGantt**

This property lets you retrieve the index of a data table field by its name and the name of the data table.

	Data Type	Explanation
Parameter:		
⇒ dataTableName	System.String	Name of the data table that holds the field of which the index is to be retrieved
⇒ dataTableFieldName	System.String	Name of the data table field of which the index is to be retrieved
Return value	System.Int32	Index of the data table field returned

Example Code VB.NET

```
'Find the index of a DataTableField
Dim fieldIndex As Integer

fieldIndex = VcGantt1.DetectFieldIndex("Maindata", "Name")
```

Example Code C#

```
//Find the index of a DataTableField
int fieldIndex = vcGantt1.DetectFieldIndex("Maindata", "Name");
```

DumpConfiguration**Method of VcGantt**

This method lets you save the configuration that consists of the .INI and the .IFD file.

The method should only be used for diagnosis purposes.

	Data Type	Explanation
Parameter:		
⇒ FileName	System.String	File name (including a path, if necessary)

⇒ encoding	VcEncoding Possible Values: .vcANSIEncoding 1 .vcUnicodeEncoding 2	Mode of encoding If a file was saved in ANSI encoding, it depends on the local settings of the Windows operating system. The file then contains characters which can be read correctly only if the language settings are the same as the ones that it was stored by. Saving a file in Unicode encoding makes it independent of whatever settings and hence should be the preferred mode if possible. If a file that was saved in Unicode encoding is to be loaded in Visual Basic 6 independently of the VARCHART control, it has to be treated in a special way.
Return value	System.Boolean	File was (True)/was not (False) stored successfully.

EndLoading

Method of VcGantt

This method indicates the finish of the loading procedure on the methods **InsertNodeRecord** and **InsertLinkRecord**, simultaneously triggering an update of the chart.

	Data Type	Explanation
Return value	System.Boolean	Loading was successfully finished

Example Code VB.NET

```
VcGantt1.EndLoading()
```

Example Code C#

```
vcGantt1.EndLoading();
```

ExportGraphicsToFileEx

Method of VcGantt

This method lets you store a Gantt diagram to a file without generating a **Save as** dialog box. Possible formats for saving:

- *.BMP (Microsoft Windows Bitmap)
- *.EMF (Enhanced Metafile or Enhanced Metafile Plus)
- *.GIF (Graphics Interchange Format)

- *.JPG (Joint Photographic Experts Group)
- *.PNG (Portable Network Graphics)
- *.TIF (Tagged Image File Format)
- *.VMF (Viewer Metafile)
- *.WMF (Microsoft Windows Metafile, probably with EMF included)

EMF, EMF+ VMF and WMF are vector formats that allow to store a file independent of pixel resolution. All other formats are pixel-oriented and confined to a limited resolution.

The VMF format basically has been deprecated, but it will still be supported for some time to maintain compatibility with existing applications.

When exporting to bitmap formats, setting 0 to the desired number of pixels of both, the x and the y direction, will keep the aspect ratio. If both pixel numbers equal 0, the size (in pixels) of the exported chart is calculated by VARCHART XGantt as listed below:

- PNG: a resolution of 100 dpi and a zoom factor of 100% are assumed. If alternatively a value of ≤ -50 is specified in the parameter SizeX, the absolute number will be used as DPI input. The number of DPIs will be stored to the PNG file, so with a given zoom factor display software can find the correct size for display.
- GIF, TIFF, BMP, JPEG: a resolution of 100 dpi and a zoom factor of 100% are assumed. If alternatively a value of ≤ -50 is specified in the parameter SizeX, the absolute number will be used as DPI input. In addition, an internal limit of 50 MBs of memory size is required for the uncompressed source bit map in the memory; so larger diagrams may have a smaller resolution than expected.

To formats of vector graphics, no pixel number can be set, but the below coordinate spaces:

- WMF: A fixed resolution is assumed where the longer side uses coordinates between 0 and 10,000 while the shorter side uses correspondingly smaller values to keep the aspect ratio.

- EMF/EMF+: The total resolution is adopted, using coordinates scaled by 1/100 mm in both, the x and y direction.

For further details on the different formats please read the chapter "Important Concepts: Graphics Formats".

	Data Type	Explanation
Parameter:		
⇒ fileName	System.String	File name (including a path, if necessary)
⇒ printOutputFormat	PrintOutputFormat	Format of the file to be stored.
	Possible Values: .vcBMP 2 .vcEMF 9 .vcEMFPlus 12 .vcEMFWithEMFPlusIncluded 11 .vcEPS 3 .vcGIF 4 .vcJPG 5 .vcPCX 6 .vcPNG 7 .vcTIF 8 .vcVMF 0 .vcWMF 1 .vcWMFWithEMFIncluded 10	File will be written in the format BMP. File will be written in the format EMF. File will be written in the format EMF+, the standard extension is EMF. File will be written in the format EMF, additionally including the format EMF+. The standard extension is EMF. Deprecated File will be written in the format GIF. File will be written in the format JPG. Deprecated File will be written in the format PNG. File will be written in the format TIF. File will be written in the format VMF. File will be written in the format WMF. File will be written in the format WMF, additionally including the format EMF. The standard extension is WMF.
⇒ SizeX	System.Int16	Width of the exported diagram in pixels. Available with pixel formats only. If this value is set to 0, its true size will be calculated from the aspect ratio.
⇒ SizeY	System.Int16	Height of the exported diagram in pixels. Available with pixel formats only. If this value is set to 0, its true size will be calculated from the aspect ratio.
Return value	System.Boolean	File was (true) / was not (false) stored successfully.

Example Code VB.NET

```
VcGantt1.ExportGraphicsToFileEx "C:\Tmp\test1.vmf", vcVMF, 0, 0
```

Example Code C#

```
vcGantt1.ExportGraphicsToFileEx(@"c:\Tmp\test.vmf",  
VcPrintOutputFormat.vcVMF, 0, 0);
```

FitChartIntoView

Method of VcGantt

This method allows you to adjust the diagram to the control size while keeping the width-to-height-ratio so that either the height or the width of the diagram is completely visible. The method returns the relative enlargement or reduction in percent * 1000.

Please see also the property **ZoomFactor** and the method **Zoom()** of VcGantt.

	Data Type	Explanation
Parameter: ⇒ fitMode	VcFitMode Possible Values: .vcFitHeight 23 .vcFitMaximumOfWidthAndHeight 1051 .vcFitMinimumOfWidthAndHeight 1052 .vcFitWidth 24 .vcUseLargerZoomFactor 1053 .vcUseSmallerZoomFactor 1054	Selection of zoom factor The diagram is adjusted height-wise to the control size. The largest dimension of the diagram is adjusted to the control size. The smallest dimension of the diagram is adjusted to the control size. The diagram is adjusted width-wise to the control size. The larger of the zoom factors is used. The corresponding dimension of the diagram does not fit into the frame of the control. The smaller of the zoom factors is used and the corresponding dimension of the diagram fits completely into the control.
Return value	System.Int32	Relative zoom factor

Example Code VB.NET

```
VcGantt1.(FitChartIntoView(VcFitMode.vcFitWidth))
```

Example Code C#

```
vcGantt1.FitChartIntoView(VcFitMode.vcFitWidth);
```

FitHistogramsIntoView

Method of VcGantt

This method matches the visible histograms of the Gantt object into a view. For this, the histograms are re-scaled proportionally, so that their size ratio is maintained.

	Data Type	Explanation
Return value	System.Boolean	The histograms had to (True) / did not have to (False) be re-scaled.

Example Code VB.NET

```
VcGantt1.FitHistogramsIntoView = True
```

Example Code C#

```
VcGantt1.FitHistogramsIntoView = true;
```

FitRangeIntoView

Method of VcGantt

This method lets you match an arbitrary section of the time scale into a window to make the section visible. The size of the time units displayed will change in accordance with the window size and the size of the section defined. The beginning and the end are set by the **startValue** and **endValue** parameter, respectively. The parameter **gapAsNoOfTimeUnits** lets you set the number of time units, by which the visible section is to differ from the date at the beginning of the section displayed and by which the true end of the time scale is to differ from the end of the section displayed. The time unit itself you can set on the **General** property page.

	Data Type	Explanation
Parameter:		
⇒ startDate	System.DateTime	Start date of the area to be matched
⇒ endDate	System.DateTime	End date of the area to be matched
⇒ gapAsNoOfTimeUnits	System.Int32	Number of time units to form the "gap" between startDate/endDate and the beginning of the visible section of the time scale start/end
Return value	System.Boolean	Area could/could not be matched.

Example Code VB.NET

```
VcGantt1.FitRangeIntoView("14.09.2014", "21.09.2014", 1)
```

Example Code C#

```
vcGantt1.FitRangeIntoView(Convert.ToDateTime("14.09.2014"),  
Convert.ToDateTime("21.09.2014"),1);
```

GetAValueFromARGB

Method of VcGantt

A color value is composed by four parts: A (alpha), R (red), G (green) and B (blue). A value of 0 in the alpha position will result in complete transparency whereas 255 represents a completely solid color. Ascending values of R, G and B show increasingly lightening colors, the ultimate values 0,0,0 and 255,255,255 representing black and white, respectively. This method retrieves the alpha value of an ARGB value.

	Data Type	Explanation
Parameter: ⇒ argb	System.Int32	ARGB value, from which the alpha value is to be identified
Return value	System.Int32	Alpha value returned

Example Code VB.NET

```
Dim alpha As Integer
Dim red As Integer
Dim green As Integer
Dim blue As Integer
Dim argb As Long
alpha = alpha + 11
red = red + 11
green = green + 11
blue = blue + 11
argb = VcGantt1.MakeARGB(alpha, red, green, blue)
alpha = VcGantt1.GetAValueFromARGB(argb)
```

Example Code C#

```
int alpha;
int red;
int green;
int blue;
long argb;
alpha = alpha + 11;
red = red + 11;
green = green + 11;
blue = blue + 11;
argb = VcGantt1.MakeARGB(alpha, red, green, blue);
alpha = VcGantt1.GetAValueFromARGB(argb);
```

GetBValueFromARGB

Method of VcGantt

A color value is composed by four parts: A (alpha), R (red), G (green) and B (blue). A value of 0 in the alpha position will result in complete transparency whereas 255 represents a completely solid color. Ascending values of R, G and B show increasingly lightening colors, the ultimate values 0,0,0 and

255,255,255 representing black and white, respectively. This method retrieves the "blue" value of an ARGB value.

	Data Type	Explanation
Parameter: ⇒ argb	System.Int32	ARGB value, from which the "blue" value is to be identified
Return value	System.Int32	"Blue" value returned

Example Code VB.NET

```
Dim alpha As Integer
Dim red As Integer
Dim green As Integer
Dim blue As Integer
Dim argb As Long
alpha = alpha + 11
red = red + 11
green = green + 11
blue = blue + 11
argb = VcGantt1.MakeARGB(alpha, red, green, blue)
blue = VcGantt1.GetBValueFromARGB(argb)
```

Example Code C#

```
int alpha;
int red;
int green;
int blue;
long argb;
alpha = alpha + 11;
red = red + 11;
green = green + 11;
blue = blue + 11;
argb = VcGantt1.MakeARGB(alpha, red, green, blue);
blue = VcGantt1.GetBValueFromARGB(argb);
```

GetCurrentComponentStart

Method of VcGantt

This method lets you retrieve the scroll value in 1/100 mm of a graphical element of the VARCHART XGantt control (time scale, diagram, histogram, table, table caption etc.) in any direction.

	Data Type	Explanation
Parameter: ⇐ component	VcComponentType Possible Values: .vcAdditionalListComponent 1 .vcBottomListTitleComponent 14 .vcBottomRightListTitleComponent 17 .vcBottomTimeScaleComponent 15 .vcDiagramComponent 4	Type of graphical element additional table bottom title bar bottom right table bottom time scale diagram

	.vcHistogramComponent 8 .vcHistogramVerScaleComponent 7 .vcLegendComponent 10 .vcListComponent 0 .vcListTitleComponent 2 .vcRightListComponent 5 .vcRightListTitleComponent 16 .vcTimeScaleComponent 3 .vcTopTitleComponent 11	histogram numeric scale (vertical histogram scale) legend (currently functionless; return values 00) table table title right table table title of right table upper time scale upper title bar
↔ scrollOrientation	VcScrollOrientation Possible Values: .vcHorizontal 1 .vcVertical 2	Direction of scrolling horizontal scrolling vertical scrolling
Return value	System.Int32	Scroll value in 1/100 mm

GetCurrentViewDates

Method of VcGantt

This method lets you retrieve the start and end dates of the visible section of the time scale.

	Data Type	Explanation
Parameter:		
↔ leftDate	System.DateTime	Start date of the visible section of the time scale
↔ rightDate	System.DateTime	End date of the visible section of the time scale
Return value	System.Boolean	Start/end dates of the visible section of the time scale are returned/not returned.

Example Code VB.NET

```
Dim bGetCurrentViewDates As Boolean
Dim leftDate As Date
Dim rightDate As Date
GetCurrentViewDates = VcGantt1.GetCurrentViewDates(leftDate, rightDate)
```

Example Code C#

```
DateTime leftDate = new DateTime();
DateTime rightDate = new DateTime();
bool bGetCurrentViewDates = vcGantt1.GetCurrentViewDates(ref leftDate, ref rightDate);
```

GetDate

Method of VcGantt

This method lets you retrieve the date that corresponds to a x coordinate in the diagram section.

	Data Type	Explanation
Parameter: ⇨ x	System.Int32	X coordinate in the Gantt diagram, the corresponding date of which is to be retrieved
Return value	System.DateTime	Date retrieved

Example Code VB.NET

```
Private Sub VcGantt1_VcDiagramLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcDiagramClickingEventArgs) Handles
VcGantt1.VcDiagramLeftClicking
    Label1.Name = VcGantt1.GetDate(e.X)
End Sub
```

Example Code C#

```
private void vcGantt1_VcDiagramLeftClicking(object sender,
NETRONIC.XGantt.VcDiagramClickingEventArgs e)
{
    label1.Text = vcGantt1.GetDate(e.X).ToString();
}
```

GetDateAsString

Method of VcGantt

Description:

	Data Type	Explanation
Parameter: ⇨ x	System.Int32	X coordinate in the Gantt diagram, the corresponding date of which is to be retrieved
Return value	System.String	Date retrieved

Example Code VB.NET

```
Private Sub VcGantt1_VcDiagramLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcDiagramClickingEventArgs) Handles
VcGantt1.VcDiagramLeftClicking

    MsgBox(VcGantt1.GetDateAsString(e.X))

End Sub
```

Example Code C#

```
private void vcGantt1_VcDiagramLeftClicking(object sender,
NETRONIC.XGantt.VcDiagramClickingEventArgs e)
{
    MessageBox.Show(vcGantt1.GetDateAsString(e.X));
}
```

GetGValueFromARGB**Method of VcGantt**

A color value is composed by four parts: A (alpha), R (red), G (green) and B (blue). A value of 0 in the alpha position will result in complete transparency whereas 255 represents a completely solid color. Ascending values of R, G and B show increasingly lightening colors, the ultimate values 0,0,0 and 255,255,255 representing black and white, respectively. This method retrieves the "green" value of an ARGB value.

	Data Type	Explanation
Parameter: ⇒ argb	System.Int32	ARGB value, from which the "green" value is to be identified
Return value	System.Int32	"Green" value returned

Example Code VB.NET

```
Dim alpha As Integer
Dim red As Integer
Dim green As Integer
Dim blue As Integer
Dim argb As Long
alpha = alpha + 11
red = red + 11
green = green + 11
blue = blue + 11
argb = VcGantt1.MakeARGB(alpha, red, green, blue)
green = VcGantt1.GetRValueFromARGB(argb)
```

Example Code C#

```
int alpha;
int red;
int green;
int blue;
long argb;
alpha = alpha + 11;
red = red + 11;
green = green + 11;
blue = blue + 11;
argb = VcGantt1.MakeARGB(alpha, red, green, blue);
green = VcGantt1.GetGValueFromARGB(argb);
```

GetLinkByID

Method of VcGantt

This method gives access to a link by its identification which was specified on the **Administrative Data Tables** dialog. If the identification consists of more than one field (composite primary key), the multipart ID has to be noted as shown below:

ID=ID1|ID2|ID3

	Data Type	Explanation
Parameter: ⇒ linkID	System.Object	Link identification
Return value	VcLink	Link

Example Code VB.NET

```
Dim link As VcLink
Dim successor As Integer
link = VcGantt1.GetLinkByID(" 1")
successor = link.DataField(2)
```

Example Code C#

```
VcLink link = vcGantt1.GetLinkByID(" 1");
int successor = Convert.ToInt32(link.get_DataField(2));
```

GetLinkByNodeIDs

Method of VcGantt

This method lets you access a link by the ID of its predecessor and successor node. If the identification consists of more than one field (composite primary key), the multipart ID has to be noted as shown below:

ID=ID1|ID2|ID3

	Data Type	Explanation
Parameter: ⇒ predecessorID	System.String	Identification of the predecessor node
⇒ successorID	System.String	Identification of the successor node
Return value	VcLink	Link

Example Code VB.NET

```
Dim link As VcLink
link = VcGantt1.GetLinkByNodeIDs(" 2", " 3")
```

Example Code C#

```
VcLink link = vcGantt1.GetLinkByNodeIDs(" 2", " 3");
```

GetNodeByID**Method of VcGantt**

This method lets you access a node by its identification which was specified on the **Administrative Data Tables** dialog. If the identification consists of more than one field (composite primary key), the multipart ID needs to be noted as shown below:

ID=ID1|ID2|ID3

	Data Type	Explanation
Parameter: ⇒ nodeID	System.Object	Node identification
Return value	VcNode	Node

Example Code VB.NET

```
Dim node As VcNode
node = VcGantt1.GetNodeByID("10")
```

Example Code C#

```
VcNode node = vcGantt1.GetNodeByID("10");
```

GetRValueFromARGB**Method of VcGantt**

A color value is composed by four parts: A (alpha), R (red), G (green) and B (blue). A value of 0 in the alpha position will result in complete transparency whereas 255 represents a completely solid color. Ascending values of R, G and B show increasingly lightening colors, the ultimate values 0,0,0 and 255,255,255 representing black and white, respectively. This method retrieves the "red" value of an ARGB value.

	Data Type	Explanation
Parameter: ⇒ argb	System.Int32	ARGB value, from which the "red" value is to be identified
Return value	System.Int32	"Red" value returned

Example Code VB.NET

```
Dim alpha As Integer
Dim red As Integer
Dim green As Integer
Dim blue As Integer
Dim argb As Long
alpha = alpha + 11
red = red + 11
green = green + 11
blue = blue + 11
argb = VcGantt1.MakeARGB(alpha, red, green, blue)
red = VcGantt1.GetRValueFromARGB(argb)
```

Example Code C#

```
int alpha;
int red;
int green;
int blue;
long argb;
alpha = alpha + 11;
red = red + 11;
green = green + 11;
blue = blue + 11;
argb = VcGantt1.MakeARGB(alpha, red, green, blue);
red = VcGantt1.GetRValueFromARGB(argb);
```

GetViewComponentSize

Method of VcGantt

This method lets you require at run time the size and position of a graphical element of the VARCHART XGantt control (time scale, diagram, histogram, table, table caption etc.) (see event **VcViewComponentsSizeModified**).

Note:

- 1. The position refers to the origin of the graphical element of the VARCHART XGantt control.
- 2. The values returned are pixel values.

	Data Type	Explanation
Parameter: ⇒ viewComponent	VcComponentType Possible Values: .vcAdditionalListComponent 1 .vcBottomListTitleComponent 14 .vcBottomRightListTitleComponent 17 .vcBottomTimeScaleComponent 15 .vcDiagramComponent 4 .vcHistogramComponent 8 .vcHistogramVerScaleComponent 7 .vcLegendComponent 10	Component type additional table bottom title bar bottom right table bottom time scale diagram histogram numeric scale (vertical histogram scale) legend (currently functionless; return values 00)

	.vcListComponent 0 .vcListTitleComponent 2 .vcRightListComponent 5 .vcRightListTitleComponent 16 .vcTimeScaleComponent 3 .vcTopTitleComponent 11	table table title right table table title of right table upper time scale upper title bar
⇐ x	System.Int32	X coordinate of the component
⇐ y	System.Int32	Y coordinate of the component
⇐ width	System.Int32	Component width
⇐ height	System.Int32	Component height
Return value	Void	

Example Code VB.NET

```
Private Sub handleHideHistogram()
    Dim x As Integer
    Dim y As Integer
    Dim width As Integer
    Dim height As Integer
    VcGantt1.GetViewComponentSize(VcComponentType.vcHistogramVerScaleComponent,
x, y, width, height)
    ' plus 6 because of the sash
    TextBox1.Top = VcGantt1.Top + y + 6
    TextBox1.Left = VcGantt1.Left + x
    ' minus 25 because of the numeric scale
    TextBox1.Width = width - 25
    ' minus 6 because of the sash
    TextBox1.Height = height - 6
End Sub
```

Example Code C#

```
private void handleHideHistogram()
{
    int x;
    int y;
    int width;
    int height;
    vcGantt1.GetViewComponentSize(VcComponentType.vcHistogramVerScaleComponent,
ref x, ref y, ref width, ref height);
    // plus 6 because of the sash
    textBox1.Top = vcGantt1.Top + y + 6;
    textBox1.Left = vcGantt1.Left + x;
    // minus 25 because of the numeric scale
    textBox1.Width = width - 25;
    // minus 6 because of the sash
    textBox1.Height = height - 6;
}
```

GroupNodes

Method of VcGantt

This methods lets you activate/deactivate the grouping. If you have set a grouping field by the **GroupingDataFieldIndex** property or if you have set the grouping order by the **GroupSortingDataFieldIndex** property, you need to activate the grouping by **GroupNodes**.

	Data Type	Explanation
Parameter: ⇒ onOff	System.Boolean	Grouping on/off
Return value	System.Boolean	Nodes were/were not grouped successfully.

Example Code VB.NET

```
VcGantt1.GroupingDataFieldIndex(0) = 11
VcGantt1.GroupSortingDataFieldIndex(0) = 12
```

```
VcGantt1.GroupNodes(True)
```

Example Code C#

```
vcGantt1.set_GroupingDataFieldIndex(0, 11);
vcGantt1.set_GroupSortingDataFieldIndex(0, 12);
```

```
vcGantt1.GroupNodes(true);
```

IdentifyField

Method of VcGantt

This method lets you identify the index of a data field the content of which is to be displayed in the table field at the given cursor position.

	Data Type	Explanation
Parameter: ⇒ x	System.Int32	X coordinate of the cursor
⇒ y	System.Int32	Y coordinate of the cursor
Return value	System.Int32	Data field index identified -1 if there is no table field at the given position

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles VcGantt1.VcNodeRightClicking
    Dim intField As Integer
    intField = VcGantt1.IdentifyField(e.X, e.Y)
    Label1.Text = e.Node.DataField(intField)
End Sub
```

Example Code C#

```

private void vcGantt1_VcNodeRightClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    int i = vcGantt1.IdentifyField(e.X, e.Y) ;

    if (i > 1) then

        Label1.Text =
Convert.ToString(e.Node.get_DataField(vcGantt1.IdentifyField(i)) ) ;
    }

```

IdentifyLayerAt**Method of VcGantt**

This method lets you identify a layer. If a node was identified by the method **IdentifyObjectAt**, you can use it as a reference object for identifying its layer at the same position by a call of **IdentifyLayerAt**.

	Data Type	Explanation
Parameter:		
⇒ x	System.Int32	X coordinate of the cursor
⇒ y	System.Int32	Y coordinate of the cursor
⇒ referenceNode	VcNode	Reference node
⇐ identifiedLayer	VcLayer	Layer identified
Return value	System.Boolean	Object identified/no object identified

Example Code VB.NET

```
Dim identifiedObj As Object
Dim identifiedObjType As VcObjectType
Dim identifiedLayer As VcLayer
Dim node As VcNode

VcGantt1.IdentifyObjectAt(e.X, e.Y, identifiedObj, identifiedObjType)
If identifiedObjType Is VcObjectType.vcObjTypeNodeInDiagram Then
    node = identifiedObj
End If
Point mousePos=VcGantt1.PointToClient(new Point(){X=mousePos.X, Y=mousePos.Y})

Select Case identifiedObjType
    Case VcObjectType.vcObjTypeNodeInDiagram
        VcGantt1.IdentifyLayerAt(X, Y, identifiedLayer, identifiedLayerType)
        If Not identifiedLayer Is Nothing Then
            MsgBox("The Node " + node.DataField(0) + " , Layer " +
identifiedLayer.Name + " , was identified in the diagram area.")
        Else
            MsgBox("The Node" + node.DataField(0) + " was identified in diagram
area; no layer was identified")
        End If
    Case VcObjectType.vcObjTypeNodeInTable
        MsgBox("The Node" + node.DataField(0) + " was identified via table")
    Case Else
        MsgBox("No node was identified")
End Select
```

Example Code C#

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using NETRONIC.XGantt;

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();

            using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using NETRONIC.XGantt;

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();

            private void vcGantt1_MouseDown(object sender, MouseEventArgs e)
            {
                object identifiedObj = null;
                VcObjectType identifiedObjType =
VcObjectType.vcObjTypeNodeInDiagram;
                VcLayer identifiedLayer = null;
                VcNode node;

                vcGantt1.IdentifyObjectAt(e.X, e.Y, ref identifiedObj, ref
identifiedObjType);
                //Please note: the .NET events DragOver, DragDrop and DragEnter will
return screen coordinates, not client coordinates.
                //In case you used those events before, you will need to convert
manually the coordinates returned into client coordinates:
                //Point mousePos=vcGantt1.PointToClient(new Point() {X=e.X, Y=e.Y});
                //vcGantt1.IdentifyObjectAt(mousePos.X, mousePos.Y, ref
identifiedObj, ref identifiedObjType);

                switch (identifiedObjType)
                {
                    case VcObjectType.vcObjTypeNodeInDiagram:
                        node = (VcNode)identifiedObj;
                        vcGantt1.IdentifyLayerAt(e.X, e.Y, node, ref
identifiedLayer);
                        if (identifiedLayer != null)

```

```

        MessageBox.Show("The Node " + node.get_DataField(0) + "
, Layer " + identifiedLayer.Name + ", was identified in the diagram area.");
        break;
    default:
        break;
    }
}
}
}
}
}

```

IdentifyObject

Method of VcGantt

This method lets you identify any object in VARCHART XGantt. The object type will be returned. When a node was identified by this method, you can use it as a reference object for identifying its layer at the same position by a second call of **IdentifyObject**.

If you use a development environment that always requires a reference to an object please use the method **IdentifyObjectAt** because in this method the parameter **reference object** is not needed.

	Data Type	Explanation
Parameter:		
⇒ x	System.Int32	X coordinate of the cursor
⇒ y	System.Int32	Y coordinate of the cursor
⇒ referenceObject	VcObject	Reference object that the ID refers to
⇐ identifiedObject	System.Object	Object identified
⇐ identifiedObjectType	VcObjectType	Type of the object identified
	Possible Values: .vcObjTypeBox 15 .vcObjTypeCalendarGrid 18 .vcObjTypeCurve 12 .vcObjTypeDateLine 9 .vcObjTypeGroup 7 .vcObjTypeGroupInDiagram 11 .vcObjTypeGroupInTable 7 .vcObjTypeHistogram 13 .vcObjTypeLayer 8 .vcObjTypeLinkCollection 3 .vcObjTypeNodeInDiagram 2 .vcObjTypeNodeInLegend 17 .vcObjTypeNodeInTable 1 .vcObjTypeNone 0 .vcObjTypeNumericScale 10 .vcObjTypeSummaryNode 14 .vcObjTypeTable 4 .vcObjTypeTableCaption 5 .vcObjTypeTimeScale 6	object type box object type calendar grid object type curve object type date line object type group object type group in diagram area object type group in table area object type histogram object type layer object type link collection object type node in diagram area object type node in legend area object type node in table area no object object type numeric scale object type summary bar object type table object type table caption object type time scale
Return value	System.Boolean	Object identified/no object identified

IdentifyObjectAt

Method of VcGantt

This method lets you identify any object in VARCHART XGantt. The object type will be returned. If a node was identified by this method, you can use it as a reference object for identifying its layer at the same position by a call of **IdentifyLayerAt**. If you want to identify a curve in a histogram you have to use the method **IdentifyObject**.

	Data Type	Explanation
Parameter:		
⇒ x	System.Int32	X coordinate of the cursor
⇒ y	System.Int32	Y coordinate of the cursor
⇐ identifiedObject	System.Object	Object identified
⇐ identifiedObjectType	VcObjectType	Type of the object identified
	Possible Values: .vcObjTypeBox 15 .vcObjTypeCalendarGrid 18 .vcObjTypeCurve 12 .vcObjTypeDateLine 9 .vcObjTypeGroup 7 .vcObjTypeGroupInDiagram 11 .vcObjTypeGroupInTable 7 .vcObjTypeHistogram 13 .vcObjTypeLayer 8 .vcObjTypeLinkCollection 3 .vcObjTypeNodeInDiagram 2 .vcObjTypeNodeInLegend 17 .vcObjTypeNodeInTable 1 .vcObjTypeNone 0 .vcObjTypeNumericScale 10 .vcObjTypeSummaryNode 14 .vcObjTypeTable 4 .vcObjTypeTableCaption 5 .vcObjTypeTimeScale 6	object type box object type calendar grid object type curve object type date line object type group object type group in diagram area object type group in table area object type histogram object type layer object type link collection object type node in diagram area object type node in legend area object type node in table area no object object type numeric scale object type summary bar object type table object type table caption object type time scale
Return value	System.Boolean	Object identified/no object identified

Example Code VB.NET

```

Private Sub VcGantt1_MouseMove(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles VcGantt1.MouseMove

    Dim identifiedObject As Object = Nothing
    Dim identifiedObjectType As VcObjectType = VcObjectType.vcObjTypeNone
    Dim node As VcNode = Nothing
    Dim identifiedLayer As VcLayer = Nothing

    VcGantt1.IdentifyObjectAt(e.X, e.Y, identifiedObject,
identifiedObjectType)

    Select Case identifiedObjectType
        Case VcObjectType.vcObjTypeNodeInDiagram
            node = identifiedObject

            VcGantt1.IdentifyLayerAt(e.X, e.Y, node, identifiedLayer)

            If identifiedLayer IsNot Nothing Then
                Labell1.Text = "X = " & e.X & "    Y = " & e.Y & vbCrLf & _
                    "Node ID = " & node.DataField(0) & vbCrLf & _
                    "Layer Name = " & identifiedLayer.Name
            End If

        Case Else
            Labell1.Text = ""
        End Select

End Sub

```

Example Code C#

```

private void VcGantt1_MouseMove(object sender, MouseEventArgs e)
{
    object identifiedObject = null;
    VcObjectType identifiedObjectType = VcObjectType.vcObjTypeNone;
    VcNode node = null;
    VcLayer identifiedLayer = null;

    VcGantt1.IdentifyObjectAt(e.X, e.Y, ref identifiedObject, ref
identifiedObjectType);

    switch (identifiedObjectType)
    {
        case VcObjectType.vcObjTypeNodeInDiagram:
            {
                node = (VcNode)identifiedObject;

                VcGantt1.IdentifyLayerAt(e.X, e.Y, node, ref
identifiedLayer);

                if (identifiedLayer != null)
                    labell1.Text = "X = " + e.X + "    Y = " + e.Y +
                        "\nNode ID = " + node.get_DataField(0) +
                        "\nLayer Name = " + identifiedLayer.Name;

                break;
            }
        default:
            {
                labell1.Text = "";
                break;
            }
    }
}

```

ImportConfiguration

Method of VcGantt

This method enables a configuration file (*.ini) to be loaded, which all settings are adopted from, including the corresponding data interface (*.ifd).

You may specify either a local file that includes the path or a URL.

Note: When loading a different configuration file, the current data will be lost and may have to be loaded anew.

	Data Type	Explanation
Parameter: ⇒ fileName	System.String	Name of file to be imported
Return value	Void	

Example Code VB.NET

```
VcGantt1.ImportConfiguration ( "c:\VARCHART\XGantt\sample.ini")
'or
VcGantt1.ImportConfiguration
("http://members.tripod.de/netronic_te/xgantt_sample.ini")
```

Example Code C#

```
vcGantt1.ImportConfiguration (@"c:\VARCHART\XGantt\sample.ini");
// or
vcGantt1.ImportConfiguration
(@"http://members.tripod.de/netronic_te/xgantt_sample.ini");
```

InitializeForWebService

Method of VcGantt

For internal use only.

	Data Type	Explanation
Return value	Void	

InsertLinkRecord

Method of VcGantt

This method lets you load the data of a link that connects two nodes. The data will be passed as a CSV string or as a data field in accordance with the structure defined in the **Administrative Data Tables** dialog in the **Relations**

table. The method **EndLoading** should be invoked when the process of loading (links and nodes) is completed.

	Data Type	Explanation
Parameter: ⇒ linkRecordContent	System.Object[]	Content of the link record
Return value	VcLink	Link

Example Code VB.NET

```
VcGantt1.InsertNodeRecord("A100;Activity 1;12.09.14;17.09.14;5;Planning")
VcGantt1.InsertNodeRecord("A105;Activity 5;13.09.14;18.09.14;7;Testing")
VcGantt1.InsertLinkRecord("1;A100;A105;FS;0")
VcGantt1.EndLoading()
```

Example Code C#

```
vcGantt1.InsertNodeRecord("A100;Activity 1;12.09.14;17.09.14;5;Planning");
vcGantt1.InsertNodeRecord("A105;Activity 5;13.09.14;18.09.14;7;Testing");
vcGantt1.InsertLinkRecord("1;A100;A105;FS;2");
vcGantt1.EndLoading();
```

InsertNodeRecord

Method of VcGantt

The data will be passed as a CSV string or as a data field in accordance with the structure defined in the **Administrate Data Tables** dialog in the **Maindata** table. The method **EndLoading** should be invoked when the process of loading (links and nodes) is completed.

	Data Type	Explanation
Parameter: ⇒ nodeRecordContent	Data field	Content of the node record
Return value	VcNode	Node

Example Code VB.NET

```
Dim nodeRecord As String
' data format: "Number;Name;Start date;Finish date;Group code;Group name"
nodeRecord = "A100;Activity 1;12.09.14;17.09.14;5;Planning"
VcGantt1.InsertNodeRecord(nodeRecord)
VcGantt1.EndLoading()

'or
Dim nodeRecord() As Object = New Object(5) {"A100", "Activity 1", "12.09.14",
"17.09.14", "5", "Planning"}
VcGantt1.InsertNodeRecord(nodeRecord)
VcGantt1.EndLoading()
```

Example Code C#

```
string nodeRecord = "A100;Activity 1;12.09.14;17.09.14;5;Planning";
vcGantt1.InsertNodeRecord(nodeRecord);
vcGantt1.EndLoading();
```

Load

Method of VcGantt

This method lets you load the records of the data tables of the selected file which had been saved earlier with the method **SaveAsEx(...)** in CSV format. The records are allocated to the corresponding data tables by using an appropriate identification line. CSV-Files may be retrieved and written in ANSI as well as in Unicode coding which is automatically recognized when read

```
**** table name ****
```

Example:

```
**** Maindata ****
1;Node 1;07.05.2007;;5
2;Node 2;14.05.2007;;5
3;Node 3;21.05.2007;;5
**** Relations ****
1;1;2
2;2;3
```

Records of non existing tables are ignored when read. The contents of the data tables is replaced completely.

	Data Type	Explanation
Parameter: ⇒ fileName	System.String	File name
Return value	System.Boolean	File was/was not opened successfully.

Example Code VB.NET

```
vcgantt1.Load("c:\Data\project1.bar")
```

Example Code C#

```
vcGantt1.Load(@"c:\Data\project1.bar");
```

MakeARGB

Method of VcGantt

This method lets you compose an ARGB value from the four single values of a color.

	Data Type	Explanation
Parameter:		
⇒ alpha	System.Int32	Alpha value
⇒ red	System.Int32	"Red" value
⇒ green	System.Int32	"Green" value
⇒ blue	System.Int32	"Blue" value
Return value	System.Int32	ARGB value returned

Example Code VB.NET

```
Dim alpha As Integer
Dim red As Integer
Dim green As Integer
Dim blue As Integer
Dim argb As Long
alpha = FF
red = A0
green = 34
blue = AB
argb = VcGantt1.MakeARGB(alpha, red, green, blue)
```

Example Code C#

```
long argb;
int alpha = FF;
int red = A0;
int green = 34;
int blue = AB;
argb = VcGantt1.MakeARGB(alpha, red, green, blue);
```

OptimizeTimeScaleStartEnd

Method of VcGantt

This method lets you define the start and the end date of the timescale so that all nodes are completely visible. The start and end date are set in dependency on the displayed nodes. The parameter **NoOfUnits** lets you specify by how many time units the scale is to start on the left before the earliest start and by how many time units it is to end on the right after latest finish of all activities. This property also can be set on the **General** property page.

	Data Type	Explanation
Parameter: ⇒ noOfUnits	System.Int16	Number of time units
Return value	System.Boolean	<p>Timescale was/was not optimized successfully.</p> <p>The return value is false if both TimeScaleStart and TimeScaleEnd have not been modified.</p> <p>If no activities exist, the return value is always false because there are no date modifications. The specified number of time units is meaningless in such cases.</p>

Example Code VB.NET

```
VcGantt1.OptimizeTimeScaleStartEnd(5)
```

Example Code C#

```
vcGantt1.OptimizeTimeScaleStartEnd(5);
```

PrintEx

Method of VcGantt

This method lets you print the diagram directly. A dialog box will not be displayed. If the printing was not successful the return value indicates the reason. This could be e.g. an entry in a log file.

	Data Type	Explanation			
Return value	VcPrintResultStatus	Possible values:			
		Name	parameter position	description	
		vcPrintingSucceeded	0	Printing was performed successfully.	
		vcNoPrinterInstalled	1	No printer was found	neither the one specified by the call VcPrinter.PrinterName nor the one labeled as default printer by the Windows operating system.
		vcPrintingAbortedByUser	2	Printing was aborted by the user.	
		vcPrintingAbortedByDriver	3	Printing was aborted by the Windows printer driver.	
		vcUnprintablePageLayout	4	Printing could not be performed since the page layout did not match the printer properties such as paper size or margins.	

Example Code C#

```
VcPrintResultStatus status = VcGantt1.PrintDirectEx();
if (status != VcPrintResultStatus.vcPrintingSucceeded)
    System.Diagnostics.Trace.WriteLine("Printing failed: "+status.ToString);
```

PrintToFile**Method of VcGantt**

This method lets you print the diagram directly into a file. Whether this is successful depends on the printer driver because many PDF printer drivers don't accept file names.

	Data Type	Explanation
Parameter:		
⇒ fileName	System.String	File name

Return value	Void	
---------------------	------	--

RecalculateAllStructureCodes

Method of VcGantt

By this method you can recalculate the structure code of the node hierarchy. The code is recalculated automatically after any modification. To avoid the recalculation for a set of actions, you can put them between the methods VcGantt.SuspendUpdate(true) and VcGantt.SuspendUpdate(false).

	Data Type	Explanation
Return value	Void	

Reset

Method of VcGantt

This methods lets you either delete objects (nodes, links, calendars etc.) from the diagram, the extent depending on the selected value of resetAction, or restore the settings of the property pages carried out at design time.

	Data Type	Explanation
Parameter: ⇒ resetAction	VcResetAction Possible Values: .vcEmptyAllDataTables 4 .vcReloadConfiguration 2 .vcRemoveGroups 0 .vcRemoveNodes 1	Objects to be initialized or deleted The contents of all data tables are deleted but the data tables are kept. Complete reinitialization with the INI-file. All settings and created objects expire. All groups and dependent objects and with that also all nodes and links are deleted. All nodes and dependent objects and, if necessary all links are deleted.
Return value	System.Boolean	The objects in the diagram were deleted successfully. {True}

Example Code VB.NET

```
VcGantt1.Reset (VcResetAction.vcRemoveNodes)
```

Example Code C#

```
vcGantt1.Reset (VcResetAction.vcRemoveNodes);
```


SaveAsEx

Method of VcGantt

This method lets you save the records of all data tables to a file of CSV format, using the structure defined on the property page **Data Tables** invoked by the property page **Objects**. Data tables that do not contain records will not be saved. If no file name was specified, the file most recently used by the **Open** method will be overwritten (corresponding to the common **Save** function).

	Data Type	Explanation
Parameter: ⇒ fileName ⇒ encoding	System.String VcEncoding Possible Values: .vcANSIEncoding 1 .vcUnicodeEncoding 2	Name of the file to be saved Mode of encoding If a file was saved in ANSI encoding, it depends on the local settings of the Windows operating system. The file then contains characters which can be read correctly only if the language settings are the same as the ones that it was stored by. Saving a file in Unicode encoding makes it independent of whatever settings and hence should be the preferred mode if possible. If a file that was saved in Unicode encoding is to be loaded in Visual Basic 6 independently of the VARCHART control, it has to be treated in a special way.
Return value	System.Boolean	File was/was not stored successfully.

Example Code VB.NET

```
VcGantt1.SaveAsEx ("C:\ProjectData.txt", VcEncoding.vcANSIEncoding)
```

Example Code C#

```
vcGantt1.SaveAsEx (@"C:\ProjectData.txt", VcEncoding.vcANSIEncoding);
```

ScheduleProject

Method of VcGantt

This method triggers a forward and a backward calculation of the current project. If you pass the start date, first a forward calculation will be performed, followed by a backward calculation. , first a backward calculation will be performed, followed by a forward calculation.

	Data Type	Explanation
Parameter: ⇒ startDate	System.DateTime	Start date

⇒ endDate	System.DateTime	End date
Return value	System.Boolean	Scheduling was/was not successful

Example Code VB.NET

```
' Vorwärtsberechnung (ASAP)
VcScheduler.ScheduleProject(2.5.2017, newDate(0))

' Rückwärtsberechnung (JIT)
VcScheduler.ScheduleProject(newDate(0), 2.5.2017)
```

Example Code C#

```
// Vorwärtsberechnung (ASAP)
vcScheduler.ScheduleProject(2.5.2017, newDate(0));

// Rückwärtsberechnung (JIT)
vcScheduler.ScheduleProject(newDate(0), 2.5.2017);
```

ScrollComponentStartTo

Method of VcGantt

This method lets you scroll a graphical element of the VARCHART XGantt control (time scale, diagram, histogram, table, table caption etc.) in any direction to the indicated scroll value (the start coordinate) in 1/100 mm.

	Data Type	Explanation
Parameter: ⇐ component	VcComponentType Possible Values: .vcAdditionalListComponent 1 .vcBottomListTitleComponent 14 .vcBottomRightListTitleComponent 17 .vcBottomTimeScaleComponent 15 .vcDiagramComponent 4 .vcHistogramComponent 8 .vcHistogramVerScaleComponent 7 .vcLegendComponent 10 .vcListComponent 0 .vcListTitleComponent 2 .vcRightListComponent 5 .vcRightListTitleComponent 16 .vcTimeScaleComponent 3 .vcTopTitleComponent 11	Type of graphical element additional table bottom title bar bottom right table bottom time scale diagram histogram numeric scale (vertical histogram scale) legend (currently functionless; return values 00) table table title right table table title of right table upper time scale upper title bar
⇐ scrollOrientation	VcScrollOrientation Possible Values: .vcHorizontal 1 .vcVertical 2	Direction of scrolling horizontal scrolling vertical scrolling
Return value	System.Boolean	Desired scroll value is/is not returned

ScrollToDate

Method of VcGantt

This method allows you to scroll to a particular date in the time scale. The **gapAsNoOfTimeUnits** parameter sets the number of time units that the gap between the specified date and the left or right edge of the timescale consists of (**vcLeftAligned** or **vcRightAligned**). By the parameter **horAlignment** you can specify if the date is to occur on the left or on the right side of the visible section of the timescale.

The time unit can be set on the **General** property page.

N.B: In case workfree times were collapsed, the collapsed times will be included in time calculations correctly, but they will not be displayed, which may lead to a seeming deviation from the values set.

	Data Type	Explanation
Parameter:		
⇒ date	System.DateTime	Date
⇒ horAlignment	VcHorizontalAlignment	Horizontal alignment
	Possible Values: .vcHorCenterAligned - 1 .vcLeftAligned -3 .vcRightAligned -2	horizontally centered left aligned right aligned
⇒ gapAsNoOfTimeUnits	System.Int32	Number of time units
Return value	System.Boolean	Scrolling was/was not performed successfully.

Example Code VB.NET

```
VcGantt1.ScrollToDate("20.10.14", VcHorizontalAlignment.vcLeftAligned, 2)
```

Example Code C#

```
vcGantt1.ScrollToDate(Convert.ToDateTime("20.10.14"),  
VcHorizontalAlignment.vcRightAligned, 2);
```

ScrollToGroupLine

Method of VcGantt

This method allows to scroll to the row containing a particular group node and to specify whether that group node should be displayed at the top, in the center or at the bottom of the screen.

	Data Type	Explanation
Parameter:		
⇒ group	VcGroup	Group to be scrolled to
⇒ verAlignment	VcVerticalAlignment	Vertical alignment
	Possible Values: .vcBottomAligned 2 .vcTopAligned 1 .vcVerCenterAligned - 1	bottom aligned top aligned vertically centered
Return value	System.Boolean	Scrolling was (true) / was not (false) performed successfully.

ScrollToNode

Method of VcGantt

This method allows to scroll to a particular node and to specify whether that node should be displayed at the top, in the center or at the bottom of the screen.

	Data Type	Explanation
Parameter:		
⇒ node	VcNode	Node
⇒ verAlignment	VcVerticalAlignment	Vertical alignment
	Possible Values: .vcBottomAligned 2 .vcTopAligned 1 .vcVerCenterAligned - 1	bottom aligned top aligned vertically centered
Return value	System.Boolean	Scrolling was/was not performed successfully.

Example Code VB.NET

```
Dim node As VcNode
node = VcGantt1.GetNodeByID(" 2")
VcGantt1.ScrollToNodeLine(node, VcVerticalAlignment.vcVerCenterAligned)
```

Example Code C#

```

object[] objDataRecord = new object[5];

vcGantt1.ExtendedDataTablesEnabled = true;
vcGantt1.MinimumRowHeight = 1000;

vcGantt1.TimeScaleEnd = new DateTime(2010, 8, 1);
vcGantt1.TimeScaleStart = new DateTime(2010, 6, 1);

objDataRecord[2] = new DateTime(2010, 6, 3);
objDataRecord[3] = new DateTime(2010, 6, 10);
objDataRecord[4] = 5;

VcDataRecordCollection dataRecordCol =
vcGantt1.DataTableCollection.DataTableByName("Maindata").DataRecordCollection;
for (int i = 1; i < 100; i++)
{
    objDataRecord[0] = i;
    objDataRecord[1] = "Node " + i.ToString();

    dataRecordCol.Add(objDataRecord);
}
vcGantt1.EndLoading();
vcGantt1.ScrollToNode(vcGantt1.GetNodeByID("50"),
VcVerticalAlignment.vcTopAligned);

```

ScrollToNodeLine**Method of VcGantt**

This method allows to scroll to the row containing a particular node and to specify whether that node should be displayed at the top, in the center or at the bottom of the screen.

Note: If you choose the option **In one line**, all activities in a group will be displayed in one line. If the activities in the group coincide, they will be automatically displayed underneath one another in expanded mode to prevent overlapping. In this case using the **ScrollToNodeLine** method scrolls to the appropriate group row containing the selected node. Then it may happen that the selected node is not displayed in the center of the screen and is not visible.

	Data Type	Explanation
Parameter:		
⇒ node	VcNode	Node to be scrolled to
⇒ verAlignment	VcVerticalAlignment	Vertical alignment
	Possible Values: .vcBottomAligned 2 .vcTopAligned 1 .vcVerCenterAligned - 1	bottom aligned top aligned vertically centered
Return value	System.Boolean	Scrolling was (true) / was not (false) performed successfully.

Example Code VB.NET

```
Imports NETRONIC.XGantt
...
Dim i As Integer
Dim objDataRecord(2) As Object
Dim dataRecordCol As VcDataRecordCollection

VcGantt1.ExtendedDataTablesEnabled = True
dataRecordCol =
VcGantt1.DataTableCollection.DataTableByName("Maindata").DataRecordCollection
    For i = 1 To 100
        objDataRecord(0) = i
        objDataRecord(1) = "Node " + i.ToString()
        dataRecordCol.Add(objDataRecord)
    Next
VcGantt1.EndLoading()
VcGantt1.ScrollToNodeLine(VcGantt1.GetNodeByID("50"),
VcVerticalAlignment.vcTopAligned)
```

Example Code C#

```
using NETRONIC.XGantt;
...
object[] objDataRecord = new object[2];
vcGantt1.ExtendedDataTablesEnabled = true;
VcDataRecordCollection dataRecordCol =
vcGantt1.DataTableCollection.DataTableByName("Maindata").DataRecordCollection;
    for (int i = 1; i < 100; i++)
    {
        objDataRecord[0] = i;
        objDataRecord[1] = "Node " + i.ToString();
        dataRecordCol.Add(objDataRecord);
    }
vcGantt1.EndLoading();
vcGantt1.ScrollToNodeLine(vcGantt1.GetNodeByID("50"),
VcVerticalAlignment.vcTopAligned);
```

SetImageResource**Method of VcGantt**

With this method, a specified name can be assigned at runtime to an image object already existing in the application. The method provides an alternative to the one available so far, where the image name specified on the XGantt property pages always leads to reading the image object out of the addressed file. It should be carried out when starting the application, for instance in the method **Form_Load**. The image names can be chosen at will. To distinguish the file names, characters that are otherwise forbidden in file names, can be used, e.g, the asterisk (*). All image objects are permitted: bitmaps (BMP, JPG, GIF, PNG, TIFF) and metafiles (WMF, EMF). If the parameter **image** is set to "null", all former assignments are cancelled.

Example: Add an image or file resource (in Visual Studio in the Project Properties: **Resources/Add Resource/Add Existing File**) and enter a code line in **Form_Load** like the one shown below:

For bitmap resources:

```
vcGantt1.SetImageResource("*PlusImage",  
<namespace>.Properties.Resources.plusImage);
```

For metafile resources:

```
vcGantt1.SetImageResource("*MinusImage", new Metafile(new  
memoryStream(<namespace>.Properties.Resources.minusImage)));
```

	Data Type	Explanation
Parameter: ⇒ imageName	System.String	Name assigned to image
Return value	System.Drawing.Image	image

Example Code C#

```
vcGantt1.SetImageResource("*PlusImage",  
<namespace>.Properties.Resources.plusImage);
```

ShowAboutDialog

Method of VcGantt

This method lets you open the **About** box. It contains an overview of the program and the library files currently used with the absolute path and version numbers. This feature makes the hotline support more comfortable. The overview can be selected with the help of the mouse and copied by Ctrl+C and inserted by Ctrl+V into a mail.

	Data Type	Explanation
Return value	Void	

Example Code VB.NET

```
VcGantt1.ShowAboutDialog()
```

Example Code C#

```
vcGantt1.ShowAboutDialog();
```

ShowEditGroupDialog

Method of VcGantt

This method invokes the **Edit Group data** dialog box for the group passed.

	Data Type	Explanation
Parameter: ⇒ group	VcGroup	group whose data are to be edited
Return value	System.Boolean	group data were edited/editing was cancelled.

ShowExportGraphicsDialog

Method of VcGantt

This method lets you invoke the **Save As** dialog for saving the diagram. Possible formats for saving:

- *.BMP (Microsoft Windows Bitmap)
- *.EMF (Enhanced Metafile or Enhanced Metafile Plus)
- *.GIF (Graphics Interchange Format)
- *.JPG (Joint Photographic Experts Group)
- *.PNG (Portable Network Graphics)
- *.TIF (Tagged Image File Format)
- *.VMF (Viewer Metafile)
- *.WMF (Microsoft Windows Metafile, probably with EMF included)

EMF, EMF+, VMF and WMF are vector formats that allow to store a file independent of pixel resolution. All other formats are pixel-oriented and confined to a limited resolution.

The VMF format basically has been deprecated, but it will still be supported for some time to maintain compatibility with existing applications.

Further details on the different formats please find in the chapter **Important Concepts: Graphics Formats**.

When exporting, the size of the exported diagram will be calculated this way:

- PNG: a resolution of 100 dpi and a zoom factor of 100% are assumed. If alternatively a value of ≤ -50 is specified in the parameter SizeX, the absolute number will be used as DPI input.
- GIF, TIFF, BMP, JPEG: a resolution of 100 dpi and a zoom factor of 100% are assumed. If alternatively a value of ≤ -50 is specified in the parameter SizeX, the absolute number will be used as DPI input. In addition, an internal limit of 50 MBs of memory size is required for the uncompressed source bit map in the memory; so larger diagrams may have a smaller resolution than expected.
- WMF: A fixed resolution is assumed where the longer side uses coordinates between 0 and 10,000 while the shorter side uses correspondingly smaller values to keep the aspect ratio.
- EMF/EMF+: The total resolution is adopted, using coordinates scaled by 1/100 mm.

	Data Type	Explanation
Return value	System.Boolean	Graphics successfully (true) /not successfully (false) exported

Example Code VB.NET

```
VcGantt1.ShowExportGraphicsDialog()
```

Example Code C#

```
vcGantt1.ShowExportGraphicsDialog();
```

ShowLinkEditDialog

Method of VcGantt

This method invokes the **Edit Link** dialog box for the link passed.

	Data Type	Explanation
Parameter: ⇒ link	VcLink	Link the data of which are to be edited
Return value	System.Boolean	Link data were edited/edition was cancelled.

Example Code VB.NET

```

Private Sub VcGantt1_VcLinksLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcLinksClickingEventArgs) Handles VcGantt1.VcLinksLeftClicking
    Dim linkCltn As VcLinkCollection
    Dim link As VcLink
    linkCltn = VcGantt1.LinkCollection
    If linkCltn.Count > 0 Then
        For Each link In linkCltn
            VcGantt1.ShowLinkEditDialog(link)
        Next
    End If
End Sub

```

Example Code C#

```

private void vcGantt1_VcLinksLeftClicking(object sender,
NETRONIC.XGantt.VcLinksClickingEventArgs e)
{
    if (e.LinkCollection.Count > 0)
    {
        foreach (VcLink link in e.LinkCollection)
            vcGantt1.ShowLinkEditDialog(link);
    }
}

```

ShowNodeEditDialog

Method of VcGantt

This method invokes the **Edit Data** dialog box for the node passed.

	Data Type	Explanation
Parameter: ⇒ node	VcNode	Node whose data are to be edited
Return value	System.Boolean	Node data were edited./Editing was cancelled.

Example Code VB.NET

```

Private Sub VcGantt1_VcNodeLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles VcGantt1.VcNodeLeftClicking
    VcGantt1.ShowNodeEditDialog(node)
End Sub

```

Example Code C#

```

private void vcGantt1_VcNodeLeftClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    vcGantt1.ShowNodeEditDialog(e.Node);
}

```

ShowPageSetupDialog

Method of VcGantt

This method lets you invoke the **Page Setup** dialog.

	Data Type	Explanation
Return value	System.Boolean	Dialog box was/was not opened successfully.

Example Code VB.NET

```
VcGantt1.ShowPageSetupDialog()
```

Example Code C#

```
vcGantt1.ShowPageSetupDialog();
```

ShowPrintDialog

Method of VcGantt

This method triggers the printing of the diagram. The Windows **Print** dialog will open, using the parameters defined in the **ShowPageSetupDialog**.

	Data Type	Explanation
Return value	System.Boolean	Chart was/was not printed successfully.

Example Code VB.NET

```
VcGantt1.ShowPrintDialog()
```

Example Code C#

```
vcGantt1.ShowPrintDialog();
```

ShowPrinterSetupDialog

Method of VcGantt

This method lets you invoke the Windows **Print Setup** dialog box.

	Data Type	Explanation
Return value	System.Boolean	Dialog box was/was not opened successfully.

Example Code VB.NET

```
VcGantt1.ShowPrinterSetupDialog()
```

Example Code C#

```
vcGantt1.ShowPrinterSetupDialog();
```

ShowPrintPreviewDialog

Method of VcGantt

This method invokes the print preview.

	Data Type	Explanation
Return value	System.Boolean	Dialog box was/was not opened successfully.

Example Code VB.NET

```
VcGantt1.ShowPrintPreviewDialog()
```

Example Code C#

```
vcGantt1.ShowPrintPreviewDialog();
```

SortGroups

Method of VcGantt

This method lets you start the sorting of groups in a grouped diagram in accordance with the defined sorting parameter **GroupSortingDataFieldIndex (GroupingLevel)**.

	Data Type	Explanation
Return value	System.Boolean	Groups were/were not sorted successfully.

Example Code VB.NET

```
VcGantt1.GroupSortingDataFieldIndex(0) = 12
VcGantt1.SortGroups()
```

Example Code C#

```
vcGantt1.set_GroupSortingDataFieldIndex(0,12);
vcGantt1.SortGroups();
```

SortNodes

Method of VcGantt

This method lets you start the sorting of the activities in accordance with the defined sorting parameters (**NodeSortingDataFieldIndex (sortLevel)** and **NodeSortingOrder (sortLevel)**). If a grouping is activated, the sorting will be done separately for each group.

	Data Type	Explanation
Return value	System.Boolean	Nodes were/were not sorted successfully.

Example Code VB.NET

```
VcGantt1.NodeSortingDataFieldIndex(0) = 3
VcGantt1.NodeSortingOrder(0) = VcNodeSortingOrder.vcAscending
VcGantt1.SortNodes()
```

Example Code C#

```
vcGantt1.set_NodeSortingDataFieldIndex(0,3);
vcGantt1.set_NodeSortingOrder(0, VcNodeSortingOrder.vcAscending);
vcGantt1.SortNodes();
```

SuspendUpdate

Method of VcGantt

For projects comprising many nodes, updating procedures may be very time consuming if actions are repeated by each node. You can accelerate the updating procedure by using the **SuspendUpdate** method. Bracket the code that describes the repeated action between **SuspendUpdate (True)** and **SuspendUpdate (False)** as in the below code example. This will get the nodes to be updated all at once and improve the performance.

	Data Type	Explanation
Parameter: ⇒ suspendFlag	System.Boolean	SuspendUpdate(True): Start of the SuspendUpdate method/ SuspendUpdate(False): end of the SuspendUpdate method
Return value	Void	

Example Code VB.NET

```

VcGantt1.SuspendUpdate(True)

If updateFlag Then
    For Each node In nodeCltn
        If (node.DataField(2) < "07.09.98") Then
            node.DataField(13) = "X"
            node.Update()
            counter = counter + 1
        End If
    Next
Else
    For Each node In nodeCltn
        If (node.DataField(2) < "07.09.98") Then
            node.DataField(13) = ""
            node.Update()
            counter = counter + 1
        End If
    Next
End If

VcGantt1.SuspendUpdate(False)

```

Example Code C#

```

bool updateFlag = true;
VcNodeCollection nodeCltn = vcGantt1.NodeCollection;
int counter = 0;

vcGantt1.SuspendUpdate(true);
if (updateFlag == true)
{
    foreach (VcNode node in nodeCltn)
    {
        if (DateTime.Compare(Convert.ToDateTime(node.get_DataField(2)),
Convert.ToDateTime("12.09.07")) < 0)
        {
            node.set_DataField(13, "X");
            node.Update();
            counter = counter + 1;
        }
    }
}
else
{
    foreach (VcNode node in nodeCltn)
    {
        if (DateTime.Compare(Convert.ToDateTime(node.get_DataField(2)),
Convert.ToDateTime("12.09.07")) < 0)
        {
            node.set_DataField(13, "");
            node.Update();
            counter = counter + 1;
        }
    }
}
vcGantt1.SuspendUpdate(false);

```

UpdateLinkRecord

Method of VcGantt

This method lets you modify the data of an existing link record. The link record will be identified by the primary key set in the **Administrate Data Tables** dialog. This method is used when external modifications of link data have to be carried out by the diagram. If the link updated does not exist, it will be generated.

	Data Type	Explanation
Parameter: ⇒ linkRecordContent	System.Object	Content of the link record
Return value	VcLink	Link updated

Example Code VB.NET

```
VcGantt1.UpdateLinkRecord("A100;A105;FS;0")
```

Example Code C#

```
vcGantt1.UpdateLinkRecord("1;A100;A105;FS;0");
```

UpdateNodeRecord

Method of VcGantt

This method lets you modify the data of an existing node record. The node record will be identified by the primary key defined in the **Administrate Data Tables** dialog. This method is used when external modifications of the data have to be carried out by the diagram.

	Data Type	Explanation
Parameter: ⇒ nodeRecordContent	System.Object	Content of the node record
Return value	VcNode	Node record was/was not updated successfully.

Example Code VB.NET

```
VcGantt1.UpdateNodeRecord("A100;Activity 1;12.09.14;17.09.14;5;Planning")
```

Example Code C#

```
vcGantt1.UpdateNodeRecord("A100;Activity 1;12.09.07;17.09.07;5;Planning");
```

UpdateRowNumberFields

Method of VcGantt

This method updates the field that stores the row number of the node. This field you can select on the **Nodes** property page from the **Row number field** combo box. Using this method is useful only if neither a hierarchical arrangement nor grouping are applied.

	Data Type	Explanation
Return value	Void	

Example Code VB.NET

```
VcGantt1.UpdateRowNumberFields()
VcGantt1.SaveAs("c:\tmp\data.bar")
```

Example Code C#

```
vcGantt1.UpdateRowNumberFields();
vcGantt1.SaveAs(@"c:\tmp\data.bar");
```

Zoom

Method of VcGantt

This method lets you enlarge/reduce the diagram on screen by the percentage specified (enlarging the diagram: zoom factor > 100, reducing the diagram: zoom factor < 100).

Please also see the VcGantt method **FitChartIntoView()** and the property **ZoomFactor**.

	Data Type	Explanation
Parameter: ⇒ zoomFactor	System.Int16	relative zoom factor {11...999}, other values will remain unconsidered
Return value	System.Boolean	Zooming was/was not performed successfully.

Example Code VB.NET

```
VcGantt1.Zoom(120)
```

Example Code C#

```
vcGantt1.Zoom(120);
```


Events

KeyDown

Event of VcGantt

This event occurs when the user presses a key while VARCHART XGantt has the focus. With the help of the key events you can trigger VARCHART Windows Forms functions by the keyboard. (For the interpretation of ANSI symbols please use the KeyPress event.)

	Data Type	Explanation
Properties:		
⇒ keyCode	System.Int16	Key code, e.g. vbKeyF1 (F1 key) or vbKeyHome (POS1 key)
⇒ shift	System.Int16	An integer that corresponds to the status of the keys Shift, Ctrl and Alt at the moment when the event occurs. The shift parameter is a bit field that may hold the values: Shift key (Bit 0), Ctrl key (Bit 1) and Alt key (Bit 2). These bits correspond to the value 1, 2 and 4. Several, all or none of these bits can be set to indicate that several, all or no keys are pressed. If the Ctrl key and the Alt key are pressed, the value of shift will be 6.

Example Code VB.NET

```
Private Sub VcGantt1_KeyDown(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyEventArgs) Handles VcGantt1.KeyDown
    MsgBox("key pressed")
End Sub
```

Example Code C#

```
private void vcGantt1_KeyDown(object sender, System.Windows.Forms.KeyEventArgs
e)
{
    MessageBox.Show("key pressed");
}
```

KeyPress

Event of VcGantt

This event occurs when the user presses and releases an ANSI key while VARCHART XGantt has the focus. With the help of the key events you can trigger VARCHART Windows Forms functions by the keyboard.

	Data Type	Explanation
Properties: ⇒ keyAscii	System.Int16	An integer that returns the numerical key code of an default ANSI key. KeyAscii is returned as reference. If the parameter will be changed, another symbol will be returned to the object. If KeyAscii is set to 0, pressing a key will have no effect, i.e. no symbol will be passed to the object.

Example Code VB.NET

```
Private Sub VcGantt1_KeyPress(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyPressEventArgs) Handles VcGantt1.KeyPress
    MsgBox("key pressed and released")
End Sub
```

Example Code C#

```
private void vcGantt1_KeyPress(object sender,
System.Windows.Forms.KeyPressEventArgs e)
{
    MessageBox.Show("key pressed and released");
}
```

KeyUp

Event of VcGantt

This event occurs when the user releases a key while VARCHART XGantt has the focus. By using the key events you can trigger VARCHART Windows Forms functions by the keyboard. (For the interpretation of ANSI symbols please use the KeyPress event.)

	Data Type	Explanation
Properties: ⇒ keyCode	System.Int16	Key code, e.g. vbKeyF1 (F1 key) or vbKeyHome (POS1 key)
⇒ shift	System.Int16	An integer that corresponds to the status of the keys Shift, Ctrl and Alt at the moment when the event occurs. The shift parameter is a bit field that may hold the values: Shift key (Bit 0), Ctrl key (Bit 1) and Alt key (Bit 2). These bits correspond to the value 1, 2 and 4. Several, all or none of these bits can be set to indicate that several, all or no keys are pressed. If the Ctrl key and the Alt key are pressed, the value of Shift will be 6.

Example Code VB.NET

```
Private Sub VcGantt1_KeyUp(ByVal sender As Object, ByVal e As
System.Windows.Forms.KeyEventArgs) Handles VcGantt1.KeyUp
    MsgBox("key released")
End Sub
```

Example Code C#

```
private void vcGantt1_KeyUp(object sender, System.Windows.Forms.KeyEventArgs e)
{
    MessageBox.Show("key released");
}
```

VcBoxCreated**Event of VcGantt**

This event occurs when the interactive creation of a box is completed. The box object is returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcBoxCreatedEventArgs	Object specific to the event that is being handled

Properties of the VcBoxCreatedEventArgs object

	Data Type	Explanation
Properties:		
⇒ box	VcBox	Box created

VcBoxCreating**Event of VcGantt**

This event occurs when the user creates a box.

The data passed by this event can be read, but must not be modified. For modifying them please use the event **VcBoxCreated**.

By setting the return status the create operation can be inhibited.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcBoxCreatingEventArgs	Object specific to the event that is being handled

Properties of the VcBoxCreatingEventArgs object

	Data Type	Explanation
Properties:		
⇒ xOffset	System.Int32	X position of the box
⇒ yOffset	System.Int32	Y position of the box
⇒ width	System.Int32	Width of the box
⇒ height	System.Int32	Height of the box
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The box will not be created. The box will be created.

VcBoxLeftClicking

Event of VcGantt

This event occurs when the user clicks the left mouse button on a box. The box object hit and the position of the mouse (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcBoxClickingEventArgs	Object specific to the event that is being handled

Properties of the VcBoxClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ box	VcBox	Box hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcBoxLeftClicking(ByVal sender As Object, ByVal e As
NETRONICXGantt.VcBoxClickingEventArgs) Handles VcGantt1.VcBoxLeftClicking
    TextBox1.Text = e.Box.FieldText(1)
End Sub
```

Example Code C#

```
private void vcGantt1_VcBoxLeftClicking(object sender,
VcGanttLibVcBoxClickingEventArgs e)
{
    textBox1.Text = e.Box.get_FieldText(1);
}
```

VcBoxLeftDoubleClicking**Event of VcGantt**

This event occurs when the user double-clicks the left mouse button on a box. The VcBox object hit and the mouse position (x,y-coordinates) are handed over as parameters.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcBoxClickingEventArgs	Object specific to the event that is being handled

Properties of the VcBoxClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ box	VcBox	Box hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcBoxLeftDoubleClicking(ByVal sender As Object, ByVal e As
NETRONICXGantt.VcBoxClickingEventArgs) Handles VcGantt1.VcBoxLeftDoubleClicking
    e.Box.FieldText(0) = TextBox1.Text
End Sub
```

Example Code C#

```
private void vcGantt1_VcBoxLeftDoubleClick(object sender,
VcGanttLib.VcBoxClickingEventArgs e)
{
    e.Box.set_FieldText(1, textBox1.Text);
}
```

VcBoxModified

Event of VcGantt

This event occurs when the modification of the box is finished. The Box object modified and the modification type are passed as parameters.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcBoxModifiedEventArgs	Object specific to the event that is being handled

Properties of the VcBoxModifiedEventArgs object

	Data Type	Explanation
Properties:		
⇒ box	VcBox	Box modified
⇒ modificationType	VcBoxModificationTypes	Modification type
	Possible Values:	
	.vcBMTAnchoringModified 16	Anchoring of the box modified
	.vcBMTAnything 1	any modification
	.vcBMTNothing 0	no modification
	.vcBMTSizeModified 8	Size of the box modified
	.vcBMTTextModified 4	text modified
	.vcBMTXYOffsetModified 2	Offset modified

Example Code VB.NET

```
Private Sub VcGantt1_VcBoxModified(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcBoxModifiedEventArgs) Handles VcGantt1.VcBoxModified
    MsgBox("The box has been modified")
End Sub
```

Example Code C#

```
private void vcGantt1_VcBoxModified(object sender,
NETRONIC.XGantt.VcBoxModifiedEventArgs e)
{
    MessageBox.Show("The box has been modified");
}
```

VcBoxModifying

Event of VcGantt

This event occurs when the user has modified a box interactively. The Box object modified and the modification type are passed as parameters.

The data passed by this event can be read, but must not be modified. For modifying them please use the event **VcBoxModified**.

By setting the return status the modification can be inhibited.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcBoxModifyingEventArgs	Object specific to the event that is being handled

Properties of the VcBoxModifyingEventArgs object

	Data Type	Explanation
Properties:		
⇒ box	VcBox	Box modified
⇒ modificationType	VcBoxModificationTypes	Modification type
	Possible Values: .vcBMTAnchoringModified 16 .vcBMTAnything 1 .vcBMTNothing 0 .vcBMTSizeModified 8 .vcBMTTextModified 4 .vcBMTXYOffsetModified 2	Anchoring of the box modified any modification no modification Size of the box modified text modified Offset modified
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The modification will be revoked. The modification will be accepted.

Example Code VB.NET

```
Private Sub VcGantt1_VcBoxModifying(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcBoxModifyingEventArgs) Handles VcGantt1.VcBoxModifying
    Select Case e.ModificationType
        Case VcBoxModificationTypes.vcBMTAnything :
            MsgBox("Box modification")
        Case VcBoxModificationTypes.vcBMTXYOffsetModified :
            MsgBox("Offset changed")
        Case VcBoxModificationTypes.vcBMTTextModified :
            MsgBox("Box field text changed")
    End Select
End Sub
```

Example Code C#

```
private void vcGantt1_VcBoxModifying(object sender,
NETRONIC.XGantt.VcBoxModifyingEventArgs e)
{
    switch(e.ModificationType)
    {
        case VcBoxModificationTypes.vcBMTAnything:
            MessageBox.Show("Box modification");
            break;
        case VcBoxModificationTypes.vcBMTXOffsetModified:
            MessageBox.Show("Offset changed");
            break;
        case VcBoxModificationTypes.vcBMTTextModified:
            MessageBox.Show("Box field text changed");
            break;
    }
}
```

VcBoxRightClicking**Event of VcGantt**

This event occurs when the user clicks the right mouse button on the box. The box object and the position of the mouse (x,y-coordinates) are returned. By setting the return status you can inhibit the integrated context menu to pop up or replace it by a context menu of your own at the location delivered.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcBoxClickingEventArgs	Object specific to the event that is being handled

Properties of the VcBoxClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ box	VcBox	Box hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values:	
	.vcRetStatNoPopup 4	The context menu will be inhibited.
	.vcRetStatOK 1	The context menu will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcBoxRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcBoxClickingEventArgs) Handles VcGantt1.VcBoxRightClicking
    PopupMenu.Show(VcGantt1, New Point(e.X, e.Y))
End Sub
```

Example Code C#

```
private void vcGantt1_VcBoxRightClicking(object sender,
NETRONIC.XGantt.VcBoxClickingEventArgs e)
{
    PopupMenu.Show(vcGantt1, new Point (e.X, e.Y));
}
```

VcCalendarGridRightClicking**Event of VcGantt**

This event occurs when the user clicks the right mouse button on a calendar grid. The calendar grid object and the position of the mouse (x,y-coordinates) will be returned. By setting the return status you can inhibit the integrated context menu to pop up and replace it by a context menu of your own at the position delivered.

This event will be triggered only if the calendar grid could be identified, i.e. if the calendar grid property **Identifiable** had been set to **True**.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcCalendarGridClickingEventArgs	Object specific to the event that is being handled

Properties of the VcCalendarGridClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ calendarGrid	VcCalendarGrid	Calendar grid hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status

Example Code VB.NET

```
Private Sub VcGantt1_VcCalendarGridRightClicking(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcCalendarGridClickingEventArgs) Handles
VcGantt1.VcCalendarGridRightClicking
    MsgBox(e.CalendarGrid.Name)
End Sub
```

Example Code C#

```
private void vcGantt1_VcCalendarGridRightClicking(object sender,
NETRONIC.XGantt.VcCalendarGridClickingEventArgs e)
{
    MessageBox.Show(e.CalendarGrid.Name);
}
```

VcComponentScrolled

Event of VcGantt

For each interactive scrolling action this event lets you identify the below listed values:

1. the scrolled component (only `vcDiagramComponent`, `vcHistogramComponent`, `vcListComponent` and `vcRightListComponent` are considered as "Master scrollers" because the other components depend on these and are scrolled together with them)
2. the scrolling direction (horizontal or vertical)
3. the type of user action.

Note: The actual scroll action results from the combination of the parameters **orientation** and **scrollAction**, because in Windows programs the up/left- and down/right actions have got the same numbers, e. g.:

`vcScrollActionSBPageLeft = vcScrollActionSBPageUp = 2`

`vcScrollActionThumbTrackLeft = vcScrollActionThumbTrackUp = 107`

The following example shows the distinction by the usage of the parameter **orientation** for **VcScrollActionSBPageLeft** and **vcScrollActionSBPageUp** which have both the value 2.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcComponentScrolledEventArgs	Object specific to the event that is being handled

Properties of the VcComponentScrolledEventArgs object

	Data Type	Explanation
Properties: ⇒ component	VcComponentType Possible Values: .vcAdditionalListComponent 1 .vcBottomListTitleComponent 14 .vcBottomRightListTitleComponent 17 .vcBottomTimeScaleComponent 15 .vcDiagramComponent 4 .vcHistogramComponent 8 .vcHistogramVerScaleComponent 7 .vcLegendComponent 10 .vcListComponent 0 .vcListTitleComponent 2 .vcRightListComponent 5 .vcRightListTitleComponent 16 .vcTimeScaleComponent 3 .vcTopTitleComponent 11	Component type additional table bottom title bar bottom right table bottom time scale diagram histogram numeric scale (vertical histogram scale) legend (currently functionless; return values 00) table table title right table table title of right table upper time scale upper title bar
⇒ orientation	VcScrollOrientation Possible Values: .vcHorizontal 1 .vcVertical 2	Scrolling direction horizontal scrolling vertical scrolling
⇒ scrollAction	VcScrollAction Possible Values: .vcScrollActionAutoscrollDown 102 .vcScrollActionAutoscrollLeft 101 .vcScrollActionAutoscrollRight 102 .vcScrollActionAutoscrollUp 101 .vcScrollActionMouseWheelDown 106 .vcScrollActionMouseWheelLeft 105 .vcScrollActionMouseWheelRight 106 .vcScrollActionMouseWheelUp 105 .vcScrollActionSBLineDown 1 .vcScrollActionSBLineLeft 0 .vcScrollActionSBLineRight 1 .vcScrollActionSBLineUp 0 .vcScrollActionSBNothing -1 .vcScrollActionSBPageDown 3 .vcScrollActionSBPageLeft 2	Type of scrolling The view was automatically scrolled downward. The view was automatically scrolled towards the right. The view was automatically scrolled towards the left. The view was automatically scrolled upward. While the mouse wheel was pressed, the mouse was moved downward. While the mouse wheel was pressed, the mouse was moved towards the left. While the mouse wheel was pressed, the mouse was moved towards the right. While the mouse wheel was pressed, the mouse was moved upward. The view was automatically scrolled to its bottom limit The view was automatically scrolled to its left limit The view was automatically scrolled to its right limit The view was automatically scrolled to its top limit The view was not scrolled The view was scrolled downward by a page The view was scrolled towards the left by a page

.vcScrollActionSBPageRight 3	The view was scrolled towards the right by a page
.vcScrollActionSBPageUp 2	The view was scrolled upward by a page
.vcScrollActionSBThumbPosition 4	The scrolling by a step has been finished.
.vcScrollActionSBThumbTrack 5	The view was scrolled by a step
.vcScrollActionScrollEnd 104	Scrolling via the End button or the context menu to the diagram end (right down)
.vcScrollActionScrollHome 103	Scrolling via the Pos 1 button or the context menu to the upper left corner of the diagram
.vcScrollActionThumbTrackDown 108	Thumb (bar of the scrollbar) moved down
.vcScrollActionThumbTrackLeft 107	Thumb (bar of the scrollbar) moved toward the left
.vcScrollActionThumbTrackRight 108	Thumb (bar of the scrollbar) moved toward the right
.vcScrollActionThumbTrackUp 107	Thumb (bar of the scrollbar) moved up

Example Code VB.NET

```
Private Sub VcGantt1_VcComponentScrolled(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcComponentScrolledEventArgs) Handles
VcGantt1.VcComponentScrolled
    If e.ScrollOrientation = VcScrollOrientation.vcHorizontal And e.ScrollAction
= VcScrollAction.vcScrollActionSBPageLeft Then
        MsgBox("Scrolled left")
    ElseIf e.ScrollOrientation = VcScrollOrientation.vcHorizontal And
e.ScrollAction = VcScrollAction.vcScrollActionSBPageRight Then
        MsgBox("Scrolled right")
    ElseIf e.ScrollOrientation = VcScrollOrientation.vcVertical And
e.ScrollAction = VcScrollAction.vcScrollActionSBPageUp Then
        MsgBox("Scrolled up")
    ElseIf e.ScrollOrientation = VcScrollOrientation.vcHorizontal And
e.ScrollAction = VcScrollAction.vcScrollActionSBPageDown Then
        MsgBox("Scrolled down")
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcComponentScrolled(object sender,
NETRONIC.XGantt.VcComponentScrolledEventArgs e)
{
    if (e.ScrollOrientation == VcScrollOrientation.vcHorizontal && e.ScrollAction
== VcScrollAction.vcScrollActionSBPageLeft)
        MessageBox.Show("Scrolled left");
    else if (e.ScrollOrientation == VcScrollOrientation.vcHorizontal &&
e.ScrollAction == VcScrollAction.vcScrollActionSBPageRight)
        MessageBox.Show("Scrolled right");
    else if (e.ScrollOrientation == VcScrollOrientation.vcVertical &&
e.ScrollAction == VcScrollAction.vcScrollActionSBPageUp)
        MessageBox.Show("Scrolled up");
    else if (e.ScrollOrientation == VcScrollOrientation.vcVertical &&
e.ScrollAction == VcScrollAction.vcScrollActionSBPageDown)
        MessageBox.Show("Scrolled down");
}
```

VcComponentScrolling

Event of VcGantt

This event occurs when you have ordered a scroll action, but before the integrated scrolling process is performed. This event lets you acquire for each interactive scroll action:

- 1. the scrolled component (only vcDiagramComponent, vcHistogramComponent, vcListComponent and vcRightListComponent are considered as "Master scrollers" because the other components depend on these and are scrolled together with them)
- 2. the scrolling direction (horizontal or vertical)
- 3. the type of user action.

If you set the returnStatus to **vcRetStatFalse**, the integrated scrolling process will be suppressed, and in your application, you can react to the event with your own solution.

Note: The actual scroll action results from the combination of the parameters **orientation** and **scrollAction**, because in Windows programs the up/left- and down/right actions have got the same numbers, e. g.:

vcScrollActionSBPageLeft = vcScrollActionSBPageUp = 2

vcScrollActionThumbTrackLeft = vcScrollActionThumbTrackUp = 107

The following example shows the distinction by the usage of the parameter **orientation** for **VcScrollActionSBPageLeft** and **vcScrollActionSBPageUp** which have both the value 2.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcComponentScrollingEventArgs	Object specific to the event that is being handled

Properties of the VcComponentScrollingEventArgs object

	Data Type	Explanation
Properties:		
⇒ component	VcComponentType Possible Values: .vcAdditionalListComponent 1 .vcBottomListComponent 14 .vcBottomRightListTitleComponent 17 .vcBottomTimeScaleComponent 15 .vcDiagramComponent 4 .vcHistogramComponent 8 .vcHistogramVerScaleComponent 7 .vcLegendComponent 10 .vcListComponent 0 .vcListTitleComponent 2 .vcRightListComponent 5 .vcRightListTitleComponent 16 .vcTimeScaleComponent 3 .vcTopTitleComponent 11	Component type additional table bottom title bar bottom right table bottom time scale diagram histogram numeric scale (vertical histogram scale) legend (currently functionless; return values 00) table table title right table table title of right table upper time scale upper title bar
⇒ histogramsHeightRatio		ratio of the histogram height to the complete diagram
⇒ orientation	VcScrollOrientation Possible Values: .vcHorizontal 1 .vcVertical 2	Scrolling direction horizontal scrolling vertical scrolling
⇒ scrollAction	VcScrollAction Possible Values: .vcScrollActionAutoscrollDown 102 .vcScrollActionAutoscrollLeft 101 .vcScrollActionAutoscrollRight 102 .vcScrollActionAutoscrollUp 101 .vcScrollActionMouseWheelDown 106 .vcScrollActionMouseWheelLeft 105 .vcScrollActionMouseWheelRight 106 .vcScrollActionMouseWheelUp 105 .vcScrollActionSBLineDown 1 .vcScrollActionSBLineLeft 0 .vcScrollActionSBLineRight 1 .vcScrollActionSBLineUp 0 .vcScrollActionSBNothing -1 .vcScrollActionSBPageDown 3 .vcScrollActionSBPageLeft 2	Type of scrolling The view was automatically scrolled downward. The view was automatically scrolled towards the right. The view was automatically scrolled towards the left. The view was automatically scrolled upward. While the mouse wheel was pressed, the mouse was moved downward. While the mouse wheel was pressed, the mouse was moved towards the left. While the mouse wheel was pressed, the mouse was moved towards the right. While the mouse wheel was pressed, the mouse was moved upward. The view was automatically scrolled to its bottom limit The view was automatically scrolled to its left limit The view was automatically scrolled to its right limit The view was automatically scrolled to its top limit The view was not scrolled The view was scrolled downward by a page The view was scrolled towards the left by a page

	<code>.vcScrollActionSBPageRight</code> 3	The view was scrolled towards the right by a page
	<code>.vcScrollActionSBPageUp</code> 2	The view was scrolled upward by a page
	<code>.vcScrollActionSBThumbPosition</code> 4	The scrolling by a step has been finished.
	<code>.vcScrollActionSBThumbTrack</code> 5	The view was scrolled by a step
	<code>.vcScrollActionScrollEnd</code> 104	Scrolling via the End button or the context menu to the diagram end (right down)
	<code>.vcScrollActionScrollHome</code> 103	Scrolling via the Pos 1 button or the context menu to the upper left corner of the diagram
	<code>.vcScrollActionThumbTrackDown</code> 108	Thumb (bar of the scrollbar) moved down
	<code>.vcScrollActionThumbTrackLeft</code> 107	Thumb (bar of the scrollbar) moved toward the left
	<code>.vcScrollActionThumbTrackRight</code> 108	Thumb (bar of the scrollbar) moved toward the right
	<code>.vcScrollActionThumbTrackUp</code> 107	Thumb (bar of the scrollbar) moved up
⇒ delta	<code>System.Int32</code>	Scrolling length (in pixels)
⇔ returnStatus	<code>VcReturnStatus</code>	Return status
	Possible Values:	
	<code>.vcRetStatDefault</code> 2	The default behavior remains unchanged.
	<code>.vcRetStatFalse</code> 0	The default behavior will not be performed.
	<code>.vcRetStatNoPopup</code> 4	The popup of the context menu is inhibited.
	<code>.vcRetStatOK</code> 1	The default behavior will be performed.

Example Code VB.NET

```

Private Sub VcGantt1_VcComponentScrolling(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcComponentScrollingEventArgs) Handles
VcGantt1.VcComponentScrolling
    If e.ScrollOrientation = VcScrollOrientation.vcHorizontal And e.ScrollAction
= VcScrollAction.vcScrollActionSBPageLeft Then
        MsgBox("Scrolled left")
    ElseIf e.ScrollOrientation = VcScrollOrientation.vcHorizontal And
e.ScrollAction = VcScrollAction.vcScrollActionSBPageRight Then
        MsgBox("Scrolled right")
    ElseIf e.ScrollOrientation = VcScrollOrientation.vcVertical And
e.ScrollAction = VcScrollAction.vcScrollActionSBPageUp Then
        MsgBox("Scrolled up")
    ElseIf e.ScrollOrientation = VcScrollOrientation.vcHorizontal And
e.ScrollAction = VcScrollAction.vcScrollActionSBPageDown Then
        MsgBox("Scrolled down")
    End If
End Sub

```

Example Code C#

```
private void vcGantt1_VcComponentScrolling(object sender,
NETRONIC.XGantt.VcComponentScrollingEventArgs e)
{
    if (e.ScrollOrientation == VcScrollOrientation.vcHorizontal && e.ScrollAction
== VcScrollAction.vcScrollActionSBPageLeft)
        MessageBox.Show("Scrolled left");
    else if (e.ScrollOrientation == VcScrollOrientation.vcHorizontal &&
e.ScrollAction == VcScrollAction.vcScrollActionSBPageRight)
        MessageBox.Show("Scrolled right");
    else if (e.ScrollOrientation == VcScrollOrientation.vcVertical &&
e.ScrollAction == VcScrollAction.vcScrollActionSBPageUp)
        MessageBox.Show("Scrolled up");
    else if (e.ScrollOrientation == VcScrollOrientation.vcVertical &&
e.ScrollAction == VcScrollAction.vcScrollActionSBPageDown)
        MessageBox.Show("Scrolled down");
}
```

VcCurveLeftClicking**Event of VcGantt**

This event occurs when the user clicks the left mouse button on a histogram curve, and before a curve is marked. By setting the **VcReturnStatus** to **vcRetStatFalse** marking of the curve can be prohibited. In spite of this, the curve values can be modified. At the moment, there is no option to suppress this. The curve object hit and the position of the mouse (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcCurveClickingEventArgs	Object specific to the event that is being handled

Properties of the VcCurveClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ curve	VcCurve	Curve hit in histogram
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The curve will not be marked. The curve will be marked.

Example Code VB.NET

```
Private Sub VcGantt1_VcCurveLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcCurveClickingEventArgs) Handles VcGantt1.VcCurveLeftClicking
    e.Curve.LineColor = Color.Blue
End Sub
```

Example Code C#

```
private void vcGantt1_VcCurveLeftClicking(object sender,
NETRONIC.XGantt.VcCurveClickingEventArgs e)
{
    e.Curve.LineColor = Color.LightSteelBlue;
}
```

VcCurveLeftDoubleClicking

Event of VcGantt

This event occurs when the user double-clicks the left mouse button on a histogram curve. The VcCurve object hit and the mouse position (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcCurveClickingEventArgs	Object specific to the event that is being handled

Properties of the VcCurveClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ curve	VcCurve	Curve hit in histogram
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcCurveLeftDoubleClicking(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcCurveClickingEventArgs) Handles
VcGantt1.VcCurveLeftDoubleClicking
    Call MsgBox("x: " + e.X.ToString() + "y: " + e.Y.ToString())
End Sub
```

Example Code C#

```
private void vcGantt1_VcCurveLeftDoubleClicking(object sender,
NETRONIC.XGantt.VcCurveClickingEventArgs e)
{
    MessageBox.Show("x: " + e.X.ToString() + " y: " + e.Y.ToString());
}
```

VcCurveModified

Event of VcGantt

This event occurs when the modification of the curve specified is finished.

The curve object is returned, so that a validation can be made.

	Data Type	Explanation
Properties:		
⇒ node	VcCurve	Curve modified

VcCurveModifying

Event of VcGantt

This event occurs when the user has modified a histogram curve interactively. This is valid for histogram curves generated by API and for histogram curves generated by layers. The modified curve object, the beginning and the end of the section changed, as well as the value that the curve was changed by in y direction are returned. The curve type can be retrieved by the VcCurve property **CurveSource**.

Please note: For each modified layer that contributes to the modification of an histogram curve the event **VcCurveModifying** occurs twice (once for the start date and once for the end date of the modified curve section).

The data passed by this event can be read, but must not be modified. For modifying them please use the event **VcCurveModified**.

By setting the return status the modification can be inhibited.

Please note: By using the event **VcCurvePointModifyingEx** the y value may be a floating point value.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcCurveModifyingEventArgs	Object specific to the event that is being handled

Properties of the VcCurveModifyingEventArgs object

	Data Type	Explanation
Properties:		
⇒ curve	VcCurve	Curve modified
⇒ date1	System.DateTime	beginning of the curve section changed
⇒ date2	System.DateTime	End of the curve section changed
⇒ increment	System.Int32	Value that the curve was changed by in y direction
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The modification will be revoked. The modification will be accepted.

Example Code VB.NET

```
Private Sub VcGantt1_VcCurveModifying(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcCurveModifyingEventArgs) Handles VcGantt1.VcCurveModifying
    Select Case e.Curve.CurveSource
        Case VcCurveSource.vcCalculateFromLayer
            MsgBox("The curve is calculated from layers. Increment:" +
e.Increment.ToString() + " Changed start date: " + e.LeftDate + "Changed end
date: " + e.RightDate)
        Case VcCurveSource.vcSetCurve
            MsgBox("Curve set via API. Increment:" + e.Increment.ToString() + "
Changed start date: " + e.LeftDate + "Changed end date: " + e.RightDate)
    End Select
End Sub
```

Example Code C#

```
private void vcGantt1_VcCurveModifying(object sender,
NETRONIC.XGantt.VcCurveModifyingEventArgs e)
{
    switch (e.Curve.CurveSource)
    {
        case VcCurveSource.vcCalculateFromLayer:
            MessageBox.Show("The curve is calculated from layers. Increment: " +
e.Increment.ToString() + " Changed start date: " + e.LeftDate + " Changed end
date: " + e.RightDate);
            break;
        case VcCurveSource.vcSetCurve:
            MessageBox.Show("Curve set via API. Increment: " +
e.Increment.ToString() + " Changed start date: " + e.LeftDate + " Changed end
date: " + e.RightDate);
            break;
    }
}
```

VcCurveModifyingEx

Event of VcGantt

This event occurs when the user has modified a histogram curve interactively. This is valid for histogram curves generated by the API and for histogram curves generated by layers. The modified curve object, the beginning and the end of the section changed, as well as the value that the curve was changed by in y direction are returned. The curve type can be retrieved by the VcCurve property **CurveSource**.

Note: For each modified layer that contributes to the modification of an histogram curve the event **VcCurveModifying** occurs twice (once for the start date and once for the end date of the modified curve section).

The data passed by this event can be read, but must not be modified. For modifying them please use the event **VcCurveModified**.

By setting the return status the modification can be inhibited.

Please note: Compared to the event **OnCurveModifyEx**, this event allows the parameter **increment** to be a floating point number.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcCurveModifyingEventArgs	Object specific to the event that is being handled

Properties of the VcCurveModifyingEventArgs object

	Data Type	Explanation
Properties:		
⇒ curve	VcCurve	Curve modified
⇒ leftDate	System.DateTime	Beginning of the curve section changed
⇒ rightDate	System.DateTime	End of the curve section changed
⇒ increment	System.Int32	Value that the curve was changed by in y direction
↔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The modification will be revoked. The modification will be accepted.

Example Code VB.NET

```
Private Sub VcGantt1_VcCurveModifying(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcCurveModifyingEventArgs) Handles VcGantt1.VcCurveModifying
    Select Case e.Curve.CurveSource
        Case VcCurveSource.vcCalculateFromLayer
            MsgBox("The curve is calculated from layers. Increment:" +
e.Increment.ToString() + " Changed start date: " + e.LeftDate + "Changed end
date: " + e.RightDate)
        Case VcCurveSource.vcSetCurve
            MsgBox("Curve set via API. Increment:" + e.Increment.ToString() + "
Changed start date: " + e.LeftDate + "Changed end date: " + e.RightDate)
    End Select
End Sub
```

Example Code C#

```
private void vcGantt1_VcCurveModifying(object sender,
NETRONIC.XGantt.VcCurveModifyingEventArgs e)
{
    switch (e.Curve.CurveSource)
    {
        case VcCurveSource.vcCalculateFromLayer:
            MessageBox.Show("The curve is calculated from layers. Increment: " +
e.Increment.ToString() + " Changed start date: " + e.LeftDate + " Changed end
date: " + e.RightDate);
            break;
        case VcCurveSource.vcSetCurve:
            MessageBox.Show("Curve set via API. Increment: " +
e.Increment.ToString() + " Changed start date: " + e.LeftDate + " Changed end
date: " + e.RightDate);
            break;
    }
}
```

VcCurvePointDeleting

Event of VcGantt

This event occurs when the user deletes a curve point of an histogram curve set by the API. It returns the histogram curve, the date and the y value of the deleted curve point. By setting the return status the deleting operation can be inhibited.

Please note: By using the event **VcCurvePointDeletingEx** you may pass the y value as a floating point number.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcCurvePointDeletingEventArgs	Object specific to the event that is being handled

Properties of the VcCurvePointDeletingEventArgs object

	Data Type	Explanation
Properties:		
⇒ curve	VcCurve	Hit histogram curve
⇒ pointDate	System.DateTime	Date of the deleted curve point
⇒ value	System.Int32	Y value of the deleted curve point
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The curve point will not be deleted. The curve point will be deleted.

Example Code VB.NET

```
Private Sub VcGantt1_VcCurvePointDeleting(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcCurvePointDeletingEventArgs) Handles
VcGantt1.VcCurvePointDeleting
    If MsgBox("Do you want to delete this curve point (date:" + e.PointDate + ",
y value:" + e.Value + ")?", MsgBoxStyle.YesNo, "Deleting curve point") =
MsgBoxResult.No Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcCurvePointDeleting(object sender,
NETRONIC.XGantt.VcCurvePointDeletingEventArgs e)
{
    DialogResult retVal = MessageBox.Show("Do you want to delete this curve
point (date:" + e.PointDate + ", y value:" + e.Value + ")?", "Deleting curve
point", MessageBoxButtons.YesNo);
    if (retVal == DialogResult.No)
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcCurvePointDeletingEx

Event of VcGantt

This event occurs when the user deletes a curve point of an histogram curve set by the API. It returns the histogram curve, the date and the y value of the deleted curve point. By setting the return status the deleting operation can be inhibited.

Please note: Compared to the event **VcCurvePointDeleting**, this event allows the parameter **value** to be a floating point number.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event

⇒ e	VcCurvePointDeletingEventArgs	Object specific to the event that is being handled
-----	-------------------------------	--

Properties of the VcCurvePointDeletingEventArgs object

	Data Type	Explanation
Properties:		
⇒ curve	VcCurve	Hit histogram curve
⇒ pointDate	System.DateTime	Date of the deleted curve point
⇒ value	System.Int32	Y value of the deleted curve point
⇌ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The curve point will not be deleted. The curve point will be deleted.

Example Code VB.NET

```
Private Sub VcGantt1_VcCurvePointDeleting(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcCurvePointDeletingEventArgs) Handles
VcGantt1.VcCurvePointDeleting
    If MsgBox("Do you want to delete this curve point (date:" + e.PointDate + ",
y value:" + e.Value + ")?", MsgBoxStyle.YesNo, "Deleting curve point") =
MsgBoxResult.No Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcCurvePointDeleting(object sender,
NETRONIC.XGantt.VcCurvePointDeletingEventArgs e)
{
    DialogResult retVal = MessageBox.Show("Do you want to delete this curve
point (date:" + e.PointDate + ", y value:" + e.Value + ")?", "Deleting curve
point", MessageBoxButtons.YesNo);
    if (retVal == DialogResult.No)
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcCurvePointInserting

Event of VcGantt

This event occurs when the user has selected the histogram context menu item **Mode: Insert curve point** and then inserts a curve point to an histogram curve set by the API. It returns the histogram curve, the date and the y value of the inserted curve point. If you set the returnStatus to **vcRetStatFalse**, the inserting operation will be revoked.

Please note: By using the event **VcCurvePointInsertingEx** the y value may be a floating point value.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcCurvePointInsertingEventArgs	Object specific to the event that is being handled

Properties of the VcCurvePointInsertingEventArgs object

	Data Type	Explanation
Properties:		
⇒ curve	VcCurve	Hit histogram curve
⇒ pointDate	System.DateTime	Date of the inserted curve point
⇒ value	System.Int32	Y value of the inserted curve point
↔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcCurvePointInserting(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcCurvePointInsertingEventArgs) Handles
VcGantt1.VcCurvePointInserting
    If MsgBox("Do you want to insert this curve point (date: " + e.PointDate + "
y value: " + e.Value + ")?", MsgBoxStyle.YesNo, "Deleting curve point") =
MsgBoxResult.No Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcCurvePointInserting(object sender,
NETRONIC.XGantt.VcCurvePointInsertingEventArgs e)
{
    DialogResult retVal = MessageBox.Show("Do you want to insert this curve point
(date:" + e.PointDate + ", y value:" + e.Value + ")?", "Inserting curve point",
MessageBoxButtons.YesNo);
    if (retVal == DialogResult.No)
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```


VcCurvePointInsertingEx

Event of VcGantt

This event occurs when the user has selected the histogram context menu item **Mode: Insert curve point** and then inserts a curve point to an histogram curve set by the API. It returns the histogram curve, the date and the y value of the inserted curve point. If you set the returnStatus to **vcRetStatFalse**, the inserting operation will be revoked.

Please note: Compared to the event **VcCurvePointInserting** the y value of this event may be a floating point value.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcCurvePointInsertingEventArgs	Object specific to the event that is being handled

Properties of the VcCurvePointInsertingEventArgs object

	Data Type	Explanation
Properties:		
⇒ curve	VcCurve	Hit histogram curve
⇒ pointDate	System.DateTime	Date of the inserted curve point
⇒ value	System.Int32	Y value of the inserted curve point
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

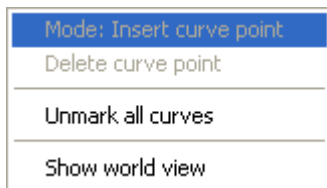
```
Private Sub VcGantt1_VcCurvePointInserting(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcCurvePointInsertingEventArgs) Handles
VcGantt1.VcCurvePointInserting
    If MsgBox("Do you want to insert this curve point (date: " + e.PointDate + "
y value: " + e.Value + ")?", MsgBoxStyle.YesNo, "Deleting curve point") =
MsgBoxResult.No Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcCurvePointInserting(object sender,
NETRONIC.XGantt.VcCurvePointInsertingEventArgs e)
{
    DialogResult retVal = MessageBox.Show("Do you want to insert this curve point
(date:" + e.PointDate + ", y value:" + e.Value + ")?", "Inserting curve point",
MessageBoxButtons.YesNo);
    if (retVal == DialogResult.No)
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcCurveRightClicking**Event of VcGantt**

This event occurs when the user clicks the right mouse button on the curve. The curve object and the position of the mouse (x,y-coordinates) are returned. By setting the return status you can inhibit the integrated context menu to pop up and replace it by a context menu of your own at the location delivered.



Above: integrated context menu

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcCurveClickingEventArgs	Object specific to the event that is being handled

Properties of the VcCurveClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ curve	VcCurve	Curve hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatNoPopUp 4	The context menu will be inhibited.

.vcRetStatOK 1

The context menu will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcCurveRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcCurveClickingEventArgs) Handles VcGantt1.VcCurveRightClicking
    PopupMenu.Show(VcGantt1, New Point(e.X, e.Y))
    e.ReturnStatus = NETRONIC.XGantt.VcReturnStatus.vcRetStatNoPopup
End Sub
```

Example Code C#

```
private void vcGantt1_VcCurveRightClicking(object sender,
NETRONIC.XGantt.VcCurveClickingEventArgs e)
{
    PopupMenu.Show(vcGantt1, new Point (e.X, e.Y));
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

VcDataModified**Event of VcGantt**

This event occurs after data were interactively modified in the chart, i.e. after the below events:

- VcBoxModified
- VcCurveModifying
- VcCurvePointDeleting
- VcGroupModified
- VcLinkCreated
- VcLinkDeleted
- VcNodeCreated
- VcNodeDeleting
- VcNodeModified

This event allows you to set a marker to the application that reminds the user or the program to save the data before closing.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDataModifiedEventArgs	Object specific to the event that is being handled

Properties of the VcDataModifiedEventArgs object

	Data Type	Explanation
Properties:		
⇐ (no parameter)		No parameter

VcDataRecordCreated

Event of VcGantt

This event occurs when the interactive creation of a data record is completed. The data record object, the creation type (**VcDataRecordCreated** and **VcDataRecordCreatedByResourceScheduling** only) and the information whether the data record created is the only one or the last one of a data record collection (momentarily always **True**) are returned, so that depending data can be validated.

If a link or a node was created, you can in addition react to the analogous link or node event and verify additional graphical data (s. **VcNodeCreated** and **VcLinkCreated**).

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDataRecordCreatedEventArgs	Object specific to the event that is being handled

Properties of the VcDataRecordCreatedEventArgs object

	Data Type	Explanation
Properties:		
⇒ dataRecord	VcDataRecord	DataRecord object created
⇒ creationType	VcCreationType	Creation type of data records

	Possible Values: .vcDataRecordCreated 6 .vcDataRecordCreatedByResourceScheduling 5 .vcLinkCreated 2 .vcNodeCreated 1	Data record created by interaction Data record automatically created by resource scheduling Link created by interaction Node created via mouse-click
⇒ isLast	System.Boolean	True: The data record created is the only one or the last one of a data record collection. False: The data record created is not the only one or the last one of a data record collection.

Example Code VB.NET

```
Private Sub VcGantt1_VcDataRecordCreated(ByVal sender As Object, ByVal e As NETRONIC.XGantt.VcDataRecordCreatedEventArgs) Handles VcGantt1.VcDataRecordCreated
    MsgBox(e.DataRecord.AllData)
End Sub
```

Example Code C#

```
private void vcGantt1_VcDataRecordCreated(object sender, NETRONIC.XGantt.VcDataRecordCreatedEventArgs e)
{
    MessageBox.Show(e.DataRecord.AllData.ToString());
}
```

VcDataRecordCreating

Event of VcGantt

This event occurs when the user creates a an object that generates a data record. The generated data record object is returned, so that the data can be validated and, if necessary, a data base entry can be made.

The data passed by this event can be read, but must not be modified. For modifying them please use the event **VcDataRecordCreated**.

By setting the return status the create operation can be inhibited.

If a link or a node was created, you can in addition react to the analogous link or node event and verify additional graphical data (s. **VcNodeCreating** and **VcLinkCreating**).

	Data Type	Explanation
Parameter: ⇒ sender	VcGantt	Reference to the object that triggered the event

⇒ e	VcDataRecordCreatingEventArgs	Object specific to the event that is being handled
-----	-------------------------------	--

Properties of the VcDataRecordCreatingEventArgs object

	Data Type	Explanation
Properties:		
⇒ dataRecord	VcDataRecord	DataRecord object created
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The data record will not be created. The data record will be created.

Example Code VB.NET

```
Private Sub VcVcGantt1_VcDataRecordCreated(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcDataRecordCreatedEventArgs) Handles
VcVcGantt1.VcDataRecordCreated
    MsgBox(e.DataRecord.AllData)
End Sub
```

Example Code C#

```
private void vcGantt1_VcDataRecordCreated(object sender,
NETRONIC.XGantt.VcDataRecordCreatedEventArgs e)
{
    MessageBox.Show(e.DataRecord.AllData.ToString());
}
```

VcDataRecordDeleted

Event of VcGantt

This event occurs when the deletion of an object based on a data record is completed. The data record and the information whether the deleted data record is the only one or the last one of a data record collection are returned, so that depending data can be validated.

If a link or a node was deleted, you can in addition react to the analogous link or node event and verify additional graphical data (s. **VcNodeDeleted** and **VcLinkDeleted**).

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDataRecordDeletedEventArgs	Object specific to the event that is being handled

Properties of the VcDataRecordDeletedEventArgs object

	Data Type	Explanation
Properties:		
⇒ dataRecord	VcDataRecord	Data record deleted
⇒ isLast	System.Boolean	True: The data record deleted is the only one or the last one of a data record collection. False: The data record deleted is not the only one or the last one of a data record collection.

VcDataRecordDeleting

Event of VcGantt

This event occurs when a user deletes an object by the context menu if the object was based on a data record. The data record object to be deleted is returned, so that you can still verify its data and prohibit the deletion on a negative result by setting the return status.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDataRecordDeletingEventArgs	Object specific to the event that is being handled

Properties of the VcDataRecordDeletingEventArgs object

	Data Type	Explanation
Properties:		
⇒ dataRecord	VcDataRecord	Data record object deleted
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The data record will not be deleted. The data record will be deleted.

Example Code VB.NET

```
Private Sub VcVcGantt1_VcDataRecordDeleting(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcDataRecordDeletingEventArgs) Handles
VcVcGantt1.VcDataRecordDeleting
    'deny deletion of data record with a certain value
    If e.DataRecord.DataField(0) = "1" Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcDataRecordDeleting(object sender,
NETRONIC.XGantt.VcDataRecordDeletingEventArgs e)
{
    // deny deletion of data record with a certain value
    if (e.DataRecord.get_DataField(0).Equals("1"))
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcDataRecordModified

Event of VcGantt

This event occurs when the modification of the box is finished.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDataRecordModifiedEventArgs	Object specific to the event that is being handled

Properties of the VcDataRecordModifiedEventArgs object

	Data Type	Explanation
Properties:		
⇒ dataRecord	VcDataRecord	Data record modified

Example Code VB.NET

```
Private Sub VcGantt1_VcDataRecordModified(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcDataRecordModifiedEventArgs) Handles
VcGantt1.VcDataRecordModified
    MsgBox("The data record has been modified")
End Sub
```

Example Code C#

```
private void vcGantt1_VcDataRecordModified(object sender,
NETRONIC.XGantt.VcDataRecordModifiedEventArgs e)
{
    MessageBox.Show("The data record has been modified");
}
```


VcDataRecordModifying

Event of VcGantt

This event occurs when the user has modified a box interactively. The modified VcBox object and the modification type are returned.

The data passed by this event can be read, but must not be modified. For modifying them please use the event **VcDataRecordModified**.

By setting the return status the modification can be inhibited.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDataRecordModifyingEventArgs	Object specific to the event that is being handled

Properties of the VcDataRecordModifyingEventArgs object

	Data Type	Explanation
Properties:		
⇒ dataRecord	VcDataRecord	Data record modified
⇒ modificationType	VcModificationTypes	Modification type
	Possible Values: .vcAnything 1 .vcChangedGroup 16 .vcEndModified 4 .vcHierarchyModified 64 .vcModifiedByResourceScheduling 128 .vcModifiedBySchedule 32 .vcMoved 8 .vcNothing 0 .vcStartModified 2	Modification type cannot be identified. Group of the node was changed (occurs with nodes only). The end date of the node was modified (occurs with nodes only). Hierarchy of the nodes has been changed Modification by resource scheduling (occurs with data records only) Modification by new date calculation Object was moved. No modification The start date of the node was modified (occurs with nodes only).
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The modification will be revoked. The modification will be accepted.

VcDataRecordNotFound

Event of VcGantt

This event occurs if a depending data record was not found. The index of the field of the current data record, which holds the key to the depending data record, is returned and thus offers some information on the data record not found.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	
⇒ e	VcDataRecordNotFoundEventArgs	

	Data Type	Explanation
Properties:		
↔ index	System.Int32	Index of the field that contains the key of the depending data record

VcDateLineModifying

Event of VcGantt

This event occurs when the user has moved a date line. The modified date line object is captured and returned so that you receive the new values.

By setting the return status the modification can be inhibited.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDateLineModifyingEventArgs	Object specific to the event that is being handled

Properties of the VcDateLineModifyingEventArgs object

	Data Type	Explanation
Properties:		
⇒ dateLine	VcDateLine	Date line
⇒ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The modification will be revoked. The modification will be accepted.

Example Code VB.NET

```
Private Sub VcGantt1_VcDateLineModifying(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcDateLineModifyingEventArgs) Handles
VcGantt1.VcDateLineModifying
    MsgBox(e.DateLine.Date)
End Sub
```

Example Code C#

```
private void vcGantt1_VcDateLineModifying(object sender,
NETRONIC.XGantt.VcDateLineModifyingEventArgs e)
{
    MessageBox.Show(e.DateLine.Date.ToString());
}
```

VcDateLineRightClicking**Event of VcGantt**

This event occurs when the user clicks the right mouse button on the date line. The date line object and the position of the mouse (x,y-coordinates) are returned. By setting the return status you can inhibit the integrated context menu to pop up and replace it by a context menu of your own at the location delivered.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDateLineClickingEventArgs	Object specific to the event that is being handled

Properties of the VcDateLineClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ dateLine	VcDateLine	Date line hit
⇒ x	System.Int32	X coordinate of the mouse cursor

⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values:	
	.vcRetStatNoPopup 4	The context menu will be inhibited.
	.vcRetStatOK 1	The context menu will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcDateLineRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcDateLineClickingEventArgs) Handles
VcGantt1.VcDateLineRightClicking
    MsgBox(e.DateLine.Name)
End Sub
```

Example Code C#

```
private void vcGantt1_VcDateLineRightClicking(object sender,
NETRONIC.XGantt.VcDateLineClickingEventArgs e)
{
    MessageBox.Show(e.DateLine.Name);
}
```

VcDateShowing

Event of VcGantt

This event occurs when the user moves the mouse inside the diagram or the time scale area. The date of the mouse position is returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDateShowingEventArgs	Object specific to the event that is being handled

Properties of the VcDateShowingEventArgs object

	Data Type	Explanation
Properties:		
⇒ dateVal	System.DateTime	Date

Example Code VB.NET

```
Private Sub VcGantt1_VcDateShowing(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcDateShowingEventArgs) Handles VcGantt1.VcDateShowing
    TextBox1.Text = e.CurDate
End Sub
```

Example Code C#

```
private void vcGantt1_VcDateShowing(object sender,
NETRONIC.XGantt.VcDateShowingEventArgs e)
{
    textBox1.Text = e.CurDate.ToString();
}
```

VcDiagramHorizontalScrolled**Event of VcGantt**

This event occurs after a scroll action was performed. The new start and end date of the visible diagram area are captured and passed. The **scrollAction** parameter provides information about the type of the performed scrolling process.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDiagramHorizontalScrolledEventArgs	Object specific to the event that is being handled

Properties of the VcDiagramHorizontalScrolledEventArgs object

	Data Type	Explanation
Properties:		
⇒ newStartDate	System.DateTime	New start date of the visible part of the diagram
⇒ newEndDate	System.DateTime	New final date of the visible part of the diagram
⇒ scrollAction	VcScrollAction	Scrolling type
	Possible Values:	
	.vcScrollActionAutoscrollDown 102	The view was automatically scrolled downward.
	.vcScrollActionAutoscrollLeft 101	The view was automatically scrolled towards the right.
	.vcScrollActionAutoscrollRight 102	The view was automatically scrolled towards the left.
	.vcScrollActionAutoscrollUp 101	The view was automatically scrolled upward.
	.vcScrollActionMouseWheelDown 106	While the mouse wheel was pressed, the mouse was moved downward.
	.vcScrollActionMouseWheelLeft 105	While the mouse wheel was pressed, the mouse was moved towards the left.
	.vcScrollActionMouseWheelRight 106	While the mouse wheel was pressed, the mouse was moved towards the right.
	.vcScrollActionMouseWheelUp 105	While the mouse wheel was pressed, the mouse was moved upward.

<code>.vcScrollActionSBLineDown 1</code>	The view was automatically scrolled to its bottom limit
<code>.vcScrollActionSBLineLeft 0</code>	The view was automatically scrolled to its left limit
<code>.vcScrollActionSBLineRight 1</code>	The view was automatically scrolled to its right limit
<code>.vcScrollActionSBLineUp 0</code>	The view was automatically scrolled to its top limit
<code>.vcScrollActionSBNothing -1</code>	The view was not scrolled
<code>.vcScrollActionSBPageDown 3</code>	The view was scrolled downward by a page
<code>.vcScrollActionSBPageLeft 2</code>	The view was scrolled towards the left by a page
<code>.vcScrollActionSBPageRight 3</code>	The view was scrolled towards the right by a page
<code>.vcScrollActionSBPageUp 2</code>	The view was scrolled upward by a page
<code>.vcScrollActionSBThumbPosition 4</code>	The scrolling by a step has been finished.
<code>.vcScrollActionSBThumbTrack 5</code>	The view was scrolled by a step
<code>.vcScrollActionScrollEnd 104</code>	Scrolling via the End button or the context menu to the diagram end (right down)
<code>.vcScrollActionScrollHome 103</code>	Scrolling via the Pos 1 button or the context menu to the upper left corner of the diagram
<code>.vcScrollActionThumbTrackDown 108</code>	Thumb (bar of the scrollbar) moved down
<code>.vcScrollActionThumbTrackLeft 107</code>	Thumb (bar of the scrollbar) moved toward the left
<code>.vcScrollActionThumbTrackRight 108</code>	Thumb (bar of the scrollbar) moved toward the right
<code>.vcScrollActionThumbTrackUp 107</code>	Thumb (bar of the scrollbar) moved up

Example Code VB.NET

```
Private Sub VcGantt1_VcDiagramHorizontalScrolled(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcDiagramHorizontalScrolledEventArgs) Handles
VcGantt1.VcDiagramHorizontalScrolled
    MsgBox(e.CurStartDate + e.CurEndDate)
End Sub
```

Example Code C#

```
private void vcGantt1_VcDiagramHorizontalScrolled(object sender,
NETRONIC.XGantt.VcDiagramHorizontalScrolledEventArgs e)
{
    MessageBox.Show(e.CurStartDate.ToString() + "\r\n" +
e.CurEndDate.ToString());
}
```

VcDiagramHorizontalScrolling

Event of VcGantt

This event occurs when you have ordered a scroll action, but before the integrated scrolling process is performed. The old start and end date of the visible diagram area are returned. The **scrollAction** parameter provides information about the type of the performed scrolling process. If you set the returnStatus to **vcRetStatFalse**, the integrated scrolling process will be

suppressed, and in your application, you can react to the event with your own solution.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDiagramHorizontalScrollingEventArgs	Object specific to the event that is being handled

Properties of the VcDiagramHorizontalScrollingEventArgs object

	Data Type	Explanation
Properties:		
⇒ curStartDate	System.DateTime	Current start date of the visible part of the diagram
⇒ curEndDate	System.DateTime	Current end date of the visible part of the diagram
⇒ scrollAction	VcScrollAction	Scrolling type
	Possible Values:	
	.vcScrollActionAutoscrollDown 102	The view was automatically scrolled downward.
	.vcScrollActionAutoscrollLeft 101	The view was automatically scrolled towards the right.
	.vcScrollActionAutoscrollRight 102	The view was automatically scrolled towards the left.
	.vcScrollActionAutoscrollUp 101	The view was automatically scrolled upward.
	.vcScrollActionMouseWheelDown 106	While the mouse wheel was pressed, the mouse was moved downward.
	.vcScrollActionMouseWheelLeft 105	While the mouse wheel was pressed, the mouse was moved towards the left.
	.vcScrollActionMouseWheelRight 106	While the mouse wheel was pressed, the mouse was moved towards the right.
	.vcScrollActionMouseWheelUp 105	While the mouse wheel was pressed, the mouse was moved upward.
	.vcScrollActionSBLineDown 1	The view was automatically scrolled to its bottom limit
	.vcScrollActionSBLineLeft 0	The view was automatically scrolled to its left limit
	.vcScrollActionSBLineRight 1	The view was automatically scrolled to its right limit
	.vcScrollActionSBLineUp 0	The view was automatically scrolled to its top limit
	.vcScrollActionSBNothing -1	The view was not scrolled
	.vcScrollActionSBPageDown 3	The view was scrolled downward by a page
	.vcScrollActionSBPageLeft 2	The view was scrolled towards the left by a page
	.vcScrollActionSBPageRight 3	The view was scrolled towards the right by a page
	.vcScrollActionSBPageUp 2	The view was scrolled upward by a page
	.vcScrollActionSBThumbPosition 4	The scrolling by a step has been finished.

⇔ returnStatus	.vcScrollActionSBThumbTrack 5	The view was scrolled by a step
	.vcScrollActionScrollEnd 104	Scrolling via the End button or the context menu to the diagram end (right down)
	.vcScrollActionScrollHome 103	Scrolling via the Pos 1 button or the context menu to the upper left corner of the diagram
	.vcScrollActionThumbTrackDown 108	Thumb (bar of the scrollbar) moved down
	.vcScrollActionThumbTrackLeft 107	Thumb (bar of the scrollbar) moved toward the left
	.vcScrollActionThumbTrackRight 108	Thumb (bar of the scrollbar) moved toward the right
	.vcScrollActionThumbTrackUp 107	Thumb (bar of the scrollbar) moved up
	VcReturnStatus	Return status
	Possible Values:	
	.vcRetStatDefault 2	The default behavior remains unchanged.
	.vcRetStatFalse 0	The default behavior will not be performed.
	.vcRetStatNoPopup 4	The popup of the context menu is inhibited.
	.vcRetStatOK 1	The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcDiagramHorizontalScrolling(ByVal sender As Object, ByVal e As NETRONIC.XGantt.VcDiagramHorizontalScrollingEventArgs) Handles VcGantt1.VcDiagramHorizontalScrolling
    If e.CurStartDate > "01.01.2014" Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcDiagramHorizontalScrolling(object sender, NETRONIC.XGantt.VcDiagramHorizontalScrollingEventArgs e)
{
    if (DateTime.Compare(e.CurStartDate, Convert.ToDateTime("01.05.14 00:00:00")).Equals(true))
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcDiagramLeftClicking

Event of VcGantt

This event occurs when the user clicks the left mouse button on the diagram in an empty space. The position of the mouse (x,y-coordinates) is returned.

	Data Type	Explanation
Parameter:		
⇔ sender	VcGantt	Reference to the object that triggered the event
⇔ e	VcDiagramClickingEventArgs	Object specific to the event that is being handled

Properties of the VcDiagramClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcDiagramLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcDiagramClickingEventArgs) Handles
VcGantt1.VcDiagramLeftClicking
    MsgBox("x: " + e.X.ToString() + " y: " + e.Y.ToString())
End Sub
```

Example Code C#

```
private void vcGantt1_VcDiagramLeftClicking(object sender,
NETRONIC.XGantt.VcDiagramClickingEventArgs e)
{
    MessageBox.Show("x: " + e.X.ToString() + " y: " + e.Y.ToString());
}
```

VcDiagramLeftDoubleClicking

Event of VcGantt

This event occurs when the user double-clicks the left mouse button on the diagram in an empty space. The position of the mouse (x,y-coordinates) is returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDiagramClickingEventArgs	Object specific to the event that is being handled

Properties of the VcDiagramClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcDiagramLeftDoubleClicking(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcDiagramClickingEventArgs) Handles
VcGantt1.VcDiagramLeftDoubleClicking
    VcGantt1.Zoom(90)
End Sub
```

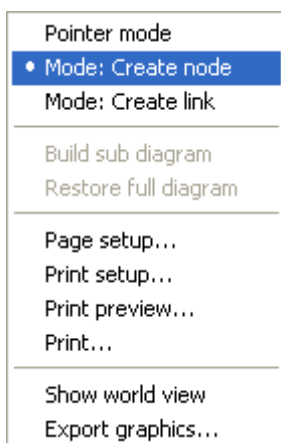
Example Code C#

```
private void vcGantt1_VcDiagramLeftDoubleClicking(object sender,
NETRONIC.XGantt.VcDiagramClickingEventArgs e)
{
    vcGantt1.Zoom(90);
}
```

VcDiagramRightClicking

Event of VcGantt

This event occurs when the user clicks the right mouse button on the diagram in an empty space. The position of the mouse (x,y-coordinates) is returned. By setting the return status you can inhibit the integrated context menu to pop up or replace it by a context menu of your own at the location delivered.



Above: integrated context menu

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDiagramClickingEventArgs	Object specific to the event that is being handled

Properties of the VcDiagramClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatNoPopup 4 .vcRetStatOK 1	The context menu will be inhibited. The context menu will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcDiagramRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcDiagramClickingEventArgs) Handles
VcGantt1.VcDiagramRightClicking
    PopupMenu.Show(VcGantt1, New Point(e.X, e.Y))
    e.ReturnStatus = NETRONIC.XGantt.VcReturnStatus.vcRetStatNoPopup
End Sub
```

Example Code C#

```
private void vcGantt1_VcDiagramRightClicking(object sender,
NETRONIC.XGantt.VcDiagramClickingEventArgs e)
{
    PopupMenu.Show(vcGantt1, new Point (e.X, e.Y));
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

VcDragCompleting

Event of VcGantt

This event is triggered at the source component to finish a drag&drop operation. It announces the drop effect.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDragCompletingEventArgs	Object specific to the event that is being handled

Properties of the VcDragCompletingEventArgs object

	Data Type	Explanation
Properties:		
⇒ DropEffect	System.Windows.Forms.DragDropEffects	Effects of a drag and drop operation

VcDragOver

Event of VcGantt

This event occurs when data are dragged onto a target.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDragEventArgs	Object specific to the event that is being handled

Properties of the VcDragEventArgs object

	Data Type	Explanation
Properties:		
⇒ AllowedEffects	System.Windows.Forms.DragDropEffects	Retrieves, which drag-and-drop operations are allowed by the originator (or source) of the drag event.
⇒ Data	System.Windows.Forms.IDataObject	Retrieves the IDataObject that contains the data associated with this event.
⇒ Effect	System.Windows.Forms.DragDropEffects	Retrieves or sets the target drop effect in a drag-and-drop operation.
⇒ KeyState	System.Int32	Gets the current state of the SHIFT, CTRL, and ALT keys, as well as the state of the mouse buttons.
⇒ X	System.Int32	Retrieves the x-coordinate of the mouse pointer, in screen coordinates.
⇒ Y	System.Int32	Retrieves the y-coordinate of the mouse pointer, in screen coordinates.
⇒ Start	System.DateTime	Retrieves the start date that appears in the tooltip of the node object during dragging.
⇒ End	System.DateTime	Retrieves the end date that appears in the tooltip of the node object during dragging.

VcDragStarting

Event of VcGantt

This event lets you specify and thus, if necessary, limit the allowed DropEffects on the start of a drag-operation. In addition, the property **LeavingControlWhileDraggingAllowed** has to be set to **True**. The property is preset to the combined value **DragDropEffects.Copy Or DragDropEffects.Move**. If, for instance, a node is always to be copied and not to be moved when being dragged out of the control, the property has to be set to **DragDropEffects.Copy**.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcDragStartingEventArgs	Object specific to the event that is being handled

Properties of the VcDragStartingEventArgs object

	Data Type	Explanation
Properties:		
⇒ allowedEffects	System.Windows.Forms.AllowedEffects	Allowed DropEffects

VcErrorOccurring

Event of VcGantt

This event occurs when an unexpected error occurs in the code of VARCHART XGantt. NETRONIC tries to avoid errors in its products; if still one occurs, this event will store it to a log file on the customer's computer and will notify the user in a convenient way. The parameter profile is provided by the ActiveX default, so some of the parameters that are passed are constant. The number of the event should always be checked, in order to prevent blocking all error types in the future program development.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcErrorOccurringEventArgs	Object specific to the event that is being handled

Properties of the VcErrorOcurringEventArgs object

	Data Type	Explanation
Properties:		
⇒ ReturnStatus	System.Boolean	If the ReturnStatus is set to vcRetStatFalse , the popping up of the message box is suppressed.
⇒ Text	System.String	Error description

VcFieldSelecting

Event of VcGantt

This event occurs, if a cell in a table or a field in a box was selected. The selection can be inhibited by setting the return status.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcFieldSelectingEventArgs	Object specific to the event that is being handled

Properties of the VcFieldSelectingEventArgs object

	Data Type	Explanation
Properties:		
⇒ editObject	VcObject	Object edited
⇒ editObjectType	VcObjectType	Object type
	Possible Values: .vcObjTypeBox 15 .vcObjTypeCalendarGrid 18 .vcObjTypeCurve 12 .vcObjTypeDateLine 9 .vcObjTypeGroup 7 .vcObjTypeGroupInDiagram 11 .vcObjTypeGroupInTable 7 .vcObjTypeHistogram 13 .vcObjTypeLayer 8 .vcObjTypeLinkCollection 3 .vcObjTypeNodeInDiagram 2 .vcObjTypeNodeInLegend 17 .vcObjTypeNodeInTable 1 .vcObjTypeNone 0 .vcObjTypeNumericScale 10 .vcObjTypeSummaryNode 14 .vcObjTypeTable 4 .vcObjTypeTableCaption 5 .vcObjTypeTimeScale 6	object type box object type calendar grid object type curve object type date line object type group object type group in diagram area object type group in table area object type histogram object type layer object type link collection object type node in diagram area object type node in legend area object type node in table area no object object type numeric scale object type summary bar object type table object type table caption object type time scale

⇒ fieldIndex	System.Int32	Field index
⇒ objRectComplete	VcRect	Complete rectangle of the object hit
⇒ objRectVisible	VcRect	Visible rectangle of the object hit
⇒ fldRectComplete	VcRect	Complete rectangle of the field hit
⇒ fldRectVisible	VcRect	Visible rectangle of the field hit
returnStatus	VcReturnStatus	
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The field will not be selected. The field will be selected.

VcGroupDeleting

Event of VcGantt

This event occurs when the user deletes a group. It returns the group object. If you set the return status to **vcRetStatFalse**, the deleting operation will be revoked. The user can delete only those groups that do not contain any elements.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcGroupDeletingEventArgs	Object specific to the event that is being handled

Properties of the VcGroupDeletingEventArgs object

	Data Type	Explanation
Properties:		
⇒ group	VcGroup	Group deleted
↔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The group will not be deleted. The group will be deleted.

Example Code VB.NET

```
Private Sub VcGantt1_VcGroupDeleting(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcGroupDeletingEventArgs) Handles VcGantt1.VcGroupDeleting
    If e.Group.Name = "A" Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
        MsgBox("Group A cannot be deleted")
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcGroupDeleting(object sender,
NETRONIC.XGantt.VcGroupDeletingEventArgs e)
{
    if (e.Group.Name == "A")
    {
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
        MessageBox.Show("Group A cannot be deleted");
    }
}
```

VcGroupLeftClicking**Event of VcGantt**

This event occurs when the user clicks the left mouse button on a group heading in the table. The group object and the mouse position (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcGroupClickingEventArgs	Object specific to the event that is being handled

Properties of the VcGroupClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ group	VcGroup	Group hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcGroupLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcGroupClickingEventArgs) Handles VcGantt1.VcGroupLeftClicking
    MsgBox(e.Group.SubGroups.Count)
End Sub
```


Example Code C#

```
private void vcGantt1_VcGroupLeftClicking(object sender,
NETRONIC.XGantt.VcGroupClickingEventArgs e)
{
    MessageBox.Show(e.Group.SubGroups.Count.ToString());
}
```

VcGroupLeftDoubleClicking**Event of VcGantt**

This event occurs when the user double-clicks the left mouse button on a group heading in the table. The group object and the mouse position (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcGroupClickingEventArgs	Object specific to the event that is being handled

Properties of the VcGroupClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ group	VcGroup	Group hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcGroupLeftDoubleClicking(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcGroupClickingEventArgs) Handles
VcGantt1.VcGroupLeftDoubleClicking
    MsgBox(e.Group.Name)
End Sub
```

Example Code C#

```
private void vcGantt1_VcGroupLeftDoubleClicking(object sender,
NETRONIC.XGantt.VcGroupClickingEventArgs e)
{
    MessageBox.Show(e.Group.Name);
}
```

VcGroupModified**Event of VcGantt**

This event occurs when the modification of the group is finished.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcGroupModifiedEventArgs	Object specific to the event that is being handled

Properties of the VcGroupModifiedEventArgs object

	Data Type	Explanation
Properties:		
⇒ group	VcGroup	Group modified

Example Code VB.NET

```
Private Sub VcGantt1_VcGroupModified(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcGroupModifiedEventArgs) Handles VcGantt1.VcGroupModified
    MsgBox("The group has been modified.")
End Sub
```

Example Code C#

```
private void vcGantt1_VcGroupModified(object sender,
NETRONIC.XGantt.VcGroupModifiedEventArgs e)
{
    MessageBox.Show("The group has been modified");
}
```

VcGroupModifying**Event of VcGantt**

This event occurs when a user interactively modifies a group. The group object, the type of modification and the return status are returned. By the **modificationType** parameter you can obtain more detailed information of the type of modification.

The data passed by this event can be read, but must not be modified. For modifying them please use the event **VcGroupModified**.

By setting the return status the modification can be inhibited.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcGroupModifyingEventArgs	Object specific to the event that is being handled

Properties of the VcGroupModifyingEventArgs object

	Data Type	Explanation
Properties:		
⇒ Rückgabewert	Void	
⇒ oldGroup	VcGoup	Group before the modification
⇒ group	VcGroup	Group modified
⇒ group	VcGroup	Group to be modified
⇒ modificationType	VcGroupModificationTypes	Type of modification
	Possible Values: .vcGMTAnything 1 .vcGMTMinusPressed 2 .vcGMTNothing 0 .vcGMTPlusPressed 4	Modification type not determined Modification type Minus symbol clicked on Modification type nothing Modification type Plus symbol clicked on
⇒ modificationType	VcModificationTypes	Type of modification
	Possible Values: .vcAnything 1 .vcChangedGroup 16 .vcEndModified 4 .vcHierarchyModified 64 .vcModifiedByResourceScheduling 128 .vcModifiedBySchedule 32 .vcMoved 8 .vcNothing 0 .vcStartModified 2	Modification type cannot be identified. Group of the node was changed (occurs with nodes only). The end date of the node was modified (occurs with nodes only). Hierarchy of the nodes has been changed Modification by resource scheduling (occurs with data records only) Modification by new date calculation Object was moved. No modification The start date of the node was modified (occurs with nodes only).
⇒ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The modification will be revoked. The modification will be accepted.

⇔ returnStatus	VcReturnStatus	Return status
	Possible Values:	
	.vcRetStatDefault 2	The default behavior remains unchanged.
	.vcRetStatFalse 0	The default behavior will not be performed.
	.vcRetStatNoPopup 4	The popup of the context menu is inhibited.
	.vcRetStatOK 1	The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcGroupModifying(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcGroupModifyingEventArgs) Handles VcGantt1.VcGroupModifying
    Select Case e.ModificationType
        Case VcGroupModificationTypes.vcGMTNothing
            MsgBox("No modification")
        Case VcGroupModificationTypes.vcGMTAnything
            MsgBox("Any modification")
        Case VcGroupModificationTypes.vcGMTMinusPressed
            MsgBox("Collapsing group:" + e.Group.Name)
        Case VcGroupModificationTypes.vcGMTPlusPressed
            MsgBox("Expanding group" + e.Group.Name)
    End Select
End Sub
```

Example Code C#

```
private void vcGantt1_VcGroupModifying(object sender,
NETRONIC.XGantt.VcGroupModifyingEventArgs e)
{
    switch (e.ModificationType)
    {
        case VcGroupModificationTypes.vcGMTNothing:
            MessageBox.Show("No modification");
            break;
        case VcGroupModificationTypes.vcGMTAnything:
            MessageBox.Show("Any modification");
            break;
        case VcGroupModificationTypes.vcGMTMinusPressed:
            MessageBox.Show("Collapsing group: " + e.Group.Name);
            break;
        case VcGroupModificationTypes.vcGMTPlusPressed:
            MessageBox.Show("Expanding group: " + e.Group.Name);
            break;
    }
}
```

VcGroupRightClicking

Event of VcGantt

This event occurs when the user clicks the right mouse button on a group heading in the table. The group object and the mouse position (x,y-coordinates) are returned. By setting the return status you can inhibit the integrated context menu to pop up and replace it by a context menu of your own at the location delivered.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcGroupClickingEventArgs	Object specific to the event that is being handled

Properties of the VcGroupClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ group	VcGroup	Group hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatNoPopup 4 .vcRetStatOK 1	The context menu will be inhibited. The context menu will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcGroupRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcGroupClickingEventArgs) Handles VcGantt1.VcGroupRightClicking
    PopupMenu.Show(VcGantt1, New Point(e.X, e.Y))
    e.ReturnStatus = NETRONIC.XGantt.VcReturnStatus.vcRetStatNoPopup
End Sub
```

Example Code C#

```
private void vcGantt1_VcGroupRightClicking(object sender,
NETRONIC.XGantt.VcGroupClickingEventArgs e)
{
    PopupMenu.Show(vcGantt1, new Point (e.X, e.Y));
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

VcGroupsMarked

Event of VcGantt

This event occurs after the operation of marking or unmarking groups was finished.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcGroupsMarkedEventArgs	Object specific to the event that is being handled

Properties of the VcGroupsMarkedEventArgs object

	Data Type	Explanation
Properties:		
⇐ (no parameter)		No parameter

Example Code VB.NET

```
Private Sub VcGantt1_VcGroupsMarked(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcGroupsMarkedEventArgs) Handles VcGantt1.VcGroupsMarked
    MsgBox("Groups have been marked successfully.")
End Sub
```

Example Code C#

```
private void vcGantt1_VcGroupsMarked(object sender,
NETRONIC.XGantt.VcGroupsMarkedEventArgs e)
{
    MessageBox.Show("Groups have been marked successfully.");
}
```

VcGroupsMarking

Event of VcGantt

This event occurs when the user selects groups for marking or when he unmarks marked groups by a click into the empty diagram. The GroupCollection contains the groups selected by the most recent marking action of the user. If the user unmarks groups by a click into the empty diagram, the group collection will be empty.

If you set the return status to **vcRetStatFalse**, you have to mark or unmark groups yourself.

The data passed by this event can be read, but must not be modified. For modifying them please use **VcGroupsMarked**.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNodesMarkingEventArgs	Object specific to the event that is being handled

Properties of the VcNodesMarkingEventArgs object

	Data Type	Explanation
Properties:		
⇒ groupCollection	VcGroupCollection	GroupCollection that contains the nodes selected by the user. If the user has clicked in the diagram, the GroupCollection is empty.
⇒ returnStatus	VcReturnStatus	Return status

Example Code VB.NET

```
Private Sub VcGantt1_VcGroupsMarking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcGroupsMarkingEventArgs) Handles VcGantt1.VcGroupsMarking
    If MsgBox("Mark this group?", MsgBoxStyle.YesNo, "Marking groups") =
MsgBoxResult.No Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcGroupsMarking(object sender,
NETRONIC.XGantt.VcGroupsMarkingEventArgs e)
{
    DialogResult retVal = MessageBox.Show("Mark this group?", "Marking groups",
MessageBoxButtons.YesNo);
    if (retVal == DialogResult.No)
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcHelpRequested**Event of VcGantt**

This event occurs if the user presses the F1 key on a dialog at run time. The application can invoke its own help system, to offer help specific to the dialog and to the application.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcHelpRequestedEventArgs	Object specific to the event that is being handled

Properties of the VcHelpRequestedEventArgs object

	Data Type	Explanation
Properties:		
⇒ DialogType	VcDialogType	Dialog for which help was requested
	Possible Values: .vcEditDataRecordDialog 5400	Help was requested for the Edit Data Record dialog.

.vcEditTimeScaleDialog 5409	Help was requested for the Edit Time Scale dialog.
.vcPageSetupDialog 4097	Help was requested for the Page Set Up dialog.
.vcPrintPreviewDialog 4096	Help was requested for the Print Preview dialog.

VcHistogramCurveNameShowingInMenu

Event of VcGantt

This event occurs when the names of histogram curves defined by the API are displayed in a context menu. If you set the returnStatus to **vcRetStat-False**, the names of the histogram curves are not displayed in a context menu.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcHistogramCurveNameShowingInMenuEventArgs	Object specific to the event that is being handled

Properties of the VcHistogramCurveNameShowingInMenuEventArgs object

	Data Type	Explanation
Properties:		
⇒ histogram	VcHistogram	Histogram hit
⇒ curveName	System.String	Name of the histogram curve
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcHistogramCurveNameShowingInMenu(ByVal sender As Object,
ByVal e As NETRONIC.XGantt.VcHistogramCurveNameShowingInMenuEventArgs) Handles
VcGantt1.VcHistogramCurveNameShowingInMenu
    e.ReturnStatus = VcReturnStatus.vcRetStatFalse
End Sub
```


Example Code C#

```
private void vcGantt1_VcHistogramCurveNameShowingInMenu(object sender,
NETRONIC.XGantt.VcHistogramCurveNameShowingInMenuEventArgs e)
{
    e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcHistogramLeftClicking**Event of VcGantt**

This event occurs when the user clicks the left mouse button on a histogram. The histogram object and the mouse position (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcHistogramClickingEventArgs	Object specific to the event that is being handled

Properties of the VcHistogramClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ histogram	VcHistogram	Histogram hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇌ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcHistogramLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcHistogramClickingEventArgs) Handles
VcGantt1.VcHistogramLeftClicking
    Call MsgBox("Histogram:" + e.Histogram.Name + " x:" + e.X.ToString() + " y: "
+ e.Y.ToString())
End Sub
```

Example Code C#

```
private void vcGantt1_VcHistogramLeftClicking(object sender,
NETRONIC.XGantt.VcHistogramClickingEventArgs e)
{
    MessageBox.Show("Histogram: " + e.Histogram.Name + " x: " + e.X.ToString() +
" y: " + e.Y.ToString());
}
```

VcHistogramLeftDoubleClicking**Event of VcGantt**

This event occurs when the user double-clicks the left mouse button on a histogram. The histogram object and the mouse position (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcHistogramClickingEventArgs	Object specific to the event that is being handled

Properties of the VcHistogramClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ histogram	VcHistogram	Histogram hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcHistogramLeftDoubleClicking(ByVal sender As Object, ByVal
e As NETRONIC.XGantt.VcHistogramClickingEventArgs) Handles
VcGantt1.VcHistogramLeftDoubleClicking
    MsgBox(e.Histogram.Name)
End Sub
```

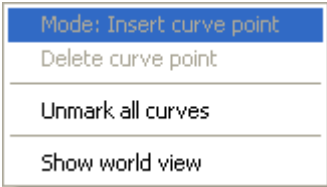
Example Code C#

```
private void vcGantt1_VcHistogramLeftDoubleClicking(object sender,
NETRONIC.XGantt.VcHistogramClickingEventArgs e)
{
    MessageBox.Show(e.Histogram.Name);
}
```

VcHistogramRightClicking

Event of VcGantt

This event occurs when the user clicks the right mouse button on a histogram. The histogram object and the mouse position (x,y-coordinates) are returned. By setting the return status you can inhibit the integrated context menu to pop up and replace it by a context menu of your own at the location delivered.



Above: integrated context menu

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcHistogramClickingEventArgs	Object specific to the event that is being handled

Properties of the VcHistogramClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ histogram	VcHistogram	Histogram hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatNoPopup 4 .vcRetStatOK 1	The context menu will be inhibited. The context menu will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcHistogramRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcHistogramClickingEventArgs) Handles
VcGantt1.VcHistogramRightClicking
    PopupMenu.Show(VcGantt1, New Point(e.X, e.Y))
    e.ReturnStatus = NETRONIC.XGantt.VcReturnStatus.vcRetStatNoPopup
End Sub
```

Example Code C#

```
private void vcGantt1_VcHistogramRightClicking(object sender,
NETRONIC.XGantt.VcHistogramClickingEventArgs e)
{
    PopupMenu.Show(vcGantt1, new Point (e.X, e.Y));
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

VcHistogramsHeightChanged

Event of VcGantt

This event occurs after the ratio of the diagram height to the histogram height modified by the user was changed. The collection of the histograms and the diagram / histogram height ratio are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcHistogramsHeightChangedEventArgs	Object specific to the event that is being handled

Properties of the VcHistogramsHeightChangedEventArgs object

	Data Type	Explanation
Properties:		
⇒ histogramCollection	VcHistogramCollection	Histogram collection
⇒ histogramsHeightRatio	System.Int32	Height ratio of the histograms to the complete diagram

VcHistogramsHeightChanging

Event of VcGantt

This event occurs when the user modifies the ratio of the diagram height to the histogram height. The collection of the histograms and the diagram /

histogram height ratio are returned. If you set the return status to `vcRetStat-False`, the modification will be revoked.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcHistogramsHeightChangingEventArgs	Object specific to the event that is being handled

Properties of the VcHistogramsHeightChangingEventArgs object

	Data Type	Explanation
Properties:		
⇒ histogramCollection	VcHistogramCollection	Histogram collection
⇒ histogramsHeightRatio	System.Int32	Height ratio of the histograms to the complete diagram
↔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The height will not change. The height will change.

VcHistogramsHeightChangingEx

Event of VcGantt

This event occurs when the user interactively modifies the height of a histogram. The histograms and the modified diagram/histogram aspect ratio are returned. By setting the return status you can inhibit the modification.

In contrast to the **VcHistogramHeightChanging** event this event returns the parameter *histogramHeightRatio* as a "Double" value, thus achieving a higher level of accuracy. The use of this event has to be enabled by the **Use-HigherDiagramHistogramHeightRatioPrecision** property or by activating the corresponding option on the **General** property page.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	
⇒ e	VcHistogramsHeightChangingExEventArgs	

	Data Type	Explanation

VcInPlaceEditorShowing

Event of VcGantt

This event occurs when the implemented editor is started.

The event will be activated only if the corresponding properties **<InPlaceEditingOnGroupsInDiagramEnabled, InPlaceEditingOnGroupsInTableEnabled, InPlaceEditingOnNodesInDiagramEnabled, InPlaceEditingOnNodesInTableEnabled>** are set to True.

By setting the return status to **False** this event can be inhibited so that your own editor can be started at the coordinates passed.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcInPlaceEditorShowingEventArgs	Object specific to the event that is being handled

Properties of the VcInPlaceEditorShowingEventArgs object

	Data Type	Explanation
Properties:		
⇒ editObject	VcObject	Object edited
⇒ editObjectType	VcObjectType	Object type
	Possible Values: .vcObjTypeBox 15 .vcObjTypeNodeInLegend 17 .vcObjTypeNodeInTable 1 .vcObjTypeNone 0	object type box object type node in legend area object type node in table area no object
⇒ fieldIndex	System.Int32	Field index
⇒ objRectComplete	VcRect	Complete rectangle of the object hit

⇒ objRectVisible	VcRect	Visible rectangle of the object hit
⇒ fldRectComplete	VcRect	Complete rectangle of the field hit
⇒ fldRectVisible	VcRect	Visible rectangle of the field hit
returnStatus	VcReturnStatus	
	Possible Values:	
	.vcRetStatDefault 2	The default behavior remains unchanged.
	.vcRetStatFalse 0	The default behavior will not be performed.
	.vcRetStatNoPopup 4	The popup of the context menu is inhibited.
	.vcRetStatOK 1	The default behavior will be performed.

Example Code VB.NET

```

Private Sub VcGantt1_VcInPlaceEditorShowing(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcInPlaceEditorShowingEventArgs) Handles
VcGantt1.VcInPlaceEditorShowing

    Dim node As VcNode
    node = e.EditObject
    If e.EditObjectType = VcObjectType.vcObjTypeNodeInTable Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
        Select Case e.FieldIndex
            Case 1 'Name
                TextBox1.Left = e.FldRectVisible.Left + VcGantt1.Left
                TextBox1.Top = e.FldRectVisible.Top + VcGantt1.Top
                TextBox1.Width = e.FldRectVisible.Width
                TextBox1.Height = e.FldRectVisible.Height
                TextBox1.Text = node.DataField(0)
                TextBox1.Visible = True
                TextBox1.Focus()
            Case 2, 3 'Start or End
                DateTimePicker1.Left = e.FldRectVisible.Left + VcGantt1.Left
                DateTimePicker1.Top = e.FldRectVisible.Top + VcGantt1.Top
                DateTimePicker1.Value = node.DataField(0)
                DateTimePicker1.Visible = True
                DateTimePicker1.Focus()
            Case 13 'Employee
                ComboBox1.Left = e.FldRectVisible.Left + VcGantt1.Left
                ComboBox1.Top = e.FldRectVisible.Top + VcGantt1.Top
                ComboBox1.Width = e.FldRectVisible.Width
                ComboBox1.Height = e.FldRectVisible.Height
                ComboBox1.Text = node.DataField(0)
                ComboBox1.Visible = True
                ComboBox1.Focus()
        End Select
    End If

```

Example Code C#

```

private void vcGantt1_VcInPlaceEditorShowing(object sender,
NETRONIC.XGantt.VcInPlaceEditorShowingEventArgs e)
{
    VcNode node = (VcNode)e.EditObject;
    if (e.EditObjectType == VcObjectType.vcObjTypeNodeInTable)
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
    switch (e.FieldIndex)
    {
        case 1: //Name
            textBox1.Left = e.FldRectVisible.Left + vcGantt1.Left;
            textBox1.Top = e.FldRectVisible.Top + vcGantt1.Top;
            textBox1.Width = e.FldRectVisible.Width;
            textBox1.Height = e.FldRectVisible.Height;
            textBox1.Text = Convert.ToString(node.get_DataField(0));
            textBox1.Visible = true;
            textBox1.Focus();
            break;
        case 2: //Start or end
            dateTimePicker1.Left = e.FldRectVisible.Left + vcGantt1.Left;
            dateTimePicker1.Top = e.FldRectVisible.Top + vcGantt1.Top;
            dateTimePicker1.Value = Convert.ToDateTime(node.get_DataField(0));
            dateTimePicker1.Visible = true;
            dateTimePicker1.Focus();
            break;
        case 13: //Employee
            comboBox1.Left = e.FldRectVisible.Left + vcGantt1.Left;
            comboBox1.Top = e.FldRectVisible.Top + vcGantt1.Top;
            comboBox1.Width = e.FldRectVisible.Width;
            comboBox1.Height = e.FldRectVisible.Height;
            comboBox1.Text = Convert.ToString(node.get_DataField(0));
            comboBox1.Visible = true;
            comboBox1.Focus();
            break;
    }
}

```

VcInteractionEnded**Event of VcGantt**

This event occurs on ending an interaction (LiveUpdate switched on).

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	
⇒ e	VcInteractionEndedEventArgs	

	Data Type	Explanation
Properties:		
⇒ InInteractionMode	VcInInteractionMode	Mode of interaction
	Possible Values:	

	.vcIIMCopyMoveNode 1014	Move copied node
	.vcIIMCopyNode 1007	Copy node
	.vcIIMCreateLinkChangeSuccessor 1101	Change successor
	.vcIIMCreateNodeResizeRightX 1012	Modify start date of layer
	.vcIIMCreateResizeObjectContainerWidthHeight 1072	Modify size of textbox
	.vcIIMDragDropNode 1018	Drag and drop node
	.vcIIMDragDropNodeInTable 1019	Move node in table by drag and drop
	.vcIIMModifySectionStartDate 1061	Modify start date of time scale section
	.vcIIMMoveCurvePointX 1052	Move curve point x
	.vcIIMMoveCurvePointXandY 1051	Move curve points x and y
	.vcIIMMoveCurvePointY 1053	Move curve point y
	.vcIIMMoveGroupInDiagram 1100	Group in diagram is moved
	.vcIIMMoveGroupInTable 1009	Move group in table
	.vcIIMMoveHorValueLine 1031	Move date line horizontally
	.vcIIMMoveLayer 1004	Move layer
	.vcIIMMoveNode 1001	Move node
	.vcIIMMoveNode 1001	Move node by drag and drop
	.vcIIMMoveNodeInRow 1002	Move node in row
	.vcIIMMoveNodeInTable 1008	Move node in table
	.vcIIMMoveNodeVertical 1003	Move node vertically
	.vcIIMMoveObjectContainer 1073	Move textbox
	.vcIIMMoveSash 1026	Move sash
	.vcIIMResizeBasicUnitWidth 1062	Modify basic unit width
	.vcIIMResizeLeftX 1005	Modify start date of layer
	.vcIIMResizeNumericBasicUnitWidth 1063	Modify numeric basic unit width
	.vcIIMResizeObjectContainerHeight 1075	Modify height of textbox
	.vcIIMResizeObjectContainerWidth 1074	Modify width of text box
	.vcIIMResizeObjectContainerWidthHeight 1076	Modify width and height of textbox
	.vcIIMResizeRightX 1006	Modify end date of layer
	.vcIIMUnKnown -1	Usually not returned by eventargs, but can be used e.g. for indicating a variable as not having been set
	.vcIIMvcIIMResizeLeftTableColumnWidth 1041	Modify column width of left table
	.vcIIMvcIIMResizeRightTableColumnWidth 1042	Modify column width of right table
⇒ InteractionObject	InteractionObject	Object affected by the interaction
⇒ ObjectType	InteractionType	type of object affected by the interaction

VcInteractionModeChanged

Event of VcGantt

This event occurs after having changed the interaction mode in the contextmenu and the change not having been prevented by setting the return status of the event **VcInteractionModeChanging** to **vcRetStatFalse**.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	
⇒ e	VcInteractionModeChangedEventArgs	

	Data Type	Explanation

VcInteractionModeChanging

Event of VcGantt

This event occurs after having changed the interaction mode in the contextmenu.

By setting the return status to **vcRetStatFalse** the modification will not be applied and the event **VcInteractionModeChanged** will not be triggered.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	
⇒ e	VcInteractionModeChangingEventArgs	

	Data Type	Explanation
Properties:		
⇒ NewInteractionMode	VcInteractionMode	Hit histogram curve
	Possible Values:	
	.vcCreateBox 36	Box creating mode
	.vcCreateLink 4	Link creating mode
	.vcCreateNode 2	Node creating mode
	.vcDeleteLink 5	Link deleting mode
	.vcDeleteNode 3	Node deleting mode
	.vcPanning 6	Panning mode
	.vcPointer 0	Select mode
⇒ returnstatus	VcReturnStatus	return status
	Possible Values:	
	.vcRetStatDefault 2	The default behavior remains unchanged.

.vcRetStatFalse 0	The default behavior will not be performed.
.vcRetStatNoPopup 4	The popup of the context menu is inhibited.
.vcRetStatOK 1	The default behavior will be performed.

VcInteractionObjectChanged

Event of VcGantt

This event occurs when there is no object yet at the beginning of an interaction (LiveUpdate switched on; such as creating nodes or boxes) and as soon as the object has been created internally.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	
⇒ e	VcInteractionObjectChangedEventArgs	

	Data Type	Explanation
Properties:		
⇒ InInteractionMode	VcInInteractionMode	Mode of interaction
	Possible Values: .vclIMCopyMoveNode 1014 .vclIMCopyNode 1007 .vclIMCreateLinkChangeSuccessor 1101 .vclIMCreateNodeResizeRightX 1012 .vclIMCreateResizeObjectContainerWidthHight 1072 .vclIMDragDropNode 1018 .vclIMDragDropNodeInTable 1019 .vclIMModifySectionStartDate 1061 .vclIMMoveCurvePointX 1052 .vclIMMoveCurvePointXandY 1051 .vclIMMoveCurvePointY 1053 .vclIMMoveGroupInDiagram 1100 .vclIMMoveGroupInTable 1009 .vclIMMoveHorValueLine 1031 .vclIMMoveLayer 1004 .vclIMMoveNode 1001 .vclIMMoveNode 1001 .vclIMMoveNodeInRow 1002 .vclIMMoveNodeInTable 1008 .vclIMMoveNodeVertical 1003 .vclIMMoveObjectContainer 1073 .vclIMMoveSash 1026 .vclIMResizeBasicUnitWidth 1062	Move copied node Copy node Change successor Modify start date of layer Modify size of textbox Drag and drop node Move node in table by drag and drop Modify start date of time scale section Move curve point x Move curve points x and y Move curve point y Group in diagram is moved Move group in table Move date line horizontally Move layer Move node Move node by drag and drop Move node in row Move node in table Move node vertically Move textbox Move sash Modify basic unit width

	.vcIIMResizeLeftX 1005	Modify start date of layer
	.vcIIMResizeNumericBasicUnitWidth 1063	Modify numeric basic unit width
	.vcIIMResizeObjectContainerHeight 1075	Modify height of textbox
	.vcIIMResizeObjectContainerWidth 1074	Modify width of text box
	.vcIIMResizeObjectContainerWidthHeight 1076	Modify width and height of textbox
	.vcIIMResizeRightX 1006	Modify end date of layer
	.vcIIMUnKnown -1	Usually not returned by eventargs, but can be used e.g. for indicating a variable as not having been set
	.vcIIMvcIIMResizeLeftTableColumnWidth 1041	Modify column width of left table
	.vcIIMvcIIMResizeRightTableColumnWidth 1042	Modify column width of right table
⇒ InteractionObject	InteractionObject	Object affected by the interaction
⇒ ObjectType	InteractionType	type of object affected by the interaction

VcInteractionStarted

Event of VcGantt

This event occurs when an interaction is started by pressing the left mouse key (LiveUpdate switched on) and it returns information about the object the mouse has hit (object and object type).

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcInteractionStartedEventArgs	Object specific to the event that is being handled

Properties of the VcInteractionStartedEventArgs object

	Data Type	Explanation
Properties:		
⇒ InInteractionMode	VcInInteractionMode	Mode of interaction
	Possible Values:	
	.vcIIMCopyMoveNode 1014	Move copied node
	.vcIIMCopyNode 1007	Copy node
	.vcIIMCreateLinkChangeSuccessor 1101	Change successor
	.vcIIMCreateNodeResizeRightX 1012	Modify start date of layer
	.vcIIMCreateResizeObjectContainerWidthHight 1072	Modify size of textbox
	.vcIIMDragDropNode 1018	Drag and drop node
	.vcIIMDragDropNodeInTable 1019	Move node in table by drag and drop

	<code>.vcIIMModifySectionStartDate</code> 1061	Modify start date of time scale section
	<code>.vcIIMMoveCurvePointX</code> 1052	Move curve point x
	<code>.vcIIMMoveCurvePointXandY</code> 1051	Move curve points x and y
	<code>.vcIIMMoveCurvePointY</code> 1053	Move curve point y
	<code>.vcIIMMoveGroupInDiagram</code> 1100	Group in diagram is moved
	<code>.vcIIMMoveGroupInTable</code> 1009	Move group in table
	<code>.vcIIMMoveHorValueLine</code> 1031	Move date line horizontally
	<code>.vcIIMMoveLayer</code> 1004	Move layer
	<code>.vcIIMMoveNode</code> 1001	Move node
	<code>.vcIIMMoveNode</code> 1001	Move node by drag and drop
	<code>.vcIIMMoveNodeInRow</code> 1002	Move node in row
	<code>.vcIIMMoveNodeInTable</code> 1008	Move node in table
	<code>.vcIIMMoveNodeVertical</code> 1003	Move node vertically
	<code>.vcIIMMoveObjectContainer</code> 1073	Move textbox
	<code>.vcIIMMoveSash</code> 1026	Move sash
	<code>.vcIIMResizeBasicUnitWidth</code> 1062	Modify basic unit width
	<code>.vcIIMResizeLeftX</code> 1005	Modify start date of layer
	<code>.vcIIMResizeNumericBasicUnitWidth</code> 1063	Modify numeric basic unit width
	<code>.vcIIMResizeObjectContainerHeight</code> 1075	Modify height of textbox
	<code>.vcIIMResizeObjectContainerWidth</code> 1074	Modify width of text box
	<code>.vcIIMResizeObjectContainerWidthHeight</code> 1076	Modify width and height of textbox
	<code>.vcIIMResizeRightX</code> 1006	Modify end date of layer
	<code>.vcIIMUnknown</code> -1	Usually not returned by eventargs, but can be used e.g. for indicating a variable as not having been set
	<code>.vcIIMvcIIMResizeLeftTableColumnWidth</code> 1041	Modify column width of left table
	<code>.vcIIMvcIIMResizeRightTableColumnWidth</code> 1042	Modify column width of right table
⇒ InteractionObject	InteractionObject	Object affected by the interaction
⇒ ObjectType	InteractionType	type of object affected by the interaction

VcLegendViewClosed

Event of VcGantt

This event occurs when the legend view popup window is closed.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcLEgendViewClosedEventArgs	Object specific to the event that is being handled

Properties of the VcLegendViewClosedEventArgs object

	Data Type	Explanation
Properties: ⇒ (no parameter)		

Example Code VB.NET

```
Private Sub VcGantt1_VcLegendViewClosed(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcLegendViewClosedEventArgs) Handles VcGantt1.VcLegendViewClosed
    MsgBox("Do you want to close the legend view window?", MsgBoxStyle.OKCancel)
End Sub
```

Example Code C#

```
private void vcGantt1_VcLegendViewClosed(object sender,
NETRONIC.XGantt.VcLegendViewClosedEventArgs e)
{
    DialogResult retVal = MessageBox.Show("Do you want to close the legend view
window?", "Closing legend view window", MessageBoxButtons.OKCancel);
}
```

VcLinkCreated

Event of VcGantt

This event occurs when the interactive creation of a link is completed. The link object, the creation type (here **VcLinkCreated**) and the information whether the created link is the only link or the last link of a link collection (always **True**) are returned, so that a validation can be made.

	Data Type	Explanation
Parameter: ⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcLinkCreatedEventArgs	Object specific to the event that is being handled

Properties of the VcLinkCreatedEventArgs object

	Data Type	Explanation
Properties: ⇒ link	VcLink	Link created
⇒ creationType	VcCreationType	Creation type of the link
	Possible Values: .vcLinkCreated 2	Link created by interaction
⇒ isLast	System.Boolean	The created link is/is not the only link or the last link of a link collection.

Example Code VB.NET

```
Private Sub VcGantt1_VcLinkCreated(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcLinkCreatedEventArgs) Handles VcGantt1.VcLinkCreated
    MsgBox(e.Link.AllData)
End Sub
```

Example Code C#

```
private void vcGantt1_VcLinkCreated(object sender,
NETRONIC.XGantt.VcLinkCreatedEventArgs e)
{
    MessageBox.Show(e.Link.AllData.ToString());
}
```

VcLinkCreating

Event of VcGantt

This event occurs when the user creates a link between two nodes. The generated link object is returned, so that a validation and if necessary a data base entry can be made. The data passed by this event can be read, but must not be modified. For modifying them please use the event **VcLinkCreated**.

By setting the return status the create operation can be inhibited.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcLinkCreatingEventArgs	Object specific to the event that is being handled

Properties of the VcLinkCreatingEventArgs object

	Data Type	Explanation
Properties:		
⇒ link	VcLink	Link created
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The link will not be created. The link will be created.

Example Code VB.NET

```
Private Sub VcGantt1_VcLinkCreated(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcLinkCreatedEventArgs) Handles VcGantt1.VcLinkCreated
    MsgBox(e.Link.AllData)
End Sub
```

Example Code C#

```
private void vcGantt1_VcLinkCreated(object sender,
NETRONIC.XGantt.VcLinkCreatedEventArgs e)
{
    MessageBox.Show(e.Link.AllData.ToString());
}
```

VcLinkDeleted**Event of VcGantt**

This event occurs when the deletion of a link is completed. The link object and the information whether the created link is the only link or the last link of a link collection are returned, so that a validation can be made.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcLinkDeletedEventArgs	Object specific to the event that is being handled

Properties of the VcLinkDeletedEventArgs object

	Data Type	Explanation
Properties:		
⇒ link	VcLink	Link deleted
⇒ isLast	System.Boolean	The deleted link is/is not the only link or the last link of a link collection.

VcLinkDeleting**Event of VcGantt**

This event occurs when a user deletes a link by the context menu. The link object to be deleted is returned, so that you can still check for - whatever - conditions and prohibit the deletion on a negative result by setting the return status.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcLinkDeletingEventArgs	Object specific to the event that is being handled

Properties of the VcLinkDeletingEventArgs object

	Data Type	Explanation
Properties:		
⇒ link	VcLink	Link deleted
↔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The link will not be deleted. The link will be deleted.

Example Code VB.NET

```
Private Sub VcGantt1_VcLinkDeleting(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcLinkDeletingEventArgs) Handles VcGantt1.VcLinkDeleting
    'deny deletion of link with a certain predecessor
    If e.Link.PredecessorNode.DataField(0) = "1" Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcLinkDeleting(object sender,
NETRONIC.XGantt.VcLinkDeletingEventArgs e)
{
    // deny deletion of link with a certain predecessor
    if (e.Link.PredecessorNode.get_DataField(0).Equals("1"))
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcLinksLeftClicking

Event of VcGantt

This event occurs when the user clicks the left mouse button on a link or on several overlapping links. The LinkCollection object and the mouse position (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcLinksClickingEventArgs	Object specific to the event that is being handled

Properties of the VcLinksClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ linkCltn	VcLinkCollection	LinkCollection object hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcLinksLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcLinksClickingEventArgs) Handles VcGantt1.VcLinksLeftClicking
    Dim linkCltn As VcLinkCollection
    Dim link As VcLink
    linkCltn = VcGantt1.LinkCollection
    'set certain data field of all links
    For Each link In linkCltn
        link.DataField(2) = "A"
    Next
End Sub
```

Example Code C#

```
private void vcGantt1_VcLinksLeftClicking(object sender,
NETRONIC.XGantt.VcLinksClickingEventArgs e)
{
    VcLinkCollection linkCltn = vcGantt1.LinkCollection;
    // set certain data field of all links
    foreach (VcLink link in linkCltn)
        link.set_DataField(2, "A");
}
```

VcLinksLeftDoubleClicking

Event of VcGantt

This event occurs when the user double-clicks the left mouse button on a link or on several overlapping links. The LinkCollection object and the mouse position (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event

⇒ e	VcLinksClickingEventArgs	Object specific to the event that is being handled
-----	--------------------------	--

Properties of the VcLinksClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ linkCltn	VcLinkCollection	LinkCollection object hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcLinksLeftDoubleClicking(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcLinksClickingEventArgs) Handles
VcGantt1.VcLinksLeftDoubleClicking
    MsgBox("Show your own dialog")
    e.ReturnStatus = VcReturnStatus.vcRetStatFalse
End Sub
```

Example Code C#

```
private void vcGantt1_VcLinksLeftDoubleClicking(object sender,
NETRONIC.XGantt.VcLinksClickingEventArgs e)
{
    MessageBox.Show("Show your own dialog");
    e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcLinksRightClicking

Event of VcGantt

This event occurs when the user clicks the right mouse button on a link or on several overlapping links. The LinkCollection object and the mouse position (x,y-coordinates) are returned. By setting the return status you can inhibit the integrated context menu to pop up and replace it by a context menu of your own at the location delivered.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event

⇒ e	VcLinksClickingEventArgs	Object specific to the event that is being handled
-----	--------------------------	--

Properties of the VcLinksClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ linkCltn	VcLinkCollection	LinkCollection object hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇒ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatNoPopup 4 .vcRetStatOK 1	The context menu will be inhibited. The context menu will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcLinksRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcLinksClickingEventArgs) Handles VcGantt1.VcLinksRightClicking
    PopupMenu.Show(VcGantt1, New Point(e.X, e.Y))
    e.ReturnStatus = NETRONIC.XGantt.VcReturnStatus.vcRetStatNoPopup
End Sub
```

Example Code C#

```
private void vcGantt1_VcLinksRightClicking(object sender,
NETRONIC.XGantt.VcLinksClickingEventArgs e)
{
    PopupMenu.Show(vcGantt1, new Point(e.X, e.Y));
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

VcNodeCreated

Event of VcGantt

This event occurs when the interactive creation of a node is completed. The node object, the creation type (here **vcNodeCreated**) and the information whether the created node is the only node or the last node of a node collection (always **True**) are returned, so that a validation can be made.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNodeCreatedEventArgs	Object specific to the event that is being handled

Properties of the VcNodeCreatedEventArgs object

	Data Type	Explanation
Properties:		
⇒ node	VcNode	Node created
⇒ creationType	VcCreationType	Creation type of nodes/links
	Possible Values:	
	.vcDataRecordCreated 6	Data record created by interaction
	.vcDataRecordCreatedByResourceScheduling 5	Data record automatically created by resource scheduling
	.vcNodeCreated 1	Node created via mouse-click
⇒ isLast	System.Boolean	The created node is/is not the only node or the last node of a node collection.

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeCreated(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeCreatedEventArgs) Handles VcGantt1.VcNodeCreated
    MsgBox(e.Node.AllData)
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeCreated(object sender,
NETRONIC.XGantt.VcNodeCreatedEventArgs e)
{
    MessageBox.Show(e.Node.AllData.ToString());
}
```

VcNodeCreating

Event of VcGantt

This event occurs when the user creates a node. The node object is returned, so that a validation can be made.

The data passed by this event can be read, but must not be modified. For modifying them please use the event **VcNodeCreated**.

By setting the return status the create operation can be inhibited.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNodeCreatingEventArgs	Object specific to the event that is being handled

Properties of the VcNodeCreatingEventArgs object

	Data Type	Explanation
Properties:		
⇒ node	VcNode	Node to be created
⇒ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The node will not be created. The node will be created.

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeCreating(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeCreatingEventArgs) Handles VcGantt1.VcNodeCreating
    MsgBox("Show your own dialog")
    e.ReturnStatus = VcReturnStatus.vcRetStatFalse
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeCreating(object sender,
NETRONIC.XGantt.VcNodeCreatingEventArgs e)
{
    MessageBox.Show("Show your own dialog");
    e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcNodeDeleted

Event of VcGantt

This event occurs when the interactive deletion of a node is completed. The node object and the information whether the deleted node was the last one of a batch are returned for data validation.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNodeDeletedEventArgs	Object specific to the event that is being handled

Properties of the VcNodeDeletedEventArgs object

	Data Type	Explanation
Properties:		
⇒ node	VcNode	Node deleted
⇒ isLast	System.Boolean	The deleted node is/is not the last node of a batch.

VcNodeDeleting

Event of VcGantt

This event occurs when the user deletes a node by the context menu. The node object to be deleted is returned, so that you can still check for - whatever - conditions and prohibit the deletion on a negative result by setting the return status.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNodeDeletingEventArgs	Object specific to the event that is being handled

Properties of the VcNodeDeletingEventArgs object

	Data Type	Explanation
Properties:		
⇒ node	VcNode	Node deleted
⇒ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The node will not be deleted. The node will be deleted.

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeDeleting(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeDeletingEventArgs) Handles VcGantt1.VcNodeDeleting
    'deny the deletion of the last node in the chart
    If VcGantt1.NodeCollection.Count = 1 Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
        MsgBox("The last node in the chart cannot be deleted.")
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeDeleting(object sender,
NETRONIC.XGantt.VcNodeDeletingEventArgs e)
{
    //deny the deletion of the last node in the chart
    if (vcGantt1.NodeCollection.Count == 1)
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
    MessageBox.Show("The last node in the chart cannot be deleted.");
}
```

VcNodeLeftClicking

Event of VcGantt

This event occurs when the user clicks the left mouse button on a node (location = vcInDiagram) or on a table entry related to an activity (location = vcInTable). The node object and the cursor position (x,y-coordinates) are captured and passed as parameters.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNodeClickingEventArgs	Object specific to the event that is being handled

Properties of the VcNodeClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ node	VcNode	Node hit
⇒ location	VcLocation	Location in the diagram
	Possible Values: .vcInDiagram 1 .vcInTable 0	Located in the node area Located in the table area
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇌ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles VcGantt1.VcNodeLeftClicking
    'change data field of the node
    e.Node.DataField(4) = 1 - Convert.ToInt64(e.Node.DataField(4))
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeLeftClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    //change data field of the node
    e.Node.set_DataField(4, Convert.ToInt64(e.Node.get_DataField(4)));
}
```


VcNodeLeftDoubleClicking

Event of VcGantt

This event occurs when the user double-clicks the left mouse button on a node (location = vcInDiagram) or on a table entry related to an activity (location = vcInTable). The node object and the cursor position (x,y-coordinates) are captured and passed. If you set the returnStatus to vcRetStatFalse, the integrated **Edit Data** dialog box will be revoked.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNodeClickingEventArgs	Object specific to the event that is being handled

Properties of the VcNodeClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ node	VcNode	Node hit
⇒ location	VcLocation	Location in the diagram
	Possible Values: .vcInDiagram 1 .vcInTable 0	Located in the node area Located in the table area
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The Edit data dialog will not appear. The Edit data dialog will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeLeftDoubleClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles
VcGantt1.VcNodeLeftDoubleClicking
    MsgBox("Show your own dialog")
    e.ReturnStatus = VcReturnStatus.vcRetStatFalse
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeLeftDoubleClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    MessageBox.Show("Show your own dialog");
    e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcNodeModified

Event of VcGantt

This event occurs when the modification of the node specified is completed.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNodeModifiedEventArgs	Object specific to the event that is being handled

Properties of the VcNodeModifiedEventArgs object

	Data Type	Explanation
Properties:		
⇒ node	VcNode	Node modified
⇒ isLast	System.Boolean	The modified node is/is not the only node or the last node of a node collection.

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeModifying(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeModifyingEventArgs) Handles VcGantt1.VcNodeModifying
    'revoke the modification if the node would change the group
    If e.ModificationType And VcModificationTypes.vcChangedGroup Then
        MsgBox("The node cannot be moved to a different group.")
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeModifying(object sender,
NETRONIC.XGantt.VcNodeModifyingEventArgs e)
{
    //revoke the modification if the node would change the group
    if (VcModificationTypes.vcChangedGroup.Equals(true))
    {
        MessageBox.Show("The node cannot be moved into another group.");
    }
}
```

VcNodeModifiedEx

Event of VcGantt

This event occurs when the modification of the marked node was completed.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event

⇒ e | VcNodeModifiedExEventArgs | Object specific to the event that is being handled

Properties of the VcNodeModifiedExEventArgs object

	Data Type	Explanation
Properties:		
⇒ node	VcNode	Node modified
⇒ isLast	System.Boolean	The modified node is/is not the only node or the last node of a node collection.
⇒ modificationType	VcModificationTypes	Modification type
	Possible Values: .vcAnything 1 .vcChangedGroup 16 .vcEndModified 4 .vcHierarchyModified 64 .vcModifiedByResourceScheduling 128 .vcModifiedBySchedule 32 .vcMoved 8 .vcNothing 0 .vcStartModified 2	Modification type cannot be identified. Group of the node was changed (occurs with nodes only). The end date of the node was modified (occurs with nodes only). Hierarchy of the nodes has been changed Modification by resource scheduling (occurs with data records only) Modification by new date calculation Object was moved. No modification The start date of the node was modified (occurs with nodes only).

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeModifiedEx(ByVal sender As System.Object, _
                                     ByVal e As
NETRONIC.XGantt.VcNodeModifiedExEventArgs)
    Handles VcGantt1.VcNodeModifiedEx
    'modify a record in the underlying database of the application
    modifyDataRecord(e.Node.AllData)
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeModifiedEx(object sender,
VcNodeModifiedExEventArgs e)
{
    //modify a record in the underlying database of the application
    modifyDataRecord(e.Node.AllData);
}
```

VcNodeModifying

Event of VcGantt

This event occurs when the user modifies a node. In the course of this, the length or the position of the node or a value in the **Edit Data** dialog may have been changed. The data of the node before and after the modification are passed. By the **modificationType** parameter you get further information of the kind of modification. By setting the return status to **vcRetStatFalse**, the modification can be inhibited.

The data passed by this event can be read, but must not be modified. For modifying them please use the event **VcNodeModified**.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNodeModifyingEventArgs	Object specific to the event that is being handled

Properties of the VcNodeModifyingEventArgs object

	Data Type	Explanation
Properties:		
⇒ oldNode	VcNode	Node before the modification
⇒ node	VcNode	Node to be modified
⇒ modificationType	VcModificationTypes	Type of modification (A combination of the values is also possible.) Possible Values: .vcAnything 1 .vcChangedGroup 16 .vcEndModified 4 .vcHierarchyModified 64 .vcModifiedByResourceScheduling 128 .vcModifiedBySchedule 32 .vcMoved 8 .vcNothing 0 .vcStartModified 2
⇒ returnStatus	VcReturnStatus Possible Values: .vcRetStatFalse 0	Return status The modification will be revoked.

	.vcRetStatOK 1	The modification will be accepted.
--	----------------	------------------------------------

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeModifying(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeModifyingEventArgs) Handles VcGantt1.VcNodeModifying
    ' revoke the modification if the node would change the group
    If e.ModificationType And VcModificationTypes.vcChangedGroup Then
        MsgBox("The node cannot be moved to a different group.")
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeModifying(object sender,
NETRONIC.XGantt.VcNodeModifyingEventArgs e)
{
    //revoke the modification if the node would change the group
    if (e.ModificationType == VcModificationTypes.vcChangedGroup)
    {
        MessageBox.Show("The node cannot be moved into another group.");
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
    }
}
```

VcNodeResizeStarting

Event of VcGantt

This event occurs when the user starts to interactively stretch or shorten a node. It may serve to set smaller modifications to the XGantt, such as making step size depend on nodes (TimeUnitsPerStep).

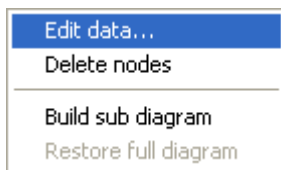
	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	
⇒ e	VcGroupDeletingEventArgs	

	Data Type	Explanation
Properties:		
⇒ node	VcNode	Node resized

VcNodeRightClicking

Event of VcGantt

This event occurs when the user clicks the right mouse button on a node (location = vcInDiagram) or on a table entry related to an activity (location = vcInTable). The node object hit and the cursor position (x,y-coordinates) are returned. By setting the return status you can inhibit the integrated context menu to pop up and replace it by a context menu of your own at the location delivered.



Above: integrated context menu

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNodeClickingEventArgs	Object specific to the event that is being handled

Properties of the VcNodeClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ node	VcNode	Node hit
⇒ location	VcLocation	Placed in the chart
	Possible Values:	
	.vcInDiagram 1	Located in the node area
	.vcInTable 0	Located in the table area
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values:	
	.vcRetStatNoPopup 4	The context menu will be inhibited.
	.vcRetStatOK 1	The context menu will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles VcGantt1.VcNodeRightClicking
    PopupMenu.Show(VcGantt1, New Point(e.X, e.Y))
    e.ReturnStatus = NETRONIC.XGantt.VcReturnStatus.vcRetStatNoPopup
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeRightClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    PopupMenu.Show(vcGantt1, new Point (e.X, e.Y));
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

VcNodesMarked**Event of VcGantt**

This event occurs after the operation of marking or unmarking nodes was finished.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNodesMarkedEventArgs	Object specific to the event that is being handled

Properties of the VcNodesMarkedEventArgs object

	Data Type	Explanation
Properties:		
⇐ (no parameter)		No parameter

Example Code VB.NET

```
Private Sub VcGantt1_VcNodesMarked(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodesMarkedEventArgs) Handles VcGantt1.VcNodesMarked
    MsgBox("Nodes have been marked successfully.")
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodesMarked(object sender,
NETRONIC.XGantt.VcNodesMarkedEventArgs e)
{
    MessageBox.Show("Nodes have been marked successfully.");
}
```

VcNodesMarking

Event of VcGantt

This event occurs when the user selects nodes for marking or when he unmarks marked nodes by a click into the empty diagram. The `NodeCollection` contains the nodes selected by the most recent marking action of the user. If the user unmarks nodes by a click into the empty diagram, the node collection will be empty.

If you set the return status to **vcRetStatFalse**, you have to mark or unmark nodes yourself.

The data passed by this event can be read, but must not be modified. For modifying them please use **VcNodesMarked**.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNodesMarkingEventArgs	Object specific to the event that is being handled

Properties of the VcNodesMarkingEventArgs object

	Data Type	Explanation
Properties:		
⇒ nodeCollection	VcNodeCollection	NodeCollection that contains the nodes selected by the user. If the user has clicked in the diagram, the NodeCollection is empty.
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	Marking has to be done manually. Marking is done automatically.

Example Code VB.NET

```
Private Sub VcGantt1_VcNodesMarking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodesMarkingEventArgs) Handles VcGantt1.VcNodesMarking
    If MsgBox("Mark this node?", MsgBoxStyle.YesNo, "Marking nodes") =
MsgBoxResult.No Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```


Example Code C#

```
private void vcGantt1_VcNodesMarking(object sender,
NETRONIC.XGantt.VcNodesMarkingEventArgs e)
{
    DialogResult retVal = MessageBox.Show("Mark this node?", "Marking nodes",
MessageBoxButtons.YesNo);
    if (retVal == DialogResult.No)
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcNumericScaleLeftClicking**Event of VcGantt**

This event occurs when the user clicks the left mouse button on the numeric scale. The numeric scale object and the cursor position (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNumericScaleClickingEventArgs	Object specific to the event that is being handled

Properties of the VcNumericScaleClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ numericScale	VcNumericScale	Numeric scale hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
↔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcNumericScaleLeftClicking(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcNumericScaleClickingEventArgs) Handles
VcGantt1.VcNumericScaleLeftClicking
    e.NumericScale.BackgroundColor = Color.Blue
End Sub
```

Example Code C#

```
private void vcGantt1_VcNumericScaleLeftClicking(object sender,
NETRONIC.XGantt.VcNumericScaleClickingEventArgs e)
{
    e.NumericScale.BackgroundColor = Color.LightSteelBlue;
}
```

VcNumericScaleLeftDoubleClicking**Event of VcGantt**

This event occurs when the user double-clicks the left mouse button on the numeric scale. The numeric scale object and the mouse position (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNumericScaleClickingEventArgs	Object specific to the event that is being handled

Properties of the VcNumericScaleClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ numericScale	VcNumericScale	Numeric scale hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcNumericScaleLeftDoubleClicking(ByVal sender As Object,
ByVal e As NETRONIC.XGantt.VcNumericScaleClickingEventArgs) Handles
VcGantt1.VcNumericScaleLeftDoubleClicking
    e.NumericScale.MajorTicks = TextBox1.Text
End Sub
```

Example Code C#

```
private void vcGantt1_VcNumericScaleLeftDoubleClicking(object sender,
NETRONIC.XGantt.VcNumericScaleClickingEventArgs e)
{
    e.NumericScale.MajorTicks = textBox1.Text;
}
```

VcNumericScaleRescaling

Event of VcGantt

This event occurs when the user rescales the numeric scale. The NumericScale object and the new BasicUnitWidth are returned, so that you can check whether the scaling is allowed. By setting the return status you can inhibit the modification.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNumericScaleRescalingEventArgs	Object specific to the event that is being handled

Properties of the VcNumericScaleRescalingEventArgs object

	Data Type	Explanation
Properties:		
⇒ numericScale	VcNumericScale	Numeric scale hit
⇒ newBasicUnitWidth	System.Int32	New width of the basic unit
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The numeric scale will not be modified. The numeric scale will be modified.

Example Code VB.NET

```
Private Sub VcGantt1_VcNumericScaleRescaling(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNumericScaleRescalingEventArgs) Handles
VcGantt1.VcNumericScaleRescaling
    Select Case e.NewBasicUnitWidth
        Case Is <= 1000
            MsgBox("New basic unit width:" + e.NewBasicUnitWidth)
        Case Is > 1000
            MsgBox("The maximum basic unit width is 1000.")
            e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End Select
End Sub
```

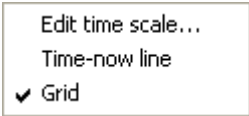
Example Code C#

```
private void vcGantt1_VcNumericScaleRescaling(object sender,
NETRONIC.XGantt.VcNumericScaleRescalingEventArgs e)
{
    switch (e.NewBasicUnitWidth)
    {
        case e.NewBasicUnitWidth.CompareTo(1000):
            MessageBox.Show("New basic unit width: " + e.NewBasicUnitWidth);
            break;
        case e.TimeScale.UnitWidth > 1000:
            MessageBox.Show("The maximum basic unit width is 1000");
            e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
            break;
    }
}
```

VcNumericScaleRightClicking

Event of VcGantt

This event occurs when the user clicks the right mouse button on a numeric scale. The numeric scale object and the mouse position (x,y-coordinates) are returned. By setting the return status you can inhibit the integrated context menu to pop up an replace it by a context menu of your own at the location delivered.



	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcNumericScaleClickingEventArgs	Object specific to the event that is being handled

Properties of the VcNumericScaleClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ numericScale	VcNumericScale	Numeric scale hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatNoPopup 4	The context menu will be inhibited.

.vcRetStatOK 1

The context menu will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcNumericScaleRightClicking(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcNumericScaleClickingEventArgs) Handles
VcGantt1.VcNumericScaleRightClicking
    If MsgBox("Change unit label of numeric scale?", MsgBoxStyle.YesNo) =
MsgBoxResult.Yes Then
        e.NumericScale.UnitLabel = TextBox1.Text
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcNumericScaleRightClicking(object sender,
NETRONIC.XGantt.VcNumericScaleClickingEventArgs e)
{
    DialogResult retVal = MessageBox.Show("Change unit label of numeric scale?",
"Changing numeric scale", MessageBoxButtons.YesNo);
    if (retVal == DialogResult.Yes)
        e.NumericScale.UnitLabel = textBox1.Text;
}
```

VcObjectDrawing

Event of VcGantt

This event is triggered before an object is drawn. It enables you to shape the object by adding your own programming code. You can finally prevent to have the object drawn by the component by setting the return status to **vcRetStatFalse**.

ObjectDraw events are only triggered after the corresponding option was set to its special object type. The option is available for layers and user-defined annotation ribbons.

To draw a layer, you either have to set the property **ObjectDrawEvents-Enabled** of the object **VcLayer** to **True** at run time, or alternatively, at design time, you tick the check box **ObjectDraw Events** for the according layer in the **Specify Bar Appearance** dialog.

To draw a user-defined annotation ribbon, you have to tick the check box **ObjectDraw Events** for the according ribbon in the **Edit Histograms** dialog.

To add something to the layer or to the annotation ribbon drawn by VARCHART XGantt, please use the event **VcObjectDrawn**.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcObjectDrawingEventArgs	Object specific to the event that is being handled

Properties of the VcObjectDrawingEventArgs object

	Data Type	Explanation
Properties:		
Graphics	System.Drawing.Graphics	Device context
ObjectToDraw	Object	Object to be drawn
ObjectType	VcObjectType	Type of object to be drawn
	Possible Values: .vcObjTypeNodeInDiagram 2 .vcObjTypeNodeInLegend 17 .vcObjTypeNumericScale 10 .vcObjTypeSummaryNode 14	object type node in diagram area object type node in legend area object type numeric scale object type summary bar
SubObject	Object	Subobject that is passed context-dependent
SubObjectType	VcObjectType	Type of subobject
	Possible Values: .vcObjTypeLayer 8	object type layer
CompleteRect	VcRect	Rectangle in device coordinates into which the complete object is to be drawn
UpdateRect	VcRect	Rectangle in device coordinates which marks the update area. This area may be the same size as or smaller than the rectangle in completeRect.
ReturnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The object will not be drawn. The object will be drawn.
LineWidth	System.Int32	Width of a thin line. May be used in case of drawing commands in order to adapt the line width to the device context (monitor or printer).
xZoomFactor	System.Int16	This parameter specifies the zoom factor in x-direction, which allows a conversion from distances specified as units of 1/100 mm into pixels, and vice versa. The zoom factor refers to the output device (screen, print preview or printer).
yZoomFactor	System.Int16	This parameter specifies the zoom factor in y-direction, which allows a conversion from distances specified as units of 1/100 mm into pixels, and vice versa. The zoom factor refers to the output device (screen, print preview or printer).

VcObjectDrawn

Event of VcGantt

This events only occurs after an object was drawn. It lets you complete or modify the shape of objects drawn by VARCHART XGantt by programming code of your own.

ObjectDraw events are only triggered after the corresponding option was set to its special object type. The option is available to for layers and user-defined annotation ribbons.

To customize a layer, you either have to set the property **ObjectDrawEventsEnabled** of the object **VcLayer** to **True** at run time, or alternatively, at design time, you tick the check box **ObjectDraw Events** for the according layer in the **Specify Bar Appearance** dialog.

To customize an annotation ribbon, you have to tick the check box **ObjectDraw Events** for the according ribbon in the **Edit Histograms** dialog

If you wish to suppress default drawing of layers or annotation ribbons and to replace it by programming code of your own, please use the event **VcObjectDrawing**.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcObjectDrawnEventArgs	Object specific to the event that is being handled

Properties of the VcObjectDrawnEventArgs object

	Data Type	Explanation
Properties:		
Graphics	System.Drawing.Graphics	Device context
ObjectType	VcObjectType	Type of object drawn
	Possible Values:	
	.vcObjTypeNodeInDiagram 2	object type node in diagram area
	.vcObjTypeNodeInLegend 17	object type node in legend area
	.vcObjTypeNumericScale 10	object type numeric scale
	.vcObjTypeSummaryNode 14	object type summary bar
SubObject	Object	Subobject that was passed context-dependent
SubObjectType	VcObjectType	Type of subobject
	Possible Values:	

	.vcObjTypeLayer 8	object type layer
CompleteRect	VcRect	Rectangle in device coordinates into which the complete object was drawn
UpdateRect	VcRect	Rectangle in device coordinates which marks the update area. This area may be the same size as or smaller than the rectangle in completeRect.
LineWidth	System.Int32	Width of a thin line. May be used in case of drawing commands in order to adapt the line width to the device context (monitor or printer).
xZoomFactor	System.Int16	This parameter specifies the zoom factor in x-direction, which allows a conversion from distances specified as units of 1/100 mm into pixels, and vice versa. The zoom factor refers to the output device (screen, print preview or printer).
yZoomFactor	System.Int16	This parameter specifies the zoom factor in y-direction, which allows a conversion from distances specified as units of 1/100 mm into pixels, and vice versa. The zoom factor refers to the output device (screen, print preview or printer).

VcResourceSchedulingProgressing

Event of VcGantt

During the resource scheduling process, this event informs on the progress of the scheduling procedure. The number of jobs scheduled and the total number of jobs are reported. By setting the return status to **vcRetStatFalse**, the scheduling procedure will be abandoned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcResourceSchedulingProgressingEventArgs	Object specific to the event that is being handled

Properties of the VcResourceSchedulingProgressingEventArgs object

	Data Type	Explanation
Properties:		
⇒ ScheduledJobCount	System.Int32	Number of scheduled jobs
⇒ TotalJobCount	System.Int32	Total number of jobs

↔ ReturnStatus	System.Object	Return status vcRetStatFalse : scheduling is abandoned vcRetStatDefault : scheduling is continued
----------------	---------------	---

VcResourceSchedulingWarning

Event of VcGantt

This event is triggered if the resource scheduling procedure finds inconsistencies in the data records (see method **process** in the object VcResourceScheduler2). This event detects certain errors in the data definition. You can cancel the scheduling procedure by setting the return status.

	Data Type	Explanation
Properties: ↔ WarningType	VcResSchedWarningType	Warning type
	Possible Values: .vcResSched-AssignmentLoadPerItemIsZero 23	In the assignment data set specified the content of the data field LoadOrConsumptionPerItem is evaluated to be 0. This leads to the assignment being ignored during scheduling.
	.vcResSched-AssignmentNoOperationID 3	In the assignment data record also passed the data field of the ID of the operations data record is empty. Because of this, the assignment will be ignored in the ongoing procedure.
	.vcResSched-AssignmentNoResourceID 1	In the assignment data record also passed the data field of the ID of the resources data record is empty. Because of this, the assignment will be ignored in the ongoing procedure.
	.vcResSched-AssignmentNoDataRecords 0	No assignment data records exist; the parameter DataRecord is null .
	.vcResSched-AssignmentNoResourceID 2	In the assignment data record also passed the resource data record corresponding to the resource data record ID was not found. Because of this, the assignment will be ignored in the ongoing procedure.
	.vcResSched-AssignmentOperationNotFound 4	In the assignment data record also passed the operations data record corresponding to the operations data record ID was not found. Because of this, the assignment will be ignored in the ongoing procedure.
	.vcResSched-AssignmentTimingResourceMultiple 5	The assignment data record also passed represents a prohibited second or other assignment of an operation to a resource of the type vcResSched-Timing . Because of this, the assignment will be ignored in the ongoing procedure.
	.vcResSchedOperationLoadPerItemIsZero 24	In the operation data set specified the content of the data field LoadPerItem is evaluated to be 0. This leads to the operation being ignored during scheduling.

.vcResSchedOperation-NoTaskID 6	In the operations data record also passed the data field of the ID of the task data record is empty. Because of this, the operation will be ignored in the ongoing procedure.
.vcResSchedOperation-OverlapQuantityOutOf-Range 19	This warning occurs if the overlap quantity of an operation exceeds the quantity of the associated task. This warning will cause the task to be excepted from scheduling.
.vcResSchedOperation-StartLockDateOutOf-Range 15	This warning occurs if the start lock date of an operation is not between the release date and the due date of the task. This warning will cause the task to be excepted from scheduling.
.vcResSchedOperation-TaskNotFound 7	In the operations data record also passed the task data record corresponding to the task data record ID was not found. Because of this, the operation will be ignored in the ongoing procedure.
.vcResSchedOperation-WorkInProgressOutOf-Range 20	This warning occurs if the quantity completed of the operation data set passed exceeds the quantity of the associated task. This warning will cause the task to be excepted from scheduling.
.vcResSchedResource-CalendarNotFound 22	This warning occurs if the calendar object of the name stored in the data field denoted by the property ResourceCalendarNameFieldIndex does not exist.
.vcResSchedResource-GroupResourceNot-Found 10	In the resources data record also passed the resource data record corresponding to the group resource data record ID was not found. Because of this, the resource cannot be allocated to a group.
.vcResSchedResource-HistogramNotFound 21	This warning occurs if the histogram of a name equal to the resource does not exist.
.vcResSchedResource-InputCurveNot-Found 11	The input curve of the resource data record also passed was not found. Input curves for resources of the type vcResSchedTiming and vcResSched-Work are capacity curves; for resources of the resource type vcResSchedMaterial they are supply curves.
.vcResSchedResource-InputCurves-CompletelyZero 12	The values of the input curve of the resource data record also passed are all zero. Input curves for resources of the type vcResSchedTiming and vcResSchedWork are capacity curves; for resources of the resource type vcResSchedMaterial they are supply curves.
.vcResSchedResource-OutputCurveNot-Found 13	The output curve of the resource data record also passed was not found. Output curves for resources of the type vcResSchedTiming and vcResSched-Work are workload curves; for resources of the resource type vcResSchedMaterial they are stock curves.
.vcResSchedResource-OutputCurveOfFalse-Type 14	The output curve of the resource data record also passed cannot be used, since it is not of the type vcSetCurve (please see method CurveType of the object VcCurve). Output curves for resources of the type vcResSchedTiming and vcResSchedWork are workload curves; for resources of the resource type vcResSchedMaterial they are stock curves.
.vcResSchedTask-CapacityBeyond-Limit 25	This warning occurs if there is at least one operation in the task whose capacity demand is above an internal limit. The capacity demand results from the task quantity in the task, the LoadPerItem in the operation and, if necessary, an efficiency factor in the resource to be allocated. The current limit is 100000.
.vcResSchedTaskDue-DateEarlierThan-ReleaseDate 9	In the task data record also passed the release date is earlier than the due date. Because of this, the task will be ignored in the ongoing procedure.

	.vcResSchedTaskDue- DateEqualToRelease- Date 18	This warning occurs if the release date of a task equals the due date. This warning will cause the task to be excepted from scheduling.
	.vcResSchedTaskDue- DateOutOfRange 17	This warning occurs if the due date of a task is not between the PlanningStartDate and the Planning- EndDate or between the dates in the visible section (default). If also the release date is outside the time span allowed, the task will be excepted from scheduling.
	.vcResSchedTask- QuantityIsZero 8	In the task data record also passed the task quantity is zero. Because of this, the task will be ignored in the ongoing procedure.
	.vcResSchedTask- ReleaseDateOutOf- Range 16	This warning occurs if the release date of a task is not between the PlanningStartDate and the PlanningEndDate or between the dates set by the default. If also the due date is outside the time span allowed, the task will be excepted from scheduling.
↔ DataRecord	VcDataRecord	Data record, to which the warning refers
↔ ReturnStatus	VcReturnStatus	Return status vcRetStatFalse: scheduling is abandoned vcRetStatDefault: scheduling is continued
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	Resource scheduling will be cancelled. Resource scheduling will be continued.

VcSashButtonClicked

Event of VcGantt

The usage of this event requires a special setting in the .ini file. Please contact NETRONIC if necessary.

This event occurs when the user makes a left click on one of the buttons having been positionend on the sash before. The Sash object and the index of the clicked button (0 or 1) are returned.

Tip: The bitmaps/WMFs arrow left/right/up/down can easily be positioned at the sash by the method **VcGantt.SetImageResource**, with the following settings for the paramater **imageName**:

****SashHorizontalFirstButton:** arrow left

****SashHorizontalSecondButton:** arrow right

****SashVerticalFirstButton:** arrow up

****SashVerticalSecondButton:** arrow down

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcSashButtonClickedEventArgs	Object specific to the event that is being handled

Properties of the VcSashButtonClickedEventArgs object

	Data Type	Explanation
Properties:		
⇒ SashType	VcSashType	Type of hit button
	Possible Values:	
	.vcDiagramHistogramSash 3	Horizontal sash between diagram and histogram
	.vcLeftTableHistogramSash 1	Vertical sash between left table and diagram
	.vcRightTableHistogramSash 2	Vertical sash between right table and diagram, only available after having specified a second table to be displayed in the ini file

VcStatusLineTextShowing

Event of VcGantt

This event occurs when a message of general interest is displayed in the status line, e.g. a functional note during loading, or some information on the node to which the cursor is pointing.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcStatusLineTextShowingEventArgs	Object specific to the event that is being handled

Properties of the VcStatusLineTextShowingEventArgs object

	Data Type	Explanation
Properties:		
⇒ text	System.String	Information text

Example Code VB.NET

```
Private Sub VcGantt1_VcStatusLineTextShowing(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcStatusLineTextShowingEventArgs) Handles
VcGantt1.VcStatusLineTextShowing
    TextBox1.Text = e.Text
End Sub
```

Example Code C#

```
private void vcGantt1_VcStatusLineTextShowing(object sender,
NETRONIC.XGantt.VcStatusLineTextShowingEventArgs e)
{
    textBox1.Text = e.Text;
}
```

VcTableCaptionLeftClicking**Event of VcGantt**

This event occurs when the user clicks the left mouse button on a table caption. The table object, the column number and the cursor position (x,y-coordinates) are returned. If the diagram is not grouped or hierarchically sorted, the activities will be sorted according to the table column hit.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcTableCaptionClickingEventArgs	Object specific to the event that is being handled

Properties of the VcTableCaptionClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ table	VcTable	Table hit
⇒ columnNumber	System.Int32	Index of the table column hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇒ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcTableCaptionLeftClicking(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcTableClickingEventArgs) Handles
VcGantt1.VcTableCaptionLeftClicking
VcGantt1.LeftTable.TableFormatCollection.FirstFormat.FormatField(0).BackColor =
Color.Blue
End Sub
```

Example Code C#

```
private void vcGantt1_VcTableCaptionLeftClicking(object sender,
NETRONIC.XGantt.VcTableClickingEventArgs e)
{
vcGantt1.LeftTable.TableFormatCollection.FirstFormat().get_FormatField(0).BackCo
lor = Color.LightSteelBlue;
}
```

VcTableCaptionLeftDoubleClicking**Event of VcGantt**

This event occurs when the user double-clicks the left mouse button on a table heading. The table object, the column number and the cursor position (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcTableCaptionClickingEventArgs	Object specific to the event that is being handled

Properties of the VcTableCaptionClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ table	VcTable	Table hit
⇒ columnNumber	System.Int32	Index of the column hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇌ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcTableCaptionLeftDoubleClicking(ByVal sender As Object,
ByVal e As NETRONIC.XGantt.VcTableClickingEventArgs) Handles
VcGantt1.VcTableCaptionLeftDoubleClicking
    VcGantt1.LeftTable.Visible = True
End Sub
```

Example Code C#

```
private void vcGantt1_VcTableCaptionLeftDoubleClicking(object sender,
NETRONIC.XGantt.VcTableClickingEventArgs e)
{
    vcGantt1.LeftTable.Visible = true;
}
```

VcTableCaptionRightClicking**Event of VcGantt**

This event occurs when the user clicks the right mouse button on a table heading. The table object, the column number and the cursor position (x,y-coordinates) are returned. By setting the return status you can inhibit the integrated context menu to pop up and replace it by a context menu of your own at the location delivered.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcTableCaptionClickingEventArgs	Object specific to the event that is being handled

Properties of the VcTableCaptionClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ table	VcTable	Table hit
⇒ columnNumber	System.Int32	Index of the hit table
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatNoPopup 4 .vcRetStatOK 1	The context menu will be inhibited. The context menu will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcTableCaptionRightClicking(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcTableClickingEventArgs) Handles
VcGantt1.VcTableCaptionRightClicking
    PopupMenu.Show(VcGantt1, New Point(e.X, e.Y))
    e.ReturnStatus = NETRONIC.XGantt.VcReturnStatus.vcRetStatNoPopup
End Sub
```

Example Code C#

```
private void vcGantt1_VcTableCaptionRightClicking(object sender,
NETRONIC.XGantt.VcTableClickingEventArgs e)
{
    PopupMenu.Show(vcGantt1, new Point (e.X, e.Y));
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

VcTableColumnWidthChanged**Event of VcGantt**

This event occurs when the user has modified the width of a table column interactively. The table, the index and the current width (as 1/100 mm) of the modified column are returned. By setting the return status, you can inhibit the modification.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	
⇒ e	VcTableColumnWidthChangedEventArgs	

	Data Type	Explanation
Properties:		
⇒ table	VcTable	Table
⇒ columnNumber	System.Int16	Index of the column modified
⇒ currentWidth	System.Int32	New column width
⇌ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The width of the table column will not be modified. The width of the table column will be modified.

VcTableColumnWidthChanging

Event of VcGantt

This event occurs when the user modifies the width of a table column. The table, the index and the current width (as 1/100 mm) of the modified column are returned. By setting the return status, you can inhibit the modification.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcTableColumnWidthChangingEventArgs	Object specific to the event that is being handled

Properties of the VcTableColumnWidthChangingEventArgs object

	Data Type	Explanation
Properties:		
⇒ table	VcTable	Table
⇒ columnNumber	System.Int16	Index of the column modified
⇒ currentWidth	System.Int32	New column width
⇌ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The width of the table column will not be modified. The width of the table column will be modified.

Example Code VB.NET

```
Private Sub VcGantt1_VcTableColumnWidthChanging(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcTableColumnWidthChangingEventArgs) Handles
VcGantt1.VcTableColumnWidthChanging
    If e.CurrentWidth > 5000 Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
        VcGantt1.LeftTable.ColumnWidth(index) = 5000
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcTableColumnWidthChanging(object sender,
NETRONIC.XGantt.VcTableColumnWidthChangingEventArgs e)
{
    if (e.CurrentWidth > 5000)
    {
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
        vcGantt1.LeftTable.set_ColumnWidth(1, 5000);
    }
}
```

VcTableColumnWidthOptimizing

Event of VcGantt

This event occurs after a double-click on the separation line between two table columns, provided that on the **General** property page the **Allow table column width optimization** check box is activated or the property **TableColumnWidthOptimizationAllowed** was set. Then the width of the column on the left will be adapted automatically to the length of the text which it contains. The table and the index of the modified column are returned. By setting the return status, you can inhibit the modification.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcTableColumnWidthOptimizingEventArgs	Object specific to the event that is being handled

Properties of the VcTableColumnWidthOptimizingEventArgs object

	Data Type	Explanation
Properties:		
⇒ table	VcTable	Table
⇒ index	System.Int16	Index of the column modified
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The width of the table column will not be optimized. The width of the table column will be optimized.

VcTableWidthChanging

Event of VcGantt

This event occurs when the user modifies the width of the table. The table and the modified table/diagram aspect ratio are returned. By setting the return status you can inhibit the modification.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event

⇒ e	VcTableWidthChangingEventArgs	Object specific to the event that is being handled
-----	-------------------------------	--

Properties of the VcTableWidthChangingEventArgs object

	Data Type	Explanation
Properties:		
⇒ table	VcTable	Table
⇒ tableWidthRatio	System.Int32	Ratio of the table width to the Width of the the total diagram (including table)
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The width of the table will not be modified. The width of the table will be modified.

Example Code VB.NET

```
Private Sub VcGantt1_VcTableWidthChanging(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcTableWidthChangingEventArgs) Handles
VcGantt1.VcTableWidthChanging
    If e.TableDiagramWidthRatio > 30 Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
        VcGantt1.LeftTableDiagramWidthRatio = 30
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcTableWidthChanging(object sender,
NETRONIC.XGantt.VcTableWidthChangingEventArgs e)
{
    if (e.TableDiagramWidthRatio > 30)
    {
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
        vcGantt1.LeftTableDiagramWidthRatio = 30;
    }
}
```

VcTableWidthChangingEx

Event of VcGantt

This event occurs when the user modifies the width of the table. The table and the modified table/diagram aspect ratio are returned. By setting the return status you can inhibit the modification.

In contrast to the **VcTableWidthChanging** event this event returns the parameter *tableWidthRatio* as "Double" value, thus achieving a higher level of accuracy. The usage of this event has to be enabled by the **UseHigher-**

TableDiagramWidthRatioPrecision property or by activating the corresponding option on the **General** property page.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcTableWidthChangingExEventArgs	Object specific to the event that is being handled

Properties of the VcTableWidthChangingExEventArgs object

	Data Type	Explanation
Properties:		
⇒ table	VcTable	Table
⇒ tableWidthRatio	System.Double	Ratio of the table width to the width of the total diagram (including table)
↔ returnStatus	VcReturnStatus	Return status

VcTextEntrySupplying

Event of VcGantt

This event only occurs when the VcGantt property **TextEntrySupplying-EventEnabled** is set to **True**. It occurs when a text is displayed. You can use this event for editing the texts of context menus, dialog boxes, info boxes, error messages and the names of days and months.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcTextEntrySupplyingEventArgs	Object specific to the event that is being handled

Properties of the VcTextEntrySupplyingEventArgs object

	Data Type	Explanation
Properties:		
⇒ controlIndex	VcTextEntryIndex	Text constant the contents of which is to be replaced

Possible Values:

.vcTXECtxmenArrowMode 2116
 .vcTXECtxmenBarGroupSepLine 2111

 .vcTXECtxmenCancelGrouping 2108

 .vcTXECtxmenCreateBoxMode 2135
 .vcTXECtxmenCreateLinkMode 2118
 .vcTXECtxmenCreateNodeMode 2117

 .vcTXECtxmenDateLineGrid 2106
 .vcTXECtxmenDeleteCurvePoint 2131

 .vcTXECtxmenDeleteLink 2102
 .vcTXECtxmenDeleteNode 2101
 .vcTXECtxmenEditGroup 2160

 .vcTXECtxmenEditLink 2154
 .vcTXECtxmenEditNode 2100
 .vcTXECtxmenFilePrint 2122
 .vcTXECtxmenFilePrintPreview 2121
 .vcTXECtxmenFilePrintSetup 2120
 .vcTXECtxmenFullDiagram 2156

 .vcTXECtxmenGraphicExport 2123
 .vcTXECtxmenGroupCollapse 2114
 .vcTXECtxmenGroupCollapseRowsBelow 2129

 .vcTXECtxmenGroupDelete 2115
 .vcTXECtxmenGrouped 2107

 .vcTXECtxmenGroupExpand 2113
 .vcTXECtxmenGroupExpandRowsBelow 2128

 .vcTXECtxmenGroupNodesBelow 2126
 .vcTXECtxmenGroupNodesInOneRow 2127
 .vcTXECtxmenGroupNodesOptimized 2124
 .vcTXECtxmenGroupNodesOverlaid 2125

 .vcTXECtxmenGroupOutlineIndent 2134
 .vcTXECtxmenGroupOutlineOutdent 2133

 .vcTXECtxmenGroupSortingOptions 2110

 .vcTXECtxmenInsertCurvePointMode 2130
 .vcTXECtxmenInvertSelection 2103

 .vcTXECtxmenPageLayout 2119
 .vcTXECtxmenReOptimizeNodesInGroup 2136

 .vcTXECtxmenShowLegendView 2157

 .vcTXECtxmenShowWorldView 2157
 .vcTXECtxmenSubDiagram 2155

 .vcTXECtxmenTimeScaleEditor 2104
 .vcTXECtxmenToggleDateLine 2105
 .vcTXECtxmenUnmarkAllCurves 2136

 .vcTXEDlgLegArrangement 2046

Text in context menu: **Pointer mode**
 Constant not longer in use but still visible in the API
 Constant not longer in use but still visible in the API
 Text in context menu: **Mode: Create box**
 Text in context menu: **Mode: Create link**
 Text in context menu: **Mode:Create node**
 Text in context menu: **Grid**
 Text in context menu: **Delete curve point**
 Text in context menu: **Delete link**
 Text in context menu: **Delete nodes**
 Text in context menu of the **group: Edit group data**
 Text in context menu: **Edit Link**
 Text in context menu: : **Edit data**
 Text in context menu: **Print**
 Text in context menu: **Print preview**
 Text in context menu: **Print setup**
 Text in context menu: **Restore full diagram**
 Text in context menu: **Export graphics**
 Text in context menu: **Collapse group**
 Text in context menu: **Collapse Rows Below**
 Text in context menu: **Delete group**
 Constant not longer in use but still visible in the API
 Text in context menu: **Expand Group**
 Text in context menu: **Expand Rows Below**
 Constant not longer in use but still visible in the API
 Text in **group** context menu: **All Nodes In One Row**
 Text in **group** context menu: **Arrange Nodes Optimized**
 Text in **group** context menu: **Arrange Nodes Overlaid**
 Text in the context menu: **Outline indent**
 Text in the context menu: **Outline outdent**
 Text in context menu: **Sorting options for groups**
 Text in context menu: **Insert curve point**
 Constant not longer in use but still visible in the API
 Text in context menu: **Page setup**
 Text in context menu: **Re-optimize nodes**
 Text in context menu: **Show legend view**
 Text in context menu: **Show world view**
 Text in context menu: **Build sub diagram**
 Text in context menu: **Edit time scale**
 Text in context menu: **Time-now line**
 Text in context menu of the **histogram: Unmark all curves**
 Text in the **Legend Attributes** dialog: **Arrangement**

.vcTXEDlgLegBottomMargin 2052	Text in the Legend Attributes dialog: Bottom margin
.vcTXEDlgLegFixedToColumns 2048	Text in the Legend Attributes dialog: Fixed to columns
.vcTXEDlgLegFixedToRows 2047	Text in the Legend Attributes dialog: Fixed to rows
.vcTXEDlgLegFixedToRowsAndColumns 2049	Text in the Legend Attributes dialog: Fixed to rows and columns
.vcTXEDlgLegIdcancel 2042	Legend Attributes dialog: Cancel button
.vcTXEDlgLegIdd 2040	Dialog Legend Attributes : Text in Title Bar
.vcTXEDlgLegIdok 2041	Button text in Legend Attributes dialog: OK
.vcTXEDlgLegLegendElements 2045	Text in the Legend Attributes dialog: Legendelements
.vcTXEDlgLegLegendFont 2053	Legend Attributes dialog: legend Font... button
.vcTXEDlgLegLegendTitleFont 2044	Legend Attributes dialog: legend title Font button...
.vcTXEDlgLegLegendTitleVisible 2043	Text in the Legend Attributes dialog: Legend title visible
.vcTXEDlgLegMargins 2050	Text in the Legend Attributes dialog: Margins
.vcTXEDlgLegTopMargin 2051	Text in the Legend Attributes dialog: Top margin :
.vcTXEDlgNedCaptionPrefix 2024	Edit data dialog, text for text line: "Node"
.vcTXEDlgNedIdapply 2027	Edit data dialog, Apply button
.vcTXEDlgNedIdcancel 2016	Text in the Edit data dialog: Cancel
.vcTXEDlgNedIdclose 2029	Edit data dialog: Close button
.vcTXEDlgNedIddd 2014	caption of the Edit data dialog
.vcTXEDlgNedIdhelp 2028	Edit data dialog: Help button
.vcTXEDlgNedIdok 2015	Text in the Edit data dialog: OK
.vcTXEDlgNedNamesColStr 2018	Text in the Edit data dialog: Fields
.vcTXEDlgNedTTGotoFirst 2032	Edit data dialog: tooltip text Show first selected activity
.vcTXEDlgNedTTGotoLast 2035	Edit data dialog, Tooltip " Show last selected activity "
.vcTXEDlgNedTTGotoPrev 2033	Edit data dialog: tooltip text Show previous selected activity
.vcTXEDlgNedValuesColStr 2019	Text in the Edit data dialog: Values
.vcTXEDlgTscEndDate 2012	Text in Edit time scale dialog: End Date
.vcTXEDlgTscIdcancel 2010	Edit time scale dialog: button text Cancel
.vcTXEDlgTscIddd 2008	Edit time scale dialog: text in title bar
.vcTXEDlgTscIdok 2009	Edit time scale dialog: button text OK
.vcTXEDlgTscScale 2013	Text in Edit time scale dialog: Scale
.vcTXEDlgTscStartDate 2011	Text in time scale editor dialog: Start Date
.vcTXEErrTxtCannotMoveToEmptyRow 2735	Message text: "Cannot insert node in not existing group."
.vcTXEErrTxtEndNotEarlierThanNextSect 2734	Message text: "End date ""%s"" is not earlier than end date of next section.\n\nThe old date will be inserted again."
.vcTXEErrTxtEndNotLaterThanStart 2732	Message text: "End date ""%s"" is not later than start date.\n\nThe old date will be inserted again."
.vcTXEErrTxtEntryTooLong 2730	Message text: "Entry is too long, %s characters are possible."
.vcTXEErrTxtSpinNoButton 2727	Constant not longer in use but still visible in the API
.vcTXEErrTxtSpinNumberFormatFloat 2724	Constant not longer in use but still visible in the API
.vcTXEErrTxtSpinNumberFormatInt 2723	Constant not longer in use but still visible in the API

.vcTXEErrTxtSpinNumberMissing 2722	Constant not longer in use but still visible in the API
.vcTXEErrTxtSpinNumberTooHigh 2725	Constant not longer in use but still visible in the API
.vcTXEErrTxtSpinNumberTooLow 2726	Constant not longer in use but still visible in the API
.vcTXEErrTxtSpinUnitInsert 2720	Constant not longer in use but still visible in the API
.vcTXEErrTxtSpinUnitNotInsert 2721	Constant not longer in use but still visible in the API
.vcTXEErrTxtSpinWrongFormatString 2728	Constant not longer in use but still visible in the API
.vcTXEErrTxtSpinWrongUnitInserted 2718	Constant not longer in use but still visible in the API
.vcTXEErrTxtSpinWrongUnitNotInserted 2719	Constant not longer in use but still visible in the API
.vcTXEErrTxtStartNotEarlierThanEnd 2731	Message text: "Start date ""%s"" is not earlier than end date.\n\nThe old date will be inserted again."
.vcTXEErrTxtStartNotLaterThanPrevSect 2733	Message text: "Start date ""%s"" is not later than start date of previous section.\n\nThe old date will be inserted again."
.vcTXEErrTxtWrongLongInteger 2729	Message text: "Entry is not an integer or too big."
.vcTXEInfWndChangeEndDate 2615	Tooltip text: Change End Date
.vcTXEInfWndChangeSectionStartDate 2618	Tooltip text: Modify section start date
.vcTXEInfWndChangeStartDate 2614	Tooltip text: Change Start Date.
.vcTXEInfWndCopyActivity 2619	Tooltip text: Copy Node
.vcTXEInfWndCreateActivity 2611	Tooltip text: Create Node
.vcTXEInfWndDate 2620	Tooltip text: Date
.vcTXEInfWndDateValue 12620	Tooltip text: Date
.vcTXEInfWndDayPI 2604	Tooltip text: days
.vcTXEInfWndDaySi 2603	Tooltip text: day
.vcTXEInfWndDuration 2602	Tooltip text: Duration
.vcTXEInfWndDurationValue 12602	Tooltip text: Duration
.vcTXEInfWndEnd 2601	Tooltip text: End
.vcTXEInfWndEndValue 12601	Tooltip text: End date
.vcTXEInfWndHourPI 2606	Tooltip text: hours
.vcTXEInfWndHourSi 2605	Tooltip text: hour
.vcTXEInfWndMinPI 2608	Tooltip text: minutes
.vcTXEInfWndMinSi 2607	Tooltip text: minute
.vcTXEInfWndMoveActivity 2612	Tooltip text: Move Node
.vcTXEInfWndMoveDateLine 2622	Tooltip text: Move date line
.vcTXEInfWndMoveLayer 2613	Tooltip text: Move layer
.vcTXEInfWndResizeBUW 2616	Tooltip text: Resize section width
.vcTXEInfWndResizeNumericBUW 2617	Tooltip text: Modify numeric scale's width
.vcTXEInfWndSecPI 2610	Tooltip text: seconds
.vcTXEInfWndSecSi 2609	Tooltip text: second
.vcTXEInfWndStart 12600	Tooltip text: Start date of date line
.vcTXEInfWndStart 2600	Tooltip text: Start
.vcTXEPrctBtAll 2306	Button text in Print Preview dialog: Overview
.vcTXEPrctBtApply 2318	Button text in Page Setup dialog: Apply
.vcTXEPrctBtCancel 2302	Button text in Print Busy box: Cancel
.vcTXEPrctBtClose 2303	Button text in Print Preview dialog: Close
.vcTXEPrctBtFitToPage 2308	Button text in Print Preview dialog: Fit To Page
.vcTXEPrctBtNext 2305	Button text in Print Preview dialog: Next
.vcTXEPrctBtOk 2301	Button text in Page Setup dialog: OK
.vcTXEPrctBtPageLayout 2311	Button text in Print Preview dialog: Page Setup
.vcTXEPrctBtPrevious 2304	Button text in Print Preview dialog: Previous

.vcTXEPrcTbtPrint 2313	Button text in Print Preview dialog: Print
.vcTXEPrcTbtPrinterSetup 2312	Button text in Print Preview dialog: Printer setup
.vcTXEPrcTbtSingle 2307	Button text in Print Preview dialog: Single
.vcTXEPrcTbtZoomPrint 2319	Button text in Print Preview dialog: Print Area...
.vcTXEPrcDtAddCuttingMarks 2514	Text in the Page Setup dialog: Show crop marks
.vcTXEPrcDtAdjustTimescale 2560	Page Layout Text: Adjust time scale to width of pages
.vcTXEPrcDtAdoptTableWidthOfView 2591	Text in Page Setup dialog: Adopt appearance from view on screen
.vcTXEPrcDtAlignment 2526	Text in the Page Setup dialog: Alignment
.vcTXEPrcDtAlignmentItems 2583	Text in the Page Setup dialog: Top left Top Top right Left Centered Right Bottom left Bottom Bottom right
.vcTXEPrcDtBottom 2521	Text in the Page Setup dialog: Bottom
.vcTXEPrcDtCm 2530	Text in the Page Setup dialog: cm
.vcTXEPrcDtCombinedFitToPage 2574	Text in the Page Setup dialog: Zoom with horizontal fitting
.vcTXEPrcDtCurrentValues 2581	Text in the Page Setup dialog: Current
.vcTXEPrcDtEnableDiagram 2559	Text in Page Setup dialog: Show diagram
.vcTXEPrcDtEnableTable 2558	Text in Page Setup dialog: Show Table
.vcTXEPrcDtFitToPage 2508	Text in the Page Setup dialog: Fit to page counts
.vcTXEPrcDtFoldingMarksItems 2577	Text in the Page Setup dialog: Form A Form B Form C
.vcTXEPrcDtFoldingMarksText 2576	Text in the Page Setup dialog: Show &folding marks (DIN 824):
.vcTXEPrcDtFooterGroup 2584	Text in the Page Setup dialog: Footer line
.vcTXEPrcDtFrameOutside 2515	Text in the Page Setup dialog: Show frame outside
.vcTXEPrcDtInch 2588	Text in the Page Setup dialog: in
.vcTXEPrcDtLeft 2520	Text in the Page Setup dialog: Left
.vcTXEPrcDtMargins 2529	Text in the Page Setup dialog: Minimum sizes for sheet margins
.vcTXEPrcDtMaxPages 2580	Text in the Page Setup dialog: pages
.vcTXEPrcDtOff 2557	Text Off dialog
.vcTXEPrcDtOptions 2528	Text in the Page Setup dialog: Options
.vcTXEPrcDtPageDescription 2562	Text in Page Setup dialog: Text
.vcTXEPrcDtPageLayout 2532	Page Setup dialog: Text in Title Bar
.vcTXEPrcDtPageNumberingItems 2582	Text in the Page Setup dialog: Row.Column Column.Row Page/Count
.vcTXEPrcDtPageNumbers 2518	Text in the Page Setup dialog: Page numbering
.vcTXEPrcDtPagePadding 2585	Text in the Page Setup dialog: &Pad pages with space
.vcTXEPrcDtPagePreview 2533	Print Preview dialog: Text in Title Bar
.vcTXEPrcDtPagesMaxHeight 2511	Text in the Page Setup dialog: Maximum height
.vcTXEPrcDtPagesMaxWidth 2510	Text in the Page Setup dialog: Maximum width
.vcTXEPrcDtPercent 2509	Text in the Page Setup dialog: %
.vcTXEPrcDtPrintDate 2564	Text in Page Setup dialog: Additionally print current &date
.vcTXEPrcDtPrintingPage 2556	Text in Print Busy Box: Printing page %1 of %2 on
.vcTXEPrcDtReduceExpand 2507	Text in the Page Setup dialog: Zoom factor

.vcTXEPrctDtRepeatTable 2565	Text in the Page Setup dialog: Repeat title/table/time scale/legend
.vcTXEPrctDtRight 2522	Text in the Page Setup dialog: Right
.vcTXEPrctDtScaling 2527	Text in the Page Setup dialog: Scaling
.vcTXEPrctDtScalingMode 2578	Text in the Page Setup dialog: &Mode:
.vcTXEPrctDtStatusBarCurrentValues 2586	Text in the Status bar of the Page Setup dialog: Page %1 selected (in row %2, column %3)
.vcTXEPrctDtStatusBarSelectedPage 2587	Text in the Status bar of the Page Setup dialog: Page %1 selected (in row %2, column %3)
.vcTXEPrctDtTableColumnRange 2575	Text in the Page Layout dialog: Show table columns (e.g. 1-5;7)
.vcTXEPrctDtTimeColumnEnd 2590	Text in Page Setup dialog: Time scale end:
.vcTXEPrctDtTimeColumnStart 2589	Text in Page Setup dialog: Time scale start:
.vcTXEPrctDtTop 2519	Text in the Page Setup dialog: Top
.vcTXEPrctDtZoomFactor 2579	Text in the Page Setup dialog: &Zoom factor:
.vcTXEPrctMtAdjustBottomAndTopMargin 2437	Message text: The bottom margin is out of range and therefore will be reduced to %1 cm.\r\nIn addition, the top margin will be adjusted to %2 cm.
.vcTXEPrctMtAdjustLeftAndRightMargin 2434	Message text: The left margin is out of range and therefore will be reduced to %1 cm.\r\nIn addition, the right margin will be reduced to %2 cm.
.vcTXEPrctMtAdjustRightAndLeftMargin 2435	Message text: The right margin is out of range and therefore will be reduced to %1 cm.\r\nIn addition, the left margin will be adjusted to %2 cm.
.vcTXEPrctMtAdjustTopAndBottomMargin 2436	Message text: The top margin is out of range and therefore will be reduced to %1 cm.\r\nIn addition, the bottom margin will be reduced to %2 cm.
.vcTXEPrctMtBottomMargin 2409	Message text: Bottom margin...
.vcTXEPrctMtIncompatibleVcVersion 2414	Message text: VcVersion incompatible
.vcTXEPrctMtLeftMargin 2406	Message text: Left margin is out of range and therefore will be reduced to %s cm.
.vcTXEPrctMtPrinterNotInstalled 2411	Message text: Printer not installed
.vcTXEPrctMtPrintingNotPossible 2402	Message text: Printing not possible at time
.vcTXEPrctMtRightMargin 2408	Message text: Right margin is out of range and therefore will be reduced to %s cm.
.vcTXEPrctMtSelectPaperSize 2413	Message text: Selected paper size too small
.vcTXEPrctMtTopMargin 2407	Message text: Top margin is out of range and therefore will be reduced to %s cm.
.vcTXEPrctMtValueOutOfRange 2404	Message text: Value out of range %1 to %2
.vcTXEPrctMtWillBeAdjustedTo 2410	Message text: Will be adjusted to...
.vcTXERelTypeLongFF 3001	Text in the Edit links dialog: Finish-to-finish (FF)
.vcTXERelTypeLongFS 3000	Text in the Edit links dialog: Finish-to-start (FS)
.vcTXERelTypeLongSF 3003	Text in the Edit links dialog: Start-to-finish (SF)
.vcTXERelTypeLongSS 3002	Text in the Edit links dialog: Start-to-start (SS)
.vcTXERibAM 2225	ribbon text for am
.vcTXERibCW 2223	ribbon text for calendar week
.vcTXERibDay0 2212	ribbon text for Monday

	.vcTXERibDay1 2213	ribbon text for Tuesday
	.vcTXERibDay2 2214	ribbon text for Wednesday
	.vcTXERibDay3 2215	ribbon text for Thursday
	.vcTXERibDay4 2216	ribbon text for Friday
	.vcTXERibDay5 2217	ribbon text for Saturday
	.vcTXERibDay6 2218	ribbon text for Sunday
	.vcTXERibMon0 2200	ribbon text for January
	.vcTXERibMon1 2201	ribbon text for February
	.vcTXERibMon10 2210	ribbon text for November
	.vcTXERibMon11 2211	ribbon text for December
	.vcTXERibMon2 2202	ribbon text for March
	.vcTXERibMon3 2203	ribbon text for April
	.vcTXERibMon4 2204	ribbon text for Mai
	.vcTXERibMon5 2205	ribbon text for June
	.vcTXERibMon6 2206	ribbon text for July
	.vcTXERibMon7 2207	ribbon text for August
	.vcTXERibMon8 2208	ribbon text for September
	.vcTXERibMon9 2209	ribbon text for October
	.vcTXERiboClock 2224	ribbon text for o'clock
	.vcTXERibPM 2226	ribbon text for pm
	.vcTXERibQuar0 2219	ribbon text for first quarter
	.vcTXERibQuar1 2220	ribbon text for second quarter
	.vcTXERibQuar2 2221	ribbon text for third quarter
	.vcTXERibQuar3 2222	ribbon text for fourth quarter
⇌ textEntry	System.String	Text entry to replace the default text
⇌ returnStatus	VcReturnStatus	Return status
	Possible Values:	
	.vcRetStatDefault 2	The default behavior remains unchanged.
	.vcRetStatFalse 0	The default behavior will not be performed.
	.vcRetStatNoPopup 4	The popup of the context menu is inhibited.
	.vcRetStatOK 1	The default behavior will be performed.

• Pointer mode	vcTXECtxmenArrowMode
Mode: Create node	vcTXECtxmenCreateNodeMode
Mode: Create link	vcTXECtxmenCreateLinkMode
Mode: Create box	vcTXECtxmenCreateBoxMode
Build sub diagram	vcTXECtxmenSubDiagram
Restore full diagram	vcTXECtxmenFullDiagram
Page setup...	vcTXECtxmenPageLayout
Print setup...	vcTXECtxmenFilePrintSetup
Print preview...	vcTXECtxmenFilePrintPreview
Print...	vcTXECtxmenFilePrint
Show world view	vcTXECtxmenShowWorldView
Show legend view	vcTXECtxmenShowLegendView
Export graphics...	vcTXECtxmenGraphicExport

Constants of the diagram's context menu

Edit data...	vcTXECtxmenEditNode
Delete nodes	vcTXECtxmenDeleteNode
Build sub diagram	vcTXECtxmenSubDiagram
Restore full diagram	vcTXECtxmenFullDiagram
Outline outdent	vcTXECtxmenGroupOutlineOutdent
Outline indent	vcTXECtxmenGroupOutlineIndent

Constants of the context menu for nodes

Collapse Group	vcTXECtxmenGroupCollapse
Collapse Rows Below	vcTXECtxmenGroupCollapseRowsBelow
All Nodes In One Row	vcTXECtxmenGroupNodesInOneRow
Arrange Nodes Overlaid	vcTXECtxmenGroupNodesOverlaid
Delete group	vcTXECtxmenGroupDelete
Edit group data...	vcTXECtxmenEditGroup

Constants of the context menu for groups with no groups collapsed

Expand Group	vcTXECtxmenGroupExpand
Expand Rows Below	vcTXECtxmenGroupExpandRowsBelow
All Nodes In One Row	
Arrange Nodes Optimized	vcTXECtxmenGroupNodesOptimized
Delete group	
Edit group data...	

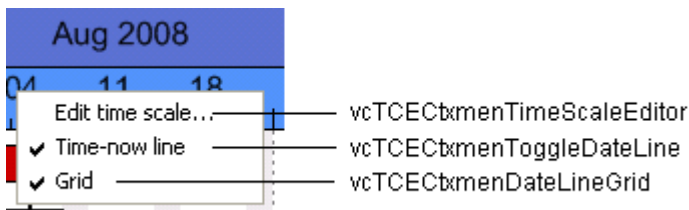
Constants of the context menu for groups with no groups expanded

Mode: Insert curve point	vcTXECtxmenInsertCurvePointMode
Delete curve point	vcTXECtxmenDeleteCurvePoint
Unmark all curves	vcTXECtxmenUnmarkAllCurves
Show world view	vcTXECtxmenShowWorldView
Show legend view	vcTXECtxmenShowLegendView
GeneratedElement	

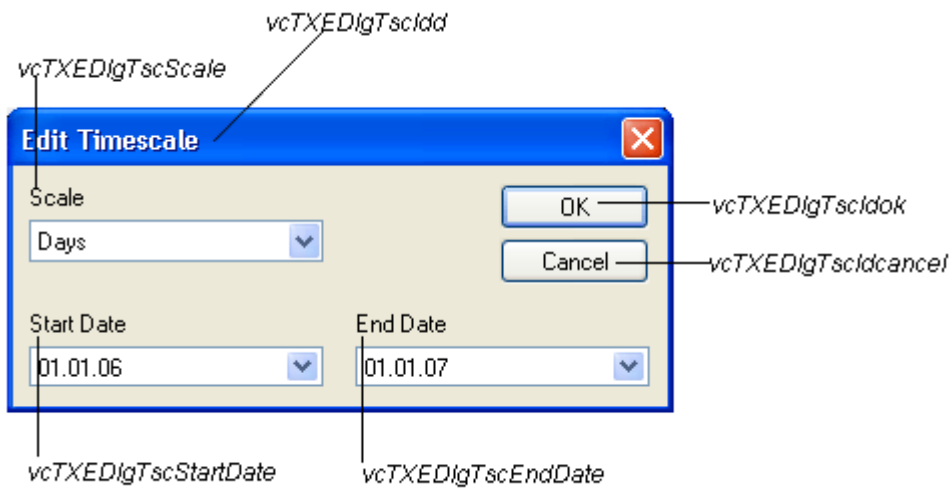
Constants of the context menu for histograms

✓ Show legend view	vcTXECtxmenShowLegendView
Actualize legend	vcTXECtxmenUpdateLegend
Legend attributes...	

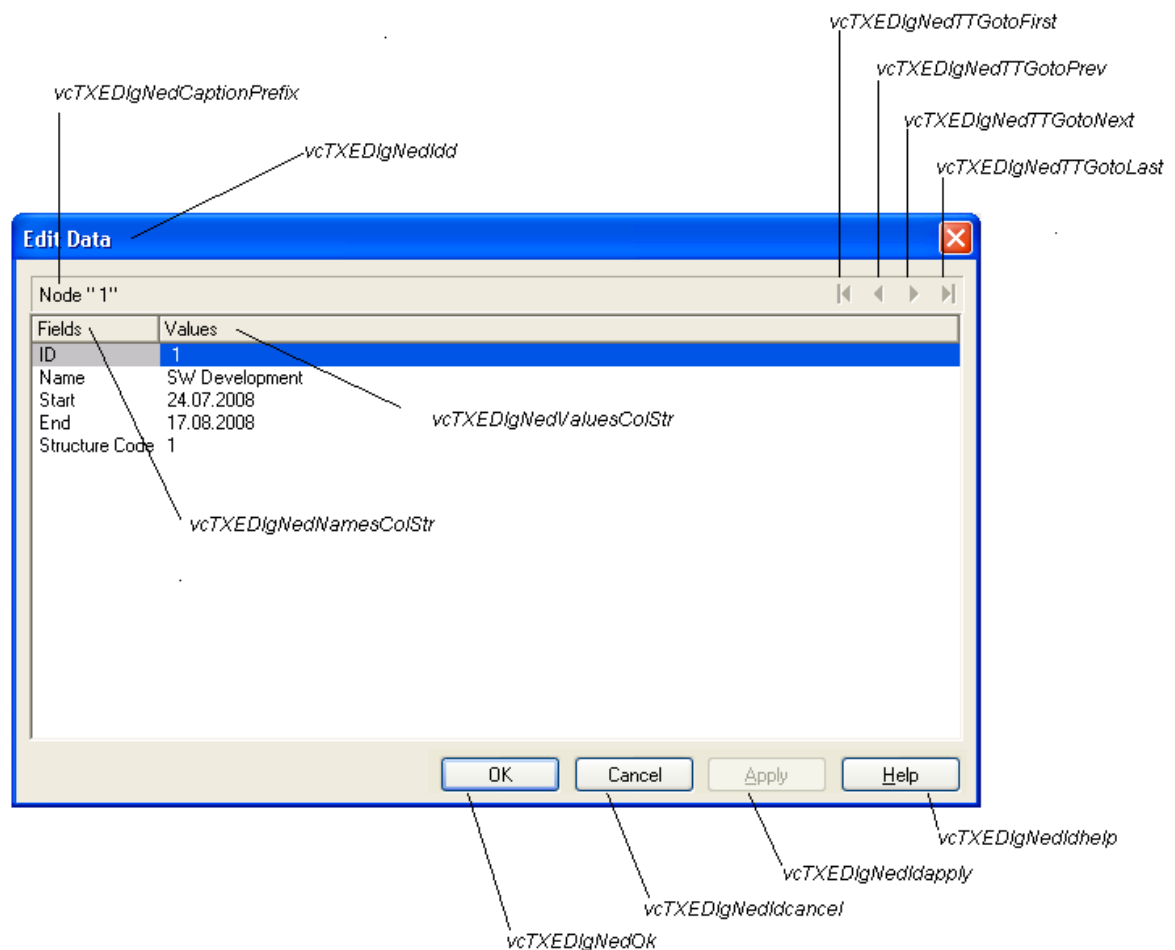
Constants of the legend's context menu



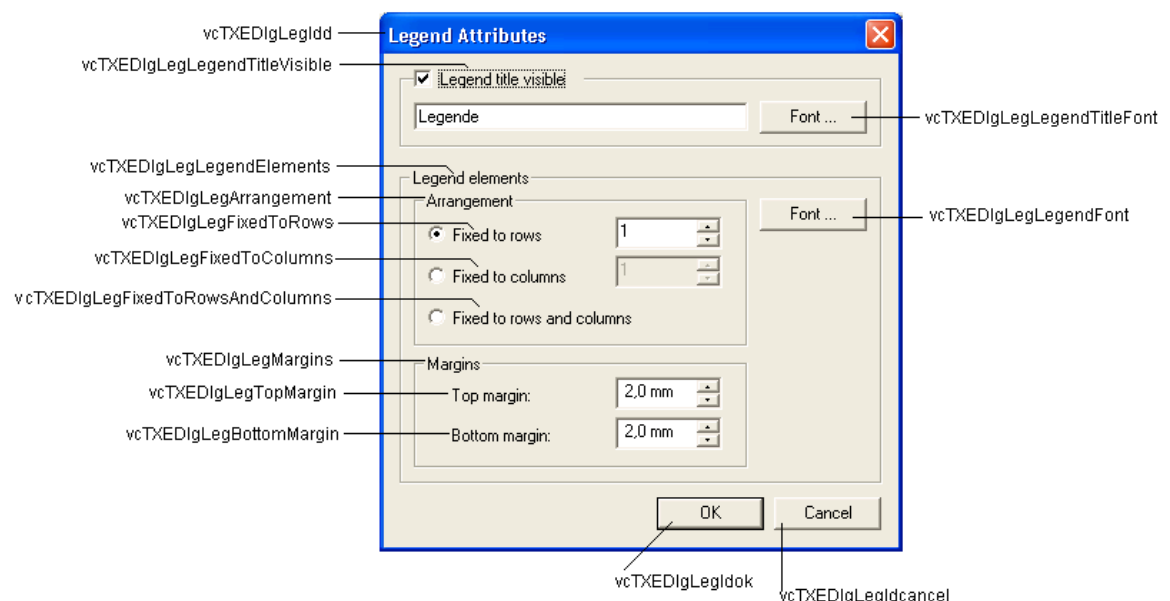
Constants of the time scale's context menu



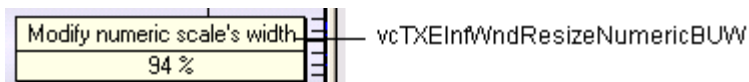
Constants of the dialog **Edit Time Scale**



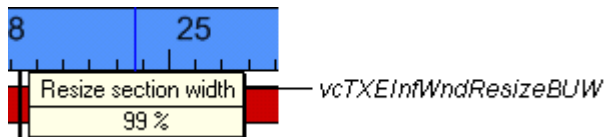
Constants of the dialogs **Edit data**, **Edit group** and **Edit link**, here illustrated by the **Edit data** dialog



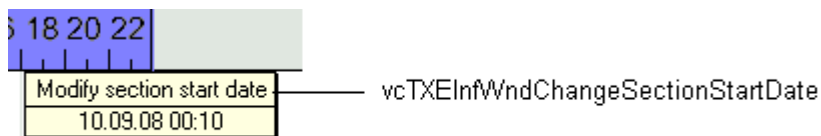
Constants of the **Legend attributes** dialog



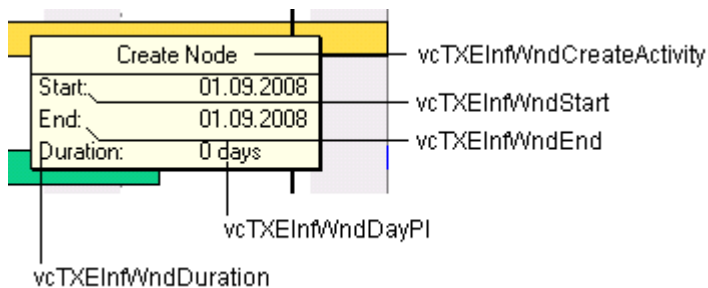
Constant of the tooltip text that appears on resizing the basic unit width of the **numeric scale in the histogram**



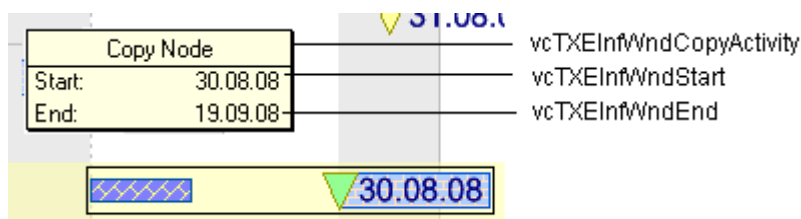
Constants of the tooltip text that appears on resizing the **time scale section width**



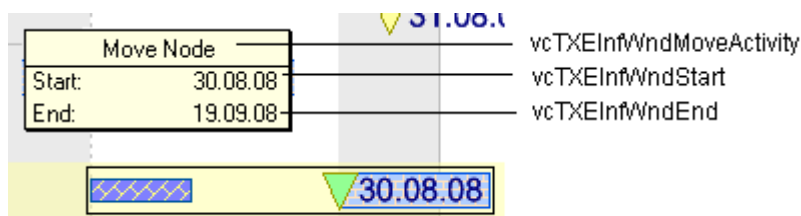
Constants of the tooltip text that appears on modifying the **start date of a time scale section**



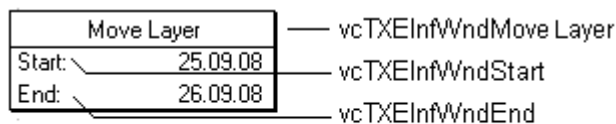
Constants of the tooltip text that appears on **creating a node**



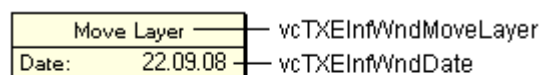
Constants of the tooltip text that appears on **copying a node**



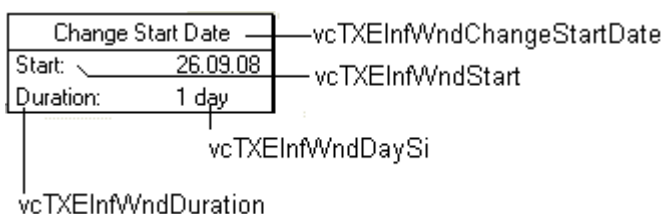
Constants of the tooltip text that appears on **moving a node**



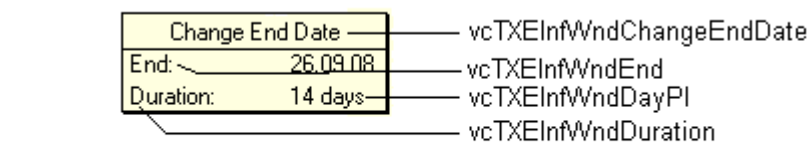
Constants of the tooltip text that appears on **moving a layer**



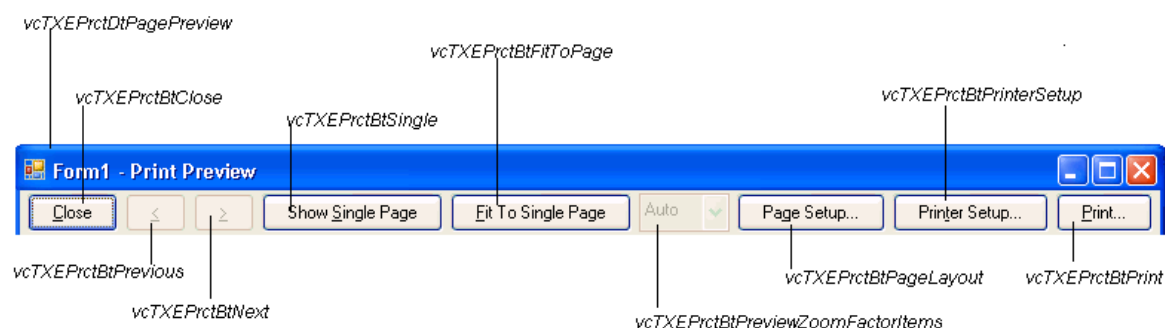
Constants of the tooltip text that appears on **moving a symbol layer**



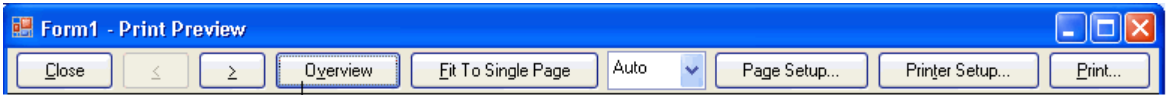
Constants of the tooltip text that appears on **modifying the start date of a node**



Constants of the tooltip text that appears on **modifying the end date of a node**

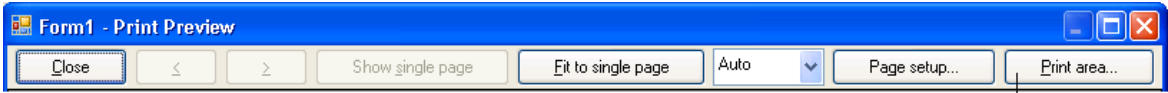


Constants of the button texts of the **Print Preview Overview**



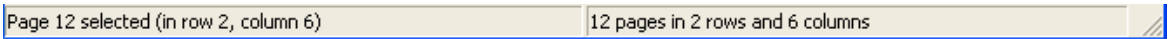
vcTXEPrctBtAll

Constants of the button texts of the **Print Preview** dialog



vcTXEPrctBtZoomPrint

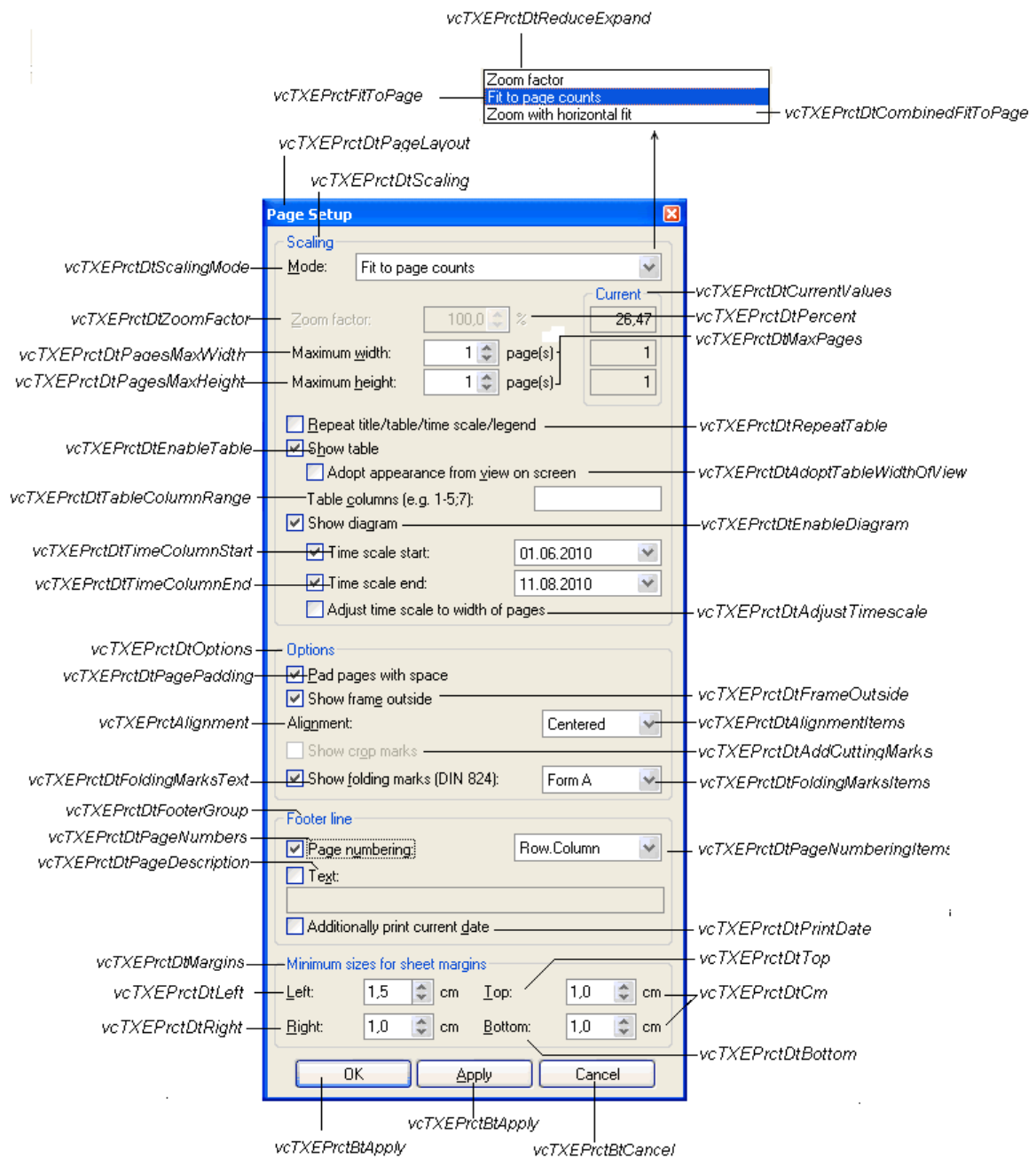
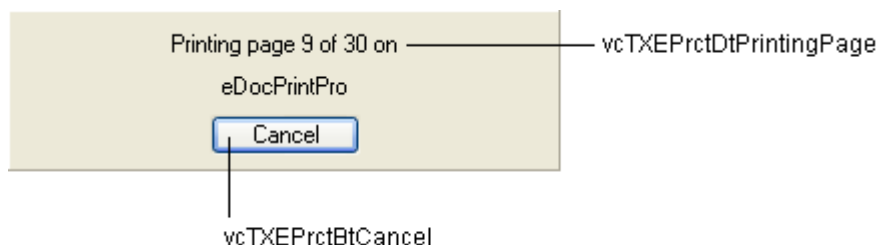
Constants of the button texts of the **Print Preview** dialog



vcTXEPrctDtStatusBarSelectedPage

vcTXEPrctDtStatusBarCurrentValues

Constants of the status bar in the dialog **Print Preview**

Constants of the **Page Setup** dialogConstants of the info box **Printing**

Example Code VB.NET

```

Private Sub VcGantt1_VcTextEntrySupplying(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcTextEntrySupplyingEventArgs) Handles
VcGantt1.VcTextEntrySupplying
    Select Case e.ControlIndex
        Case VcTextEntryIndex.vcTXEPrctBtNext
            e.Text = "Next page"
        Case VcTextEntryIndex.vcTXEPrctBtPrevious
            e.Text = "Previous page"
        End Select
    End Sub

```

Example Code C#

```

private void vcGantt1_VcTextEntrySupplying(object sender,
NETRONIC.XGantt.VcTextEntrySupplyingEventArgs e)
{
    switch (e.ControlIndex)
    {
        case VcTextEntryIndex.vcTXEPrctBtNext:
            e.Text = "Next page";
            break;
        case VcTextEntryIndex.vcTXEPrctBtPrevious:
            e.Text = "Previous page";
            break;
    }
}

```

VcTimeScaleEndModified**Event of VcGantt**

This event occurs when the modification of the end date of the time scale specified is completed.

	Data Type	Explanation
Properties:		
⇒ newEndDate	System.DateTime	New end date

VcTimeScaleLeftClicking**Event of VcGantt**

This event occurs when the user clicks the left mouse button on the timescale. The TimeScale object and the mouse position (x,y-coordinates) are returned.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcTimeScaleClickingEventArgs	Object specific to the event that is being handled

Properties of the VcTimeScaleClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ timeScale	VcTimeScale	Timescale hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```
Private Sub VcGantt1_VcTimeScaleLeftClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcTimeScaleClickingEventArgs) Handles
VcGantt1.VcTimeScaleLeftClicking
    VcGantt1.TimeScaleCollection.Active.BackgroundColor = Color.Blue
End Sub
```

Example Code C#

```
private void vcGantt1_VcTimeScaleLeftClicking(object sender,
NETRONIC.XGantt.VcTimeScaleClickingEventArgs e)
{
    vcGantt1.TimeScaleCollection.Active.BackgroundColor = Color.LightSteelBlue;
}
```

VcTimeScaleLeftDoubleClicking

Event of VcGantt

This event occurs when the user double-clicks the left mouse button on the timescale. The TimeScale object and the mouse position (x,y-coordinates) are returned. By setting the return status the appearance of the integrated dialog can be inhibited.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcTimeScaleClickingEventArgs	Object specific to the event that is being handled

Properties of the VcTimeScaleClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ timeScale	VcTimeScale	Timescale hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The Edit time scale dialog will not appear. The Edit time scale dialog will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcTimeScaleLeftDoubleClicking(ByVal sender As Object, ByVal e As NETRONIC.XGantt.VcTimeScaleClickingEventArgs) Handles VcGantt1.VcTimeScaleLeftDoubleClicking
    MsgBox("Show your own dialog")
    e.ReturnStatus = VcReturnStatus.vcRetStatFalse
End Sub
```

Example Code C#

```
private void vcGantt1_VcTimeScaleLeftDoubleClicking(object sender, NETRONIC.XGantt.VcTimeScaleClickingEventArgs e)
{
    MessageBox.Show("Show your own dialog");
    e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcTimeScaleModified

Event of VcGantt

This event occurs when zooming of the time scale specified is completed.

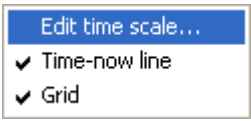
	Data Type	Explanation
Properties:		
⇒ timeScale	VcTimeScale	Time scale modified

VcTimeScaleRightClicking

Event of VcGantt

This event occurs when the user clicks the right mouse button on the timescale. The TimeScale object and the mouse position (x,y-coordinates) are returned. By setting the return status you can inhibit the integrated context

menu to pop up an replace it by a context menu of your own at the location delivered.



Above: integrated context menu

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcTimeScaleClickingEventArgs	Object specific to the event that is being handled

Properties of the VcTimeScaleClickingEventArgs object

	Data Type	Explanation
Properties:		
⇒ timeScale	VcTimeScale	Timescale hit
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y coordinate of the mouse cursor
↔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatNoPopup 4 .vcRetStatOK 1	The context menu will be inhibited. The context menu will appear.

Example Code VB.NET

```
Private Sub VcGantt1_VcTimeScaleRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcTimeScaleClickingEventArgs) Handles
VcGantt1.VcTimeScaleRightClicking
    PopupMenu.Show(VcGantt1, New Point(e.X, e.Y))
    e.ReturnStatus = NETRONIC.XGantt.VcReturnStatus.vcRetStatNoPopup
End Sub
```

Example Code C#

```
private void vcGantt1_VcTimeScaleRightClicking(object sender,
NETRONIC.XGantt.VcTimeScaleClickingEventArgs e)
{
    PopupMenu.Show(vcGantt1, new Point (e.X, e.Y));
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

VcTimeScaleSectionRescaled

Event of VcGantt

This event occurs when the user has finished rescaling a time scale section. The TimeScale object, the section index and the new basicUnitWidth are passed.

	Data Type	Explanation
Properties:		
⇒ timeScale	VcTimeScale	Timescale
⇒ sectionIndex	System.Int16	Section index
⇒ newBasicUnitWidth	System.Int32	New width of the basic unit

VcTimeScaleSectionRescaledEx

Event of VcGantt

This event occurs when the user has finished rescaling a time scale section. The TimeScale object, the section index and the new basicUnitWidth are passed.

	Data Type	Explanation
Properties:		
⇒ timeScale	VcTimeScale	Timescale
⇒ sectionIndex	System.Int16	Section index
⇒ newBasicUnitWidth	System.Double	New width of the basic unit

VcTimeScaleSectionRescaling

Event of VcGantt

This event occurs when the user rescales a section of the timescale. The TimeScale object, the section index and the current BasicUnitWidth are returned. By setting the return status you can inhibit the modification.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event

⇒ e	VcTimeScaleSectionRescalingEventArgs	Object specific to the event that is being handled
-----	--------------------------------------	--

Properties of the VcTimeScaleSectionRescalingEventArgs object

	Data Type	Explanation
Properties:		
⇒ timeScale	VcTimeScale	Timescale
⇒ sectionIndex	System.Int16	Section index
⇒ newBasicUnitWidth	System.Int32	New width of the basic unit
⇌ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The time scale section will not be modified. The time scale section will be modified.

Example Code VB.NET

```
Private Sub VcGantt1_VcTimeScaleSectionRescaling(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcTimeScaleSectionRescalingEventArgs) Handles
VcGantt1.VcTimeScaleSectionRescaling
    If e.NewBasicUnitWidth <= 1000 Then
        MsgBox("New basic unit width: " + e.NewBasicUnitWidth)
    Else
        MsgBox("The maximum basic unit width is 1000")
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcTimeScaleSectionRescaling(object sender,
NETRONIC.XGantt.VcTimeScaleSectionRescalingEventArgs e)
{
    if (e.NewBasicUnitWidth <= 1000)
        MessageBox.Show("New basic unit width: " + e.NewBasicUnitWidth);
    else
    {
        MessageBox.Show("The maximum basic unit width is 1000");
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
    }
}
```

VcTimeScaleSectionRescalingEx

Event of VcGantt

This event occurs when the user rescales a section of the timescale. The TimeScale object, the section index and the current BasicUnitWidth are returned. By setting the return status you can inhibit the modification.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcTimeScaleSectionRescalingExEventArgs	Object specific to the event that is being handled

Properties of the VcTimeScaleSectionRescalingExEventArgs object

	Data Type	Explanation
Properties:		
⇒ timeScale	VcTimeScale	Timescale
⇒ sectionIndex	System.Int16	Section index
⇒ newBasicUnitWidth	System.Double	New width of the basic unit
⇔ returnStatus	VcReturnStatus	Return status

Example Code VB.NET

```
Private Sub VcGantt1_VcTimeScaleSectionRescalingEx(ByVal sender As Object, ByVal e As NETRONIC.XGantt.VcTimeScaleSectionRescalingExEventArgs) Handles VcGantt1.VcTimeScaleSectionRescalingEx
    If e.NewBasicUnitWidth <= 1000 Then
        MsgBox("New basic unit width: " + e.NewBasicUnitWidth)
    Else
        MsgBox("The maximum basic unit width is 1000")
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcTimeScaleSectionRescalingEx(object sender, NETRONIC.XGantt.VcTimeScaleSectionRescalingExEventArgs e)
{
    if (e.NewBasicUnitWidth <= 1000)
        MessageBox.Show("New basic unit width: " + e.NewBasicUnitWidth);
    else
    {
        MessageBox.Show("The maximum basic unit width is 1000");
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
    }
}
```

VcTimeScaleSectionStartModifying

Event of VcGantt

This event occurs when the user modifies the start date of a section interactively. The TimeScale object, the section index and the current start date are returned.

The data passed by this event can be read, but must not be modified. For modifying them please use the event **VcTimeScaleSectionStartModified**.

By setting the return status the modification can be inhibited.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcTimeScaleSectionStartModifyingEventArgs	Object specific to the event that is being handled

Properties of the VcTimeScaleSectionStartModifyingEventArgs object

	Data Type	Explanation
Properties:		
⇒ timeScale	VcTimeScale	Timescale
⇒ sectionIndex	System.Int16	Section index
⇒ newStartDate	System.DateTime	Date
⇔ returnStatus	VcReturnStatus	Return status
	Possible Values: .vcRetStatFalse 0 .vcRetStatOK 1	The modification will be revoked. The modification will be accepted.

Example Code VB.NET

```
Private Sub VcGantt1_VcTimeScaleSectionStartModifying(ByVal sender As Object,
ByVal e As NETRONIC.XGantt.VcTimeScaleSectionStartModifyingEventArgs) Handles
VcGantt1.VcTimeScaleSectionStartModifying
    If MsgBox("Do you want to change the start of section No. " +
e.SectionIndex.ToString() + " to " + e.NewStartDate + "?", MsgBoxStyle.OKCancel)
= MsgBoxResult.Cancel Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
    End If
End Sub
```

Example Code C#

```
private void vcGantt1_VcTimeScaleSectionStartModifying(object sender,
NETRONIC.XGantt.VcTimeScaleSectionStartModifyingEventArgs e)
{
    DialogResult retVal = MessageBox.Show("Do you want to change the start of
section No " + e.SectionIndex.ToString() + " to " + e.NewStartDate + " ?",
"Changing numeric scale", MessageBoxButtons.OKCancel);
    if (retVal == DialogResult.Cancel)
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
}
```

VcTimeScaleStartModified

Event of VcGantt

This event occurs when the modification of the start date of the time scale specified is completed.

	Data Type	Explanation
Properties:		
⇒ newStartDate	System.DateTime	New start date

VcToolTipTextSupplying

Event of VcGantt

This event occurs if the VcGantt property **ToolTipTextSupplyingEvent-Enabled** is set to **True** or if the check box **VcToolTipSupplying events** on the **General** property page is activated. You can use this event for displaying information on the object hit by tooltip texts. The event occurs when the cursor moves on a VcGantt object. The event returns the object, the object type and the coordinates of the mouse position. By setting the returnStatus to **vcRetStatFalse** you can revoke the tooltip.

In case of a calendar grid, a tool tip text will only be retrieved if the calendar grid could be identified; i.e. if the calendar grid property **Identifiable** had been set to **True**.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcToolTipTextSupplyingEventArgs	Object specific to the event that is being handled

Properties of the VcToolTipTextSupplyingEventArgs object

	Data Type	Explanation
Properties:		
⇒ hitObject	VcObject	Object hit
⇒ hitObjectType	VcObjectType	Type of the object hit
	Possible Values: .vcObjTypeBox 15	object type box

	.vcObjTypeCalendarGrid 18 .vcObjTypeCurve 12 .vcObjTypeDateLine 9 .vcObjTypeGroup 7 .vcObjTypeGroupInDiagram 11 .vcObjTypeGroupInTable 7 .vcObjTypeHistogram 13 .vcObjTypeLayer 8 .vcObjTypeLinkCollection 3 .vcObjTypeNodeInDiagram 2 .vcObjTypeNodeInLegend 17 .vcObjTypeNodeInTable 1 .vcObjTypeNone 0 .vcObjTypeNumericScale 10 .vcObjTypeSummaryNode 14 .vcObjTypeTable 4 .vcObjTypeTableCaption 5 .vcObjTypeTimeScale 6	object type calendar grid object type curve object type date line object type group object type group in diagram area object type group in table area object type histogram object type layer object type link collection object type node in diagram area object type node in legend area object type node in table area no object object type numeric scale object type summary bar object type table object type table caption object type time scale
⇒ x	System.Int32	X coordinate of the mouse cursor
⇒ y	System.Int32	Y value of the mouse cursor
⇌ tooltipText	System.String	Tooltip text, ASP editions: no restriction Other editions: 1024 characters maximum
⇌ returnStatus	VcReturnStatus Possible Values: .vcRetStatDefault 2 .vcRetStatFalse 0 .vcRetStatNoPopup 4 .vcRetStatOK 1	Return status The default behavior remains unchanged. The default behavior will not be performed. The popup of the context menu is inhibited. The default behavior will be performed.

Example Code VB.NET

```

Private Sub VcGantt1_VcToolTipTextSupplying(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcToolTipTextSupplyingEventArgs) Handles
VcGantt1.VcToolTipTextSupplying

    Dim node As VcNode
    If Convert.ToString(e.HitObject) = "NETRONIC.XGantt.VcNode" Then
        node = DirectCast(e.HitObject, VcNode)
        Select Case e.HitObjectType
            Case VcObjectType.vcObjTypeNodeInDiagram
                e.Text = Convert.ToString(node.DataField(1))
            Case VcObjectType.vcObjTypeNodeInTable
                e.Text = Convert.ToString(node.DataField(1))
        End Select
    End If
End Sub

```

Example Code C#

```

private void vcGantt1_VcToolTipTextSupplying(object sender,
NETRONIC.XGantt.VcToolTipTextSupplyingEventArgs e)
{
    VcNode node;
    if (e.HitObject.ToString() == "NETRONIC.XGantt.VcNode")
    {
        node = (VcNode)e.HitObject;
        switch (e.HitObjectType)
        {
            case VcObjectType.vcObjTypeNodeInDiagram:
                e.Text = Convert.ToString(node.get_DataField(1));
                break;
            case VcObjectType.vcObjTypeNodeInTable:
                e.Text = Convert.ToString(node.get_DataField(1));
                break;
        }
    }
}

```

VcViewComponentsSizeModified**Event of VcGantt**

This event occurs when at run time the size of a graphical element of the VARCHART Windows Forms control (time scale, diagram, histogram, table, table caption etc.) was modified. To react to the event by the API, you have to retrieve the position and the size of all graphical elements of the VARCHARTWindows Forms control.

Note:

1. The position refers to the origin of the graphical element of the VARCHART Windows Forms control.
2. The values returned are pixel values.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcViewComponentsSizeModifiedEventArgs	Object specific to the event that is being handled

Properties of the VcViewComponentsSizeModifiedEventArgs object

	Data Type	Explanation
Properties:		
⇒ (no parameter)		

Example Code VB.NET

```
Private Sub VcGantt1_VcViewComponentsSizeModified(ByVal sender As Object, ByVal e As NETRONIC.XGantt.VcViewComponentsSizeModifiedEventArgs) Handles VcGantt1.VcViewComponentsSizeModified
```

```
    Dim x As Integer
    Dim y As Integer
    Dim width As Integer
    Dim height As Integer
```

```
VcGantt1.GetViewComponentSize(VcComponentType.vcHistogramVerScaleComponent, x, y, width, height)
```

```
    ' plus 6 because of the sash
    TextBox1.Top = VcGantt1.Top + y + 6
    TextBox1.Left = VcGantt1.Left + x
    ' minus 25 because of the numeric scale
    TextBox1.Width = Width - 25
    ' minus 6 because of the sash
    TextBox1.Height = Height - 6
End Sub
```

Example Code C#

```
private void vcGantt1_VcViewComponentsSizeModified(object sender, NETRONIC.XGantt.VcViewComponentsSizeModifiedEventArgs e)
{
    int x = 0;
    int y = 0;
    int width = 0;
    int height = 0;
    vcGantt1.GetViewComponentSize(VcComponentType.vcHistogramVerScaleComponent, ref x, ref y, ref width, ref height);
    //plus 6 because of the sash
    textBox1.Top = vcGantt1.Top + y + 6;
    textBox1.Left = vcGantt1.Left + x;
    //minus 25 because of the numeric scale
    textBox1.Width = Width - 25;
    //minus 6 because of the sash
    textBox1.Height = Height - 6;
}
```

VcWorldViewClosed**Event of VcGantt**

This event occurs when the worldview popup window is closed.

	Data Type	Explanation
Parameter:		
⇒ sender	VcGantt	Reference to the object that triggered the event
⇒ e	VcWorldViewClosedEventArgs	Object specific to the event that is being handled

Properties of the VcWorldViewClosedEventArgs object

	Data Type	Explanation
Properties:		
⇨ (no parameter)		

Example Code VB.NET

```
Private Sub VcGantt1_VcWorldViewClosed(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcWorldViewClosedEventArgs) Handles VcGantt1.VcWorldViewClosed
    MsgBox("Do you want to close the worldview window?", MsgBoxStyle.OKCancel)
End Sub
```

Example Code C#

```
private void vcGantt1_VcWorldViewClosed(object sender,
NETRONIC.XGantt.VcWorldViewClosedEventArgs e)
{
    DialogResult retVal = MessageBox.Show("Do you want to close the worldview
window?", "Closing worldview window", MessageBoxButtons.OKCancel);
}
```

VcZoomFactorModified

Event of VcGantt

This events occurs if the user modified the size of the rectangle in the world view or if he zoomed marked objects. You can zoom smoothly by keeping the **Ctrl** key pressed while turning the mouse wheel, or in discrete steps while using the **Plus** or **Minus** keys in the number pad.

	Data Type	Explanation
Parameter:		
⇨ sender	VcGantt	Reference to the object that triggered the event
⇨ e	VcZoomFactorModifiedEventArgs	Object specific to the event that is being handled

Properties of the VcZoomFactorModifiedEventArgs object

	Data Type	Explanation
Properties:		
⇨ (no parameter)		

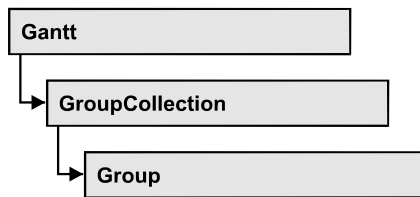
Example Code VB.NET

```
Private Sub VcGantt1_VcZoomFactorModified(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcZoomFactorModifiedEventArgs) Handles
VcGantt1.VcZoomFactorModified
    MsgBox("Zoomfactor: " + VcGantt1.ZoomFactor)
End Sub
```

Example Code C#

```
private void vcGantt1_VcZoomFactorModified(object sender,
NETRONIC.XGantt.VcZoomFactorModifiedEventArgs e)
{
    MessageBox.Show("Zoomfactor: " + vcGantt1.ZoomFactor.ToString());
}
```

7.33 VcGroup



A group contains all nodes that have the same value in the grouping field. This value can be retrieved as group name. The nodes that form a group can be accessed by the NodeCollection property.

Properties

- BodyCollapsed
- DataField
- GroupingLevel
- GroupInvisible
- ID
- Marked
- Name
- NodeCollection
- NodesAndGroupsBelowCollapsed
- NodesInHeader
- NodesOverlaid
- SubGroups
- SuperGroup
- Visible

Methods

- DataRecord
- Delete
- RelatedDataRecord
- ReOptimizeNodes
- Update

Properties

BodyCollapsed

Property of VcGroup

This property lets you set or retrieve whether (True) or not (False) a group is collapsed. This property can only be used in the clustering mode (GroupMode = vcGMClustering). The property also can be set interactively when the property VcGantt.GroupInteractionsAllowed is set.

	Data Type	Explanation
Parameter: ⇒ Rückgabewert	System.Boolean	Group collapsed/expanded
Property value	System.Boolean	Group collapsed/expanded

Example Code VB.NET

```
body.Collapsed = True
```

Example Code C#

```
body.Collapsed = true;
```

DataField

Property of VcGroup

This property lets you set or retrieve the contents of a DataField of the group record. The group record is a copy of the node record of the first node added to the group. The data field referred to by its field index. To update the group, the **Update** method needs to be invoked.

The property DataField is an Indexed Property, which in C# is addressed by the methods set_DataField (index, pvn) and get_DataField (index).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the data field
Property value	Void	

Example Code VB.NET

```

Dim groupCltn As VcGroupCollection
Dim group As VcGroup
Dim nodeCltn As VcNodeCollection
Dim node As VcNode

groupCltn = VcGantt1.GroupCollection
For Each group In groupCltn
    nodeCltn = group.NodeCollection
    For Each node In nodeCltn
        If node.DataField(3) > group.DataField(3) Then
            group.DataField(3) = node.DataField(3)
        End If
    Next
Next
group.Update()
Next

```

Example Code C#

```

VcGroupCollection groupCltn = vcGantt1.GroupCollection;
foreach (VcGroup group in groupCltn)
{
    VcNodeCollection nodeCltn = vcGantt1.NodeCollection;
    foreach (VcNode node in nodeCltn)
    {
        if (node.get_DataField(3) > group.get_DataField(3))
            group.set_DataField(3,node.get_DataField(3));
    }
    group.Update();
}

```

GroupingLevel**Read Only Property of VcGroup**

This property lets you retrieve the grouping level of the group, if there are several levels of grouping. At maximum, 25 grouping levels are possible.

	Data Type	Explanation
Property value	System.Int16	Grouping level of the group

Example Code VB.NET

```

Dim group As VcGroup
Dim subGroup As VcGroup
Dim nodeCltn As VcNodeCollection
Dim node As VcNode

nodeCltn = VcGantt1.NodeCollection
node = nodeCltn.FirstNode
group = node.SuperGroup
If group.GroupingLevel > 0 Then
    subGroup = group.SuperGroup
End If

```

Example Code C#

```
VcNodeCollection nodeCltn = vcGantt1.NodeCollection;
VcNode node = nodeCltn.FirstNode();
VcGroup group = node.SuperGroup;
VcGroup subGroup;

if (group.GroupingLevel > 0)
    subGroup = group.SuperGroup;
```

GroupInvisible

Property of VcGroup

This property lets you set or retrieve whether this group is to be displayed. The default value is the value that was specified in the group level layout.

	Data Type	Explanation

ID

Read Only Property of VcGroup

By this property you can retrieve the ID of a group.

	Data Type	Explanation
Property value	System.String	Group ID

Example Code VB.NET

```
Code-Beispiel VB.NET
Dim groupCltn As VcGroupCollection
Dim group As VcGroup
Dim groupID As String
groupCltn = VcGantt1.GroupCollection
group = groupCltn.FirstGroup
groupID = group.ID

MsgBox (group.ID)
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
VcGroup group = groupCltn.FirstGroup();
string groupID = group.ID;
MessageBox.Show(group.ID);
```

Marked

Property of VcGroup

This property lets you set or retrieve whether a group is marked.

	Data Type	Explanation

Example Code VB.NET

```
Dim groupCltn As VcGroupCollection
Dim group As VcGroup

groupCltn = VcGantt1.GroupCollection
groupCltn.SelectGroups(VcSelectionType.vcSelected)

For Each group In groupCltn
    group.Marked = False
Next
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
groupCltn.SelectGroups(VcSelectionType.vcAll);

foreach (VcGroup group in groupCltn)
{
    Group.Marked = false;
}
```

Name

Read Only Property of VcGroup

This property lets you retrieve the name of a group (= the value of the grouping field GroupField).

	Data Type	Explanation
Property value	System.String	Group name

Example Code VB.NET

```
Dim groupCltn As VcGroupCollection
Dim group As VcGroup
Dim groupName As String

groupCltn = VcGantt1.GroupCollection
group = groupCltn.FirstGroup
groupName = group.Name
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
VcGroup group = groupCltn.FirstGroup();
string groupName = group.Name;
```

NodeCollection

Read Only Property of VcGroup

This property lets you access all nodes that belong to a group.

	Data Type	Explanation
Property value	VcNodeCollection	NodeCollection object

Example Code VB.NET

```
Dim groupCltn As VcGroupCollection
Dim group As VcGroup
Dim nodeCltn As VcNodeCollection

groupCltn = VcGantt1.GroupCollection
group = groupCltn.FirstGroup
nodeCltn = group.NodeCollection
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
VcGroup group = groupCltn.FirstGroup();
VcNodeCollection nodeCltn = group.NodeCollection;
```

NodesAndGroupsBelowCollapsed

Property of VcGroup

This property applies to multi-level grouping (n levels), that is, to the levels from no.1 to (n-1). If you have chosen for the group all nodes in one row, setting this property to **True** will collapse only the subgroups of the selected group. If instead you collapse the group using the **Collapsed** property, in addition groups that do not belong to a subgroup will be collapsed as well.

	Data Type	Explanation
Property value	System.Boolean	Rows below the top row are/are not collapsed

Example Code VB.NET

```
Dim groupCltn As VcGroupCollection
Dim group As VcGroup

groupCltn = VcGantt1.GroupCollection
group = groupCltn.GroupByName("A")

group.NodesAndGroupsBelowCollapsed = True
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
VcGroup group = groupCltn.GroupByName("A");
group.NodesAndGroupsBelowCollapsed = true;
```

NodesInHeader

Property of VcGroup

This property lets you set or retrieve whether (True) or not (False) the node objects of the group are positioned the same row.

	Data Type	Explanation

NodesOverlaid

Property of VcGroup

This property lets you set or retrieve whether (False) the node layout is optimized or if nodes overlap (True).

	Data Type	Explanation
Parameter:		
↔ Rückgabewert	System.Boolean	The node layout is/is not at its optimum
Property value	System.Boolean	The node layout is/is not at its optimum

SubGroups

Read Only Property of VcGroup

In a multi-level grouping arrangement, this property lets you retrieve subgroups, that are returned by a group collection object.

	Data Type	Explanation
Property value	VcGroupCollection	GroupCollection object containing the subgroups

Example Code VB.NET

```
Dim groupCltn As VcGroupCollection
Dim group As VcGroup
Dim subGroupCltn As VcGroupCollection

groupCltn = VcGantt1.GroupCollection
group = groupCltn.GroupByName("A")
subGroupCltn = group.SubGroups
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
VcGroup group = groupCltn.GroupByName("A");
VcGroupCollection subGroupCltn = group.SubGroups;
```

SuperGroup

Read Only Property of VcGroup

In a multi-level grouping arrangement, this property lets you enquire the parent group of this group.

	Data Type	Explanation
Property value	VcGroup	Parent group

Example Code VB.NET

```
Dim group As VcGroup
Dim subGroup As VcGroup
Dim nodeCltn As VcNodeCollection
Dim node As VcNode

nodeCltn = VcGantt1.NodeCollection
node = nodeCltn.FirstNode
group = node.SuperGroup
If group.GroupingLevel > 0 Then
    superGroup = group.SuperGroup
End If
```

Example Code C#

```
VcNodeCollection nodeCltn = vcGantt1.NodeCollection;
VcNode node = nodeCltn.FirstNode();
VcGroup group = node.SuperGroup;
VcGroup superGroup;

if (group.GroupingLevel > 0)
    superGroup = group.SuperGroup;
```

Visible

Property of VcGroup

This property lets you set or retrieve whether (True) or not (False) this group is visible.

	Data Type	Explanation
Property value	System.Boolean	Group visible/invisible

Example Code VB.NET

```
Dim groupCltn As VcGroupCollection
Dim group As VcGroup

groupCltn = VcGantt1.GroupCollection
group = groupCltn.GroupByName("A")
group.Visible = True
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
VcGroup group = groupCltn.GroupByName("A");

group.Visible = true;
```

Methods

DataRecord

Method of VcGroup

This property lets you retrieve the group as a data record object. The properties of the data record object give access to the corresponding data table and the data table collection.

	Data Type	Explanation
Return value	VcDataRecord	Data record returned

Delete

Method of VcGroup

This method lets you delete a group. Deleting a group is possible only if the group is empty. Activities have to be deleted from the group before the group can be deleted.

	Data Type	Explanation
Return value	System.Boolean	Group was/was not deleted successfully

Example Code VB.NET

```
Dim groupCltn As VcGroupCollection
Dim group As VcGroup
Dim nodeCltn As VcNodeCollection
Dim node As VcNode

groupCltn = vcGantt1.GroupCollection
group = groupCltn.GroupByName("A")
nodeCltn = group.NodeCollection
For Each node In nodeCltn
    node.Delete()
Next
group.Delete()
```


Example Code C#

```

VcGroupCollection groupCltn = vcGantt1.GroupCollection;
VcGroup group = groupCltn.GroupByName("A");
VcNodeCollection nodeCltn = group.NodeCollection;

foreach (VcNode node in nodeCltn)
{
    node.Delete();
}
group.Delete();

```

RelatedDataRecord**Method of VcGroup**

This property lets you retrieve a data record from a data table that is related to the group data table. The index passed by the parameter denotes the field in the data record that holds the key of the related data record.

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of data field that holds the key
Return value	VcDataRecord	Related data record returned

ReOptimizeNodes**Method of VcGroup**

If the property **VcGantt.GroupOptimizationOnInteractionsEnabled** was set to **false** and if the nodes of the group are in the optimized state of display, this property allows to manually update the optimized arrangement after an interaction.

	Data Type	Explanation
Return value	Void	

Update**Method of VcGroup**

This method lets you update a group after having changed a data field by the **DataField** property.

	Data Type	Explanation
Return value	System.Boolean	Group successfully/not successfully updated

Example Code VB.NET

```

Dim groupCltn As VcGroupCollection
Dim group As VcGroup
Dim nodeCltn As VcNodeCollection
Dim node As VcNode

groupCltn = VcGantt1.GroupCollection
group = groupCltn.GroupByName("A")
nodeCltn = group.NodeCollection

group.DataField(3) = nodeCltn.FirstNode.DataField(3)
For Each node In nodeCltn
    If node.DataField(3) > group.DataField(3) Then
        group.DataField(3) = node.DataField(3)
    End If
Next
group.Update()

```

Example Code C#

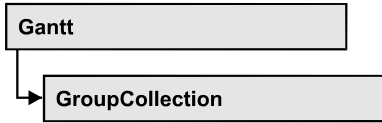
```

VcGroupCollection groupCltn = vcGantt1.GroupCollection;
VcGroup group = groupCltn.GroupByName("A");
VcNodeCollection nodeCltn = group.NodeCollection;

group.set_DataField(3, nodeCltn.FirstNode().get_DataField(3));
foreach(VcNode node in nodeCltn)
{
    if (node.get_DataField(3) > group.get_DataField(3))
        group.set_DataField(3, node.get_DataField(3));
}
group.Update();

```

7.34 VcGroupCollection



If nodes were grouped, an object of the type VcGroupCollection contains all available groups. You can access all objects in an iterative loop by **For Each group In GroupCollection** or by the methods **First...** and **Next....** You can access a single group using the method **GroupByName**. The number of groups in the collection object can be retrieved by the property **Count**.

Properties

- Count

Methods

- FirstGroup
- GetEnumerator
- GroupByName
- NextGroup
- SelectGroups

Properties

Count

Read Only Property of VcGroupCollection

This property lets you retrieve the number of groups in the group collection.

	Data Type	Explanation
Property value	System.Int32	Number of nodes

Example Code VB.NET

```
Dim groupCltn As VcGroupCollection
Dim group As VcGroup
Dim numberOfGroups As Integer

groupCltn = VcGantt1.GroupCollection
numberOfGroups = groupCltn.Count
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
int numberOfGroups = groupCltn.Count;
```

Methods

FirstGroup

Method of VcGroupCollection

This method can be used to access the initial value, i.e. the first group of a group collection, and then to continue in a forward iteration loop by the method **NextGroup** for the groups following. If there is no group in the group collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcGroup	First group of the GroupCollection

Example Code VB.NET

```
Dim groupCltn As VcGroupCollection
Dim group As VcGroup

groupCltn = VcGantt1.GroupCollection
group = groupCltn.FirstGroup
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
VcGroup group = groupCltn.FirstGroup();
```

GetEnumerator

Method of VcGroupCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the group objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

GroupName

Method of VcGroupCollection

By this method you can get a group by its name. If a group of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter:		
⇒ Rückgabewert	VcGroup	Group
⇒ groupName	System.String	Name of group
Return value	VcGroup	Group

Example Code VB.NET

```
Dim groupCltn As VcGroupCollection
Dim group As VcGroup

groupCltn = VcGantt1.GroupCollection
group = groupCltn.GroupByName("Group A")
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
VcGroup group = groupCltn.GroupByName("A");
```

NextGroup

Method of VcGroupCollection

This method can be used in a forward iteration loop to retrieve subsequent groups from a group collection after initializing the loop by the method **FirstGroup**. If there is no group left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcGroup	Subsequent group

Example Code VB.NET

```
Dim groupCltn As VcGroupCollection
Dim group As VcGroup

groupCltn = VcGantt1.GroupCollection
group = groupCltn.FirstGroup
While Not group Is Nothing
    ListBox1.Items.Add(group.Name)
    group = groupCltn.NextGroup
End While
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
VcGroup group = groupCltn.FirstGroup();
while (group != null)
{
    listBox1.Items.Add(group.Name);
    group = groupCltn.NextGroup();
}
```

SelectGroups**Method of VcGroupCollection**

This method lets you specify the groups that the group collection is to contain.

	Data Type	Explanation
Parameter: ⇒ groupSelType	VcGroupSelectionType Possible Values: .vcAllGroups 0 .vcCollapsedGroups 1 .vcExpandedGroups 2 .vcInvisibleGroups 5 .vcSelectedGroups 3 .vcVisibleGroups 4	Type of group to be selected All groups selected Collapsed groups selected Expanded groups selected Invisible groups selected Selected groups selected Visible groups selected
Return value	System.Int32	Number of groups selected

Example Code VB.NET

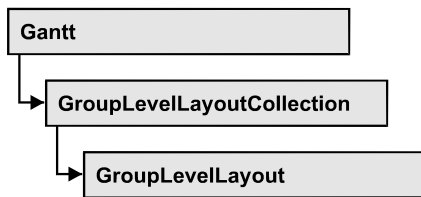
```
Dim groupCltn As VcGroupCollection

groupCltn = VcGantt1.GroupCollection
groupCltn.SelectGroups(VcGroupSelectionType.vcAllGroups)
```

Example Code C#

```
VcGroupCollection groupCltn = vcGantt1.GroupCollection;
groupCltn.SelectGroups(VcGroupSelectionType.vcAllGroups);
```

7.35 VcGroupLevelLayout



An object of the type `VcGroupLevelLayout` defines the content and the appearance of grouping levels. For this, the name of the grouping level, the level number, the grouping field, sorting and sorting order can serve, as well as various options concerning the design of calendar and line grids and of separation lines.

Properties

- `AutoCollapseGroups`
- `AutoExpandTargetGroup`
- `BodiesCollapsed`
- `BodiesCollapsedDataFieldIndex`
- `BodiesCollapsedMapName`
- `CalendarGridName`
- `CalendarGridsVisible`
- `CalendarGridsWithChildGroups`
- `CalendarNameDataFieldIndex`
- `DateLineGridName`
- `DateLineGridsVisible`
- `DateLineGridsWithChildGroups`
- `DateLineName`
- `DateLinesVisible`
- `DateLinesWithChildGroups`
- `GroupDataFieldIndex`
- `GroupNodesVisible`
- `GroupsInvisible`
- `GroupsInvisibleCollapsedMapName`
- `GroupsInvisibleDataFieldIndex`
- `Level`
- `ModificationsAllowed`
- `MovingGroupsVerticallyViaDiagramAllowed`
- `MovingGroupsVerticallyViaTableAllowed`
- `Name`
- `NodesInHeaders`

- NodesOverlaid
- OptimizedNodesSortDataFieldIndex
- OptimizedNodesSortOrder
- OverlaidNodesSortDataFieldIndex
- OverlaidNodesSortOrder
- PagebreakMode
- RestoreAutoCollapsedGroups
- RestoreAutoExpandedGroups
- RowBackColorAsARGB
- RowBackColorDataFieldIndex
- RowBackColorMapName
- RowPattern
- RowPatternColorAsARGB
- RowPatternColorDataFieldIndex
- RowPatternColorMapName
- RowPatternDataFieldIndex
- RowPatternMapName
- SeparationLineColor
- SeparationLineColorDataFieldIndex
- SeparationLineColorMapName
- SeparationLinesVisible
- SeparationLinesVisibleAtTop
- SeparationLineThickness
- SeparationLineType
- SortDataFieldIndex
- SortOrder
- Specification
- SummaryBarsVisible
- Visible

Properties

AutoCollapseGroups

Property of VcGroupLevelLayout

This property lets you set or retrieve whether in the group level layout the groups are to be collapsed automatically on interactions.

	Data Type	Explanation
Property value	System.Boolean	Groups are/are not collapsed automatically on interactions

AutoExpandTargetGroup

Property of VcGroupLevelLayout

This property lets you set or retrieve whether in the group level layout the groups are to be expanded automatically on interactions.

	Data Type	Explanation
Property value	System.Boolean	Target groups are/are not expanded automatically on interactions

BodiesCollapsed

Property of VcGroupLevelLayout

This property lets you set or retrieve, whether the groups of this level are (True) are not (False) collapsed.

	Data Type	Explanation
Property value	System.Boolean	Group collapsed/expanded

BodiesCollapsedDataFieldIndex

Property of VcGroupLevelLayout

This property lets you set or retrieve the data field index for the collapsed bodies of this grouping level. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Int16	This levels groups bodies collapsed data field index

BodiesCollapsedMapName

Property of VcGroupLevelLayout

This property lets you set or retrieve the name of a map for the bodies collapsed on this group level. If set to "" or if the property **Bodies-CollapsedDataFieldIndex** is set to -1, then no map will be used.

	Data Type	Explanation
Property value	System.String	Name of the map for collapsed bodies

CalendarGridName

Property of VcGroupLevelLayout

This property lets you set or retrieve the name of the calendar grid for this group level layout. You can also set this property in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.String	name of the calendar grid

CalendarGridsVisible

Property of VcGroupLevelLayout

This property lets you set or retrieve whether workfree periods are marked by a background color and/or a pattern. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Workfree periods are/are not accentuated

CalendarGridsWithChildGroups

Property of VcGroupLevelLayout

This property lets you set or retrieve whether calendar grids are also displayed for subgroups. You can also set this property in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	calendar grid for subgroups are/are not displayed

CalendarNameDataFieldIndex

Property of VcGroupLevelLayout

This property lets you set or retrieve the index of the data field that holds the name of the calendar to apply to the group level. This property also can be set on the **Calendar Grid** property page.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which contains the name of the calendar

DateLineGridName

Property of VcGroupLevelLayout

This property lets you set or retrieve the name of the date line grid for this group level layout. You can also set this property in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.String	name of the date line grid

DateLineGridsVisible

Property of VcGroupLevelLayout

This property lets you set or retrieve whether a vertical date grid is displayed. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Date grids are/are not displayed.

DateLineGridsWithChildGroups

Read Only Property of VcGroupLevelLayout

This property lets you set or retrieve whether the date line grids are also displayed for subgroups. You can also set this property in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	date line grids for subgroups are/are not displayed

DateLineName

Property of VcGroupLevelLayout

This property lets you set or retrieve the name of the date line for this group level layout. You can also set this property in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.String	Name of the date line

DateLinesVisible

Property of VcGroupLevelLayout

This property lets you set or retrieve whether date lines are to be displayed. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Date lines are/are not displayed.

DateLinesWithChildGroups

Property of VcGroupLevelLayout

This property lets you set or retrieve whether the date lines are to be displayed for all group elements. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Date lines for subgroups are/are not displayed

GroupDataFieldIndex

Read Only Property of VcGroupLevelLayout

This property lets you set or retrieve the data field index used for grouping of this VcGroupLevelLayout object.

	Data Type	Explanation
Property value	System.Int32	index used for grouping of this VcGroupLevelLayout object

GroupNodesVisible

Property of VcGroupLevelLayout

This property lets you set or retrieve whether this level's group nodes are displayed. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	group nodes are/are not visible

GroupsInvisible

Property of VcGroupLevelLayout

This property lets you set or retrieve whether this level's groups are displayed. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	group nodes are/are not visible

GroupsInvisibleCollapsedMapName

Property of VcGroupLevelLayout

This property lets you set or retrieve the name of a map for the for the invisible groups on this group level. If set to "" or if the property **Bodies-CollapsedDataFieldIndex** is set to -1, then no map will be used.

	Data Type	Explanation

GroupsInvisibleDataFieldIndex

Property of VcGroupLevelLayout

This property lets you set or retrieve the data field index for the invisible groups of this grouping level. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Int32	Data field index

Level

Read Only Property of VcGroupLevelLayout

This property lets you enquire the grouping level of this group level layout. At maximum, 25 grouping levels are possible.

	Data Type	Explanation
Property value	System.Int32	Grouping level of the group level layout

ModificationsAllowed

Property of VcGroupLevelLayout

This property lets you specify whether the user can collapse expanded groups of this level and vice versa. The user can collapse/expand groups by double-clicking on the group heading in the table section, by clicking on the minus or plus sign next to the group heading or by the context menu for groups. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Modifications allowed (True)/ not allowed (False)

Example Code VB.NET

```
VcGroupLevelLayout.ModificationsAllowed(0) = False
```

MovingGroupsVerticallyViaDiagramAllowed**Read Only Property of VcGroupLevelLayout**

This property lets you set or retrieve whether groups are allowed to be moved vertically in the diagram. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Vertical group movement in diagram enabled/disabled Default value: True

MovingGroupsVerticallyViaTableAllowed**Read Only Property of VcGroupLevelLayout**

This property lets you set or retrieve whether groups are allowed to be moved vertically in the table. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Vertical group movement in table enabled/disabled Default value: true

Name**Property of VcGroupLevelLayout**

This property lets you retrieve the name of a group level layout.

	Data Type	Explanation
Property value	System.String	Name of the group level

NodesInHeaders

Property of VcGroupLevelLayout

This property lets you specify/enquire whether (True) or not (False) the node objects of the group of this level are positioned the same row.

	Data Type	Explanation
Property value	System.Boolean	All nodes of the group are/are not in the same row

NodesOverlaid

Property of VcGroupLevelLayout

This property lets you specify/enquire whether (False) the node layout on this group level is optimized or if nodes overlap (True).

	Data Type	Explanation
Property value	System.Boolean	The node layout is/is not at its optimum

Example Code VB.NET

```
group.LevelLayout.NodesOverlaid = True
```

Example Code C#

```
group.LevelLayout.NodesOverlaid = true;
```

OptimizedNodesSortDataFieldIndex

Property of VcGroupLevelLayout

This property lets you set or retrieve the index of a data field that contains the sorting criterion (the drawing priority) for the display of several nodes in a single row. Setting this property only makes sense if the property **Nodes-ArrangedOptimized** was set to **True**. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Int32	Index of the data field that holds the sorting criterion

OptimizedNodesSortOrder

Property of VcGroupLevelLayout

This property lets you set or retrieve the sorting direction of the sorting criterion, which was selected by the property **OptimizedNodesSortDataFieldIndex**. Setting this property only makes sense if the property **NodesArrangedOptimized** was set to **True**. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	VcNodesSortingOrder	Direction of the sorting order Default value: vcAscending

OverlaidNodesSortDataFieldIndex

Property of VcGroupLevelLayout

This property lets you set or retrieve the index of a data field that contains the sorting criterion (the drawing priority) for the display of several nodes in a single row. Setting this property only makes sense if the property **NodesArrangedOptimized** was set to **False**. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Int32	Index of the data field that holds the sorting criterion

OverlaidNodesSortOrder

Property of VcGroupLevelLayout

This property lets you set or retrieve the sorting direction of the sorting criterion, which was selected by the property **OverlaidNodesSortDataFieldIndex**. Setting this property only makes sense if the property **NodesArrangedOptimized** was set to **False**. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	VcNodeSortingOrder	Direction of the sorting order Default value: vcAscending
	Possible Values: .vcAscending 1	ascending order

.vcDescending 2	Descending order
-----------------	------------------

PagebreakMode

Read Only Property of VcGroupLevelLayout

This property lets you set or retrieve whether and when page breaks after groups are to be carried out. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	VcPagebreakMode Possible Values: .vcPagebreakAfterEachGroup 1 .vcPagebreakNone 0 .vcPagebreakOnPageFull 2	Page break mode Default value: vcPagebreakNone Pagebreak after each group No pagebreak Pagebreak if following group wouldn't fit on page completely

RestoreAutoCollapsedGroups

Property of VcGroupLevelLayout

This property lets you set or retrieve whether in the group level layout automatically collapsed groups are to be restored automatically on interactions.

	Data Type	Explanation
Property value	System.Boolean	Automatically collapsed groups are/are not restored automatically on interactions

RestoreAutoExpandedGroups

Property of VcGroupLevelLayout

This property lets you set or retrieve whether in the group level layout automatically expanded groups are to be restored automatically on interactions.

	Data Type	Explanation
Property value	System.Boolean	Automatically expanded groups are/are not restored automatically on interactions

RowBackColorAsARGB

Property of VcGroupLevelLayout

This property lets you set or retrieve the background color of the group title row. The default color is white.

	Data Type	Explanation
Property value	System.Drawing.Color	ARGB color values {0...255},{0...255},{0...255},{0...255}

RowBackColorDataFieldIndex

Property of VcGroupLevelLayout

This property lets you set or retrieve the data field index to be used with a color map specified by the property **RowBackColorMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int32	Data field index

RowBackColorMapName

Property of VcGroupLevelLayout




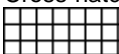





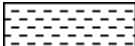
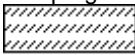
This property lets you set or retrieve the name of a color map (type `vcColorMap`). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **RowBackColorDataFieldIndex**, then the background color is controlled by the map. If no data field entry applies, the background color that is specified in the property **RowBackColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map






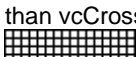
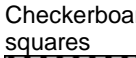

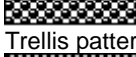
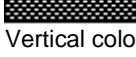


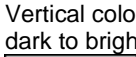






RowPattern

Read Only Property of VcGroupLevelLayout

This property lets you set or retrieve the background pattern of the group title row of this group level.

	Data Type	Explanation
Property value	VcFillPattern	Pattern type
	Possible Values: .vc05PercentPattern... .vc90PercentPattern 01 - 11	Dots in foreground color on background color, the density of the foreground color increasing with the percentage 
	.vcAeroGlassPattern 44	Vertical color gradient in the color of the fill pattern 
	.vcBDiagonalPattern 5	Diagonal lines slanting from bottom left to top right 
	.vcCrossPattern 6	Cross-hatch pattern 
	.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width 
	.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 
	.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width 
	.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 
	.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
	.vcDashedHorizontalPattern 2026	Dashed horizontal lines 
	.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 

.vcDashedVerticalPattern 2027	Dashed vertical lines
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small
.vcDiagonalBrickPattern 2032	Diagonal brick pattern
.vcDivotPattern 2036	Divot pattern
.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines
.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right
.vcHorizontalBrickPattern 2033	Horizontal brick pattern
.vcHorizontalGradientPattern 52	Horizontal color gradient
.vcHorizontalPattern 3	Horizontal lines
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern
.vcLargeConfettiPattern 2029	Confetti pattern, large
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern
.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern
.vcNoPattern 1276	No fill pattern

.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large 
.vcPlaidPattern 2035	Plaid pattern 
.vcShinglePattern 2039	Diagonal shingle pattern 
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern 
.vcSmallConfettiPattern 2028	Confetti pattern 
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern 
.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares 
.vcSpherePattern 2041	Checkerboard of spheres 
.vcTrellisPattern 2040	Trellis pattern 
.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright 
.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark 
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright 
.vcVerticalGradientPattern 62	Vertical color gradient 
.vcVerticalPattern 2	Vertical lines 
.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark 
.vcWavePattern 2031	Horizontal waves pattern 
.vcWeavePattern 2034	Interwoven stripes pattern 
.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern 
.vcWideUpwardDiagonalPattern 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of vcBDiagonalPattern 



RowPatternColorAsARGB

Property of VcGroupLevelLayout

This property lets you set or retrieve the pattern color of the group title row of this group level. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color. When casting an RGB value on an ARGB value, an alpha value of 255 has to be added.

Also see **set/getRowBackColorAsARGB**.

If in the property **RowPatternColorMapName** a map is specified, the map will control the pattern color in dependence of the data.

	Data Type	Explanation
Property value	System.Drawing.Color	ARGB color values {0...255},{0...255},{0...255},{0...255}}

RowPatternColorDataFieldIndex

Property of VcGroupLevelLayout

This property lets you set or retrieve the data field index that has to be specified if the property **RowPatternColorMapName** is used. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int32	Data field index

RowPatternColorMapName

Property of VcGroupLevelLayout

This property lets you set or retrieve the name of a color map (type vcColorMap). If set to "", no map will be used. Only if a map name and a

data field index are specified in the property **RowPatternColorDataFieldIndex**, the pattern color is controlled by the map. If no data field entry applies, the pattern color of the group title row that is specified in the property **RowPatternColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

RowPatternDataFieldIndex

Property of VcGroupLevelLayout

This property lets you set or retrieve the data field index to be used together with the property **RowPatternMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int32	Data field index

RowPatternMapName

Property of VcGroupLevelLayout

This property lets you set or retrieve the name of a pattern map (type vcPatternMap). If set to "", no map will be used. Only if a map name and additionally a data field index are specified in the property **RowPatternDataFieldIndex**, the pattern is controlled by the map. If no data field entry applies, the pattern of the layer that is specified in the property **RowPattern** will be used.

	Data Type	Explanation
Property value	System.String	Name of the pattern map

SeparationLineColor

Property of VcGroupLevelLayout

This property lets you set or retrieve the color of the separation lines of the the grouping levels.

This property also can be set in the **Grouping** dialog, section **Groupwise**, field **Separation Line**.

	Data Type	Explanation
Property value	System.Drawing.Color	Color value ({0...255},{0...255},{0...255})

SeparationLineColorDataFieldIndex

Property of VcGroupLevelLayout

This property lets you set or retrieve the data field index to be used with a map specified by the property **SeparationLineColorMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int32	Data field index

SeparationLineColorMapName

Property of VcGroupLevelLayout

This property lets you set or retrieve the name of a map for the separation line color. If set to "" or if the property **GroupLevelLayoutLineColorDataFieldIndex** is set to <-1, then no map will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

SeparationLinesVisible

Property of VcGroupLevelLayout

This property lets you set or retrieve whether separation lines are to be displayed between grouping levels.

This property also can be set in the **Groupwise** section of the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Separation lines are displayed/not displayed

SeparationLinesVisibleAtTop

Property of VcGroupLevelLayout

This property lets you set or retrieve whether separation lines are to be displayed above groups of different grouping levels (or below).

This property also can be set in the **Groupwise** section of the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Separation lines at top are displayed/not displayed

SeparationLineThickness

Property of VcGroupLevelLayout

This property lets you set or retrieve the line thickness of a separation line between group levels.

If you set this property to values between 1 and 4, an absolute line thickness is defined in pixels. Irrespective of the zoom factor a line will always show the same line thickness in pixels. When printing though, the line thickness is adapted for the sake of legibility and becomes dependent of the zoom factor:

Value	Points	mm
1	1/2 point	0.09 mm
2	1 point	0.18 mm
3	3/2 points	0.26 mm
4	2 points	0.35 mm

A point equals 1/72 inch and represents the unit of the font size.

If you set this property to values between 5 and 1,000, the line thickness is defined in 1/100 mm, so the lines will be displayed in a true thickness in pixels that depends on the zoom factor.

This property also can be set in the **Grouping** dialog, section **Groupwise**, field **Separation Line**.

	Data Type	Explanation
Property value	System.Int32	Line thickness LineType {1...4}: line thickness in pixels LineType {5...1000}: line thickness in 1/100 mm

SeparationLineType

Property of VcGroupLevelLayout

This property lets you specify/enquire the line type of a date line.

This property also can be set in the **Grouping** dialog, section **Groupwise**, field **Separation Line**.

	Data Type	Explanation
Property value	VcLineType	Type of separation lines of hierarchy levels
	Possible Values:	
	.vcDashed 4	Line dashed
	.vcDashed 4	Line dashed
	.vcDashedDotted 5	Line dashed-dotted
	.vcDashedDotted 5	Line dashed-dotted
	.vcDotted 3	Line dotted
	.vcDotted 3	Line dotted
	.vcLineType0 100	Line Type 0

	.vcLineType1 101	Line Type 1

	.vcLineType10 110	Line Type 10
	
	.vcLineType11 111	Line Type 11
	
	.vcLineType12 112	Line Type 12
	
	.vcLineType13 113	Line Type 13
	
	.vcLineType14 114	Line Type 14
	
	.vcLineType15 115	Line Type 15
	
	.vcLineType16 116	Line Type 16
	
	.vcLineType17 117	Line Type 17
	
	.vcLineType18 118	Line Type 18
	
	.vcLineType2 102	Line Type 2
	
	.vcLineType3 103	Line Type 3
	

.vcLineType4 104	Line Type 4 -----
.vcLineType5 105	Line Type 5 -----
.vcLineType6 106	Line Type 6 -----
.vcLineType7 107	Line Type 7 -----
.vcLineType8 108	Line Type 8 -----
.vcLineType9 109	Line Type 9 -----
.vcNone 1	No line type assigned
.vcNone 1	No line type
.vcSolid 2	Line solid
.vcSolid 2	Line solid

SortDataFieldIndex

Property of VcGroupLevelLayout

This property lets you set/retrieve the data field index the groups of this grouping level are sorted by. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Parameter: ⇒ sortlevel	System.Int32	Sorting level
Property value	System.Int32	Index of the data field that holds the sorting criterion

SortOrder

Property of VcGroupLevelLayout

This property lets you specify the sorting order of groups (ascending or descending). The property **SortDataFieldIndex** lets you specify the field the groups are sorted by. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Parameter: ⇒ sortLevel	System.Int32	Sorting level
Property value	VcNodesSortingOrder	Direction of the sorting order Default value: vcAscending

Specification

Read Only Property of VcGroupLevelLayout

This property lets you retrieve the specification of a group level layout. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored without problems to text files or data bases. This allows for persistency. A specification can be used to create a group level layout by the method **VcGroupLevelLayout.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the group level layout

SummaryBarsVisible

Property of VcGroupLevelLayout

This property lets you specify/enquire whether summary bars are be displayed or not.

This property also can be set in the **Groupwise** section of the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	summary bars visible (True)/ invisible (False)

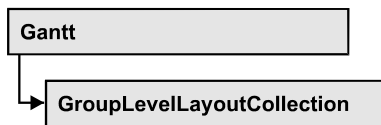
Visible

Property of VcGroupLevelLayout

This property lets you set or retrieve whether (True) or not (False) this group level is visible.

	Data Type	Explanation
Property value	System.Boolean	Group level visible/invisible

7.36 VcGroupLevelLayoutCollection



If nodes were grouped, an object of the type `VcGroupLevelLayoutCollection` contains all available layouts. You can access all objects in an iterative loop by **For Each groupLevelLayout In GroupLevelLayoutCollection** or by the methods **First...** and **Next...**. You can access a single layout using the methods **GroupLevelLayoutByName** and **GroupLevelLayoutIndex**. The number of layouts in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the layouts in the corresponding way.

Properties

- Count

Methods

- Add
- AddBySpecification
- Copy
- FirstGroupLevelLayout
- GetEnumerator
- GroupLevelLayoutByIndex
- GroupLevelLayoutByName
- NextGroupLevelLayout
- Remove
- Update

Properties

Count

Read Only Property of VcGroupLevelLayoutCollection

This property lets you retrieve the number of group level layouts in the `GroupLevelLayoutCollection` object.

	Data Type	Explanation
Property value	System.Int32	Number of group level layouts

Methods

Add

Method of VcGroupLevelLayoutCollection

This method lets you create a group level layout as a member of the GroupLevelLayoutCollection. If the name was not used before, the new group level layout object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ groupLevelLayoutName	System.String	name of group level layout
Return value	VcGroupLevelLayout	New group level layout object

AddBySpecification

Method of VcGroupLevelLayoutCollection

This method lets you create a group level layout by using a group level layout specification. This way of creating allows group level layout objects to become persistent. The specification of a group level layout can be saved and re-loaded (see VcGroupLevelLayout property **Specification**). In a subsequent session the group level layout can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter: ⇒ Specification	System.String	Group level layout specification
Return value	VcGroupLevelLayout	New group level layout object

Copy

Method of VcGroupLevelLayoutCollection

By this method you can copy a group level layout. If the group level layout that is to be copied exists, and if the name for the new group level layout does not yet exist, the new group level layout object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter:		
⇒ groupLevelLayoutName	System.String	Name of the group level layout to be copied
⇒ newGroupLevelLayoutName	System.String	Name of the new group level layout
Return value	VcGroupLevelLayout	Group level layout object

FirstGroupLevelLayout

Method of VcGroupLevelLayoutCollection

This method can be used to access the initial value, i.e. the first group level layout of a group level layout collection and then to continue in a forward iteration loop by the method **NextGroupLevelLayout** for the group level layouts following. If there is no group level layout in the GroupLevelLayoutCollection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcGroupLevelLayout	First group level layout

GetEnumerator

Method of VcGroupLevelLayoutCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the group objects included.

	Data Type	Explanation
Return value	System.Object	Reference object

GroupLevelLayoutByIndex

Method of VcGroupLevelLayoutCollection

This method lets you access a certain group level layout by its index. If a group level layout of the specified index does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the group level layout
Return value	VcGroupLevelLayout	Group level layout object returned

GroupLevelLayoutByName

Method of VcGroupLevelLayoutCollection

This method is used to access a group level layout by its name. If a group level layout of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ groupLevelLayoutName	System.String	Name of the group level layout
Return value	VcGroupLevelLayout	Group level layout

NextGroupLevelLayout

Method of VcGroupLevelLayoutCollection

This method can be used in a forward iteration loop to retrieve subsequent group level layouts from a GroupLevelLayoutCollection after initializing the loop by the method **FirstGroupLevelLayout**. If there is no group level layout left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcGroupLevelLayout	Subsequent group level layout

Remove

Method of VcGroupLevelLayoutCollection

This method lets you delete a group level layouts. If the group level layout is used in another object, it cannot be deleted. Then False will be returned, otherwise True.

	Data Type	Explanation
Parameter: ⇒ groupLevelLayoutName	System.String	Group level layout name
Return value	System.Boolean	Group level layout deleted (True)/not deleted (False)

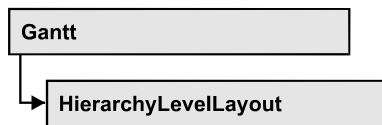
Update

Method of VcGroupLevelLayoutCollection

This method has to be used when group level layout modifications have been carried out. The method **Update** updates all objects that are concerned by the group level layout you have edited. You should call this method at the end of the code that defines the group level layouts and the group level layout collection. Otherwise the update will be processed before all group level layout definitions are processed.

	Data Type	Explanation
Return value	System.Boolean	update successful (True)/ not successful (False)

7.37 VcHierarchyLevelLayout



An object of the type **VcHierarchyLevelLayout** defines the content and the appearance of the hierarchical order of nodes.

Properties

- AutoCollapseGroups
- AutoExpandTargetGroup
- BodiesCollapsed
- BodiesCollapsedDataFieldIndex
- BodiesCollapsedMapName
- HierarchyDataFieldIndex
- LevelMaximumForPagebreaks
- NodeSeparationLinesVisible
- NodesInHeaders
- NodesOverlaid
- PagebreakMode
- RestoreAutoCollapsedGroups
- RestoreAutoExpandedGroups
- SeparationLineColor
- SeparationLinesVisible
- SeparationLineThickness
- SeparationLineType
- SummaryBarsVisible

Properties

AutoCollapseGroups

Property of VcHierarchyLevelLayout

This property lets you set or retrieve whether in the hierarchy level layout the groups are to be collapsed automatically on interactions.

	Data Type	Explanation
Property value	System.Boolean	Groups are/are not collapsed automatically on interactions

AutoExpandTargetGroup

Property of VcHierarchyLevelLayout

This property lets you set or retrieve whether in the hierarchy level layout the groups are to be expanded automatically on interactions.

	Data Type	Explanation
Property value	System.Boolean	Target groups are/are not expanded automatically on interactions

BodiesCollapsed

Property of VcHierarchyLevelLayout

This property lets you set or retrieve, whether (True) or not (False) all groups are collapsed.

	Data Type	Explanation
Property value	System.Boolean	Group collapsed/expanded

BodiesCollapsedDataFieldIndex

Property of VcHierarchyLevelLayout

This property lets you set or retrieve the data field index for the collapsed bodies of this hierarchy level. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Int16	This hierarchy levels groups bodies collapsed data field index

BodiesCollapsedMapName

Property of VcHierarchyLevelLayout

This property lets you set or retrieve the name of a map for the bodies collapsed on this hierarchy level. If set to "" or if the property **BodiesCollapsedDataFieldIndex** is set to -1, then no map will be used.

	Data Type	Explanation
Property value	System.String	Name of the map for collapsed bodies

HierarchyDataFieldIndex

Property of VcHierarchyLevelLayout

This property lets you set/retrieve the data field index used for grouping of this **VcGroupLevelLayout** object

	Data Type	Explanation
Property value	System.Int32	Data field which defines the hierarchical order of activities

LevelMaximumForPagebreaks

Property of VcHierarchyLevelLayout

This property lets you set or retrieve up to which hierarchy level page breaks are to be carried out.

If this property is set to the default -1 the page breaks are carried out on each hierarchy level.

	Data Type	Explanation

NodeSeparationLinesVisible

Property of VcHierarchyLevelLayout

This property lets you set or retrieve whether or not separation lines are to be displayed.

This property can also be set in the **Node** section of the **Grouping** dialog.

	Data Type	Explanation

NodesInHeaders

Property of VcHierarchyLevelLayout

This property lets you specify/enquire whether (True) or not (False) the node objects of the group of this level are positioned in the same row.

	Data Type	Explanation
Property value	System.Boolean	All nodes of the group are/are not in the same row

NodesOverlaid

Property of VcHierarchyLevelLayout

This property lets you set or retrieve whether (False) the node layout on this group level is optimized or if nodes overlap (True).

	Data Type	Explanation
Parameter: ⇐ Rückgabewert	System.Boolean	The node layout is/is not at its optimum
Property value	System.Boolean	The node layout is/is not at its optimum

PagebreakMode

Property of VcHierarchyLevelLayout

This property lets you set or retrieve whether and when page breaks after groups are to be carried out. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	VcPagebreakMode Possible Values: .vcPagebreakAfterEachGroup 1 .vcPagebreakNone 0	Page break mode Default value: vcPagebreakNone Pagebreak after each group No pagebreak

.vcPagebreakOnPageFull 2

Pagebreak if following group wouldn't fit on page completely

RestoreAutoCollapsedGroups

Property of VcHierarchyLevelLayout

This property lets you set or retrieve whether in the hierarchy level layout automatically collapsed groups are to be restored automatically on interactions.

	Data Type	Explanation
Property value	System.Boolean	Automatically collapsed groups are/are not restored automatically on interactions

RestoreAutoExpandedGroups

Property of VcHierarchyLevelLayout

This property lets you set or retrieve whether in the hierarchy level layout automatically expanded groups are to be restored automatically on interactions.

	Data Type	Explanation
Property value	System.Boolean	Automatically expanded groups are/are not restored automatically on interactions

SeparationLineColor

Property of VcHierarchyLevelLayout

This property lets you set or retrieve the color of the separation lines of the the hierarchy levels.

This property also can be set in the **Hierarchy** section of the **Grouping** dialog by clicking on  next to **Separation Line**.

	Data Type	Explanation
Property value	System.Drawing.Color	Color value ({0...255},{0...255},{0...255})

SeparationLinesVisible

Property of VcHierarchyLevelLayout

This property lets you set or retrieve whether separation lines are to be displayed between hierarchy levels.

This property also can be set in the **Hierarchy** section of the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Separation lines are displayed/not displayed

SeparationLineThickness

Property of VcHierarchyLevelLayout

This property lets you set or retrieve the line thickness of a separation line between hierarchy levels.

If you set this property to values between 1 and 4, an absolute line thickness is defined in pixels. Irrespective of the zoom factor a line will always show the same line thickness in pixels. When printing though, the line thickness is adapted for the sake of legibility and becomes dependent of the zoom factor:

Value	Points	mm
1	1/2 point	0.09 mm
2	1 point	0.18 mm
3	3/2 points	0.26 mm
4	2 points	0.35 mm

A point equals 1/72 inch and represents the unit of the font size.

If you set this property to values between 5 and 1,000, the line thickness is defined in 1/100 mm, so the lines will be displayed in a true thickness in pixels that depends on the zoom factor.

This property also can be set in the **Hierarchy** section of the **Grouping** dialog by clicking on  next to **Separation Line**.

	Data Type	Explanation
Property value	System.Int32	Line thickness LineType {1...4}: line thickness in pixels LineType {5...1000}: line thickness in 1/100 mm

SeparationLineType

Property of VcHierarchyLevelLayout

This property lets you specify/enquire the line type of a date line.

This property also can be set in the **Hierarchy** section of the **Grouping** dialog by clicking on  next to **Separation Line**.

	Data Type	Explanation
Property value	LineTypeEnum	Type of separation lines of hierarchy levels

SummaryBarsVisible

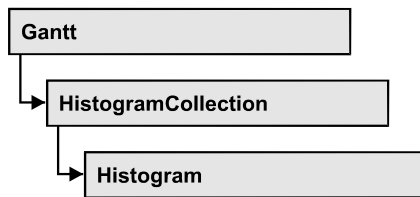
Property of VcHierarchyLevelLayout

This property lets you specify/enquire whether summary bars are be displayed or not.

This property also can be set in the **Hierarchy** section of the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	summary bars visible (True)/ invisible (False)

7.38 VcHistogram



An object of the type **VcHistogram** is an element of the object **VcHistogramCollection** and is designed to contain capacity curves referring to the values of the Gantt diagram located above it. You can define a scale and create curves, that can obtain its data from different sources.

Properties

- CalendarGridsVisible
- CalendarName
- CurveCollection
- Name
- NominalScaleMaximum
- NominalScaleMinimum
- NumericScaleCollection
- RowBackColorAsARGB
- RowPattern
- RowPatternColorAsARGB
- Visible

Methods

- FitRangeIntoView
- GetActualScaleValues
- GetCurrentYValues
- PutInOrderAfter
- ScrollToValue

Properties

CalendarGridsVisible

Property of VcHistogram

This property lets you set or retrieve whether workfree periods are marked by a background color and/or a pattern. This property also can be set in the **Administrate Histograms** dialog.

	Data Type	Explanation
Property value	System.Boolean	Workfree periods are/are not accentuated

CalendarName

Read Only Property of VcHistogram

This property lets you assign a calendar to the histogram. The calendar holds the time pattern to be displayed by the grid. The calendar is to be specified by its name.

	Data Type	Explanation
Property value	System.String	Character string that passes the calendar name

CurveCollection

Read Only Property of VcHistogram

This property gives access to the curve collection object, that is, to the curves that it contains.

	Data Type	Explanation
Property value	VcCurveCollection	CurveCollection object

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim histogram As VcHistogram
Dim curveCltn As VcCurveCollection

histogramCltn = VcGantt1.HistogramCollection
histogram = histogramCltn.HistogramByName("HISTOGRAM_1")
curveCltn = histogram.CurveCollection
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcHistogram histogram = histogramCltn.HistogramByName("HISTOGRAM_1");
VcCurveCollection curveCltn = histogram.CurveCollection;
```

Name**Read Only Property of VcHistogram**

This property lets you retrieve the name of a histogram curve.

	Data Type	Explanation
Property value	System.String	Name of the histogram

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim histogram As VcHistogram

histogramCltn = VcGantt1.HistogramCollection
histogram = histogramCltn.Active
MsgBox(histogram.Name)
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcHistogram histogram = histogramCltn.Active;
MessageBox.Show(histogram.Name);
```

NominalScaleMaximum**Property of VcHistogram**

This property lets you set the maximum value of the numeric scale of the histogram. If the y values of the histogram curves exceed the maximum value set, the numeric scale will be adapted to the y values of the curves.

	Data Type	Explanation
Property value	System.Int32	Maximum y value

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim histogram As VcHistogram

histogramCltn = VcGantt1.HistogramCollection
histogram = histogramCltn.HistogramByName("HISTOGRAM_1")
histogram.NominalScaleMaximum (20)
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcHistogram histogram = histogramCltn.HistogramByName("HISTOGRAM_1");
histogram.NominalScaleMaximum(20);
```

NominalScaleMinimum

Property of VcHistogram

This property lets you specify a minimum value of the numeric scale of the histogram.

	Data Type	Explanation
Property value	System.Int32	Minimum y

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim histogram As VcHistogram

histogramCltn = VcGantt1.HistogramCollection
histogram = histogramCltn.HistogramByName("HISTOGRAM_1")
histogram.NominalScaleMinimum (2)
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcHistogram histogram = histogramCltn.HistogramByName("HISTOGRAM_1");
histogram.NominalScaleMinimum(2);
```

NumericScaleCollection

Read Only Property of VcHistogram

This property lets you access the NumericScaleCollection object, that contains all numeric scales available.

	Data Type	Explanation
Property value	VcNumericScaleCollection	NumericScaleCollection object

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim histogram As VcHistogram
Dim numericScaleCltn As VcNumericScaleCollection

histogramCltn = VcGantt1.HistogramCollection
histogram = histogramCltn.HistogramByName("HISTOGRAM_1")
numericScaleCltn = histogram.NumericScaleCollection
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcHistogram histogram = histogramCltn.HistogramByName("HISTOGRAM_1");
VcNumericScaleCollection numericScaleCltn = histogram.NumericScaleCollection;
```

RowBackColorAsARGB

Read Only Property of VcHistogram

This property lets you set or retrieve the background color of the histogram. This property also can be set in the **Administrate Histograms** property page.

	Data Type	Explanation
Property value	System.Drawing.Color	ARGB color values ({0...255},{0...255},{0...255},{0...255})







Example Code VB.NET



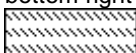
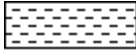
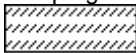



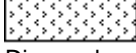

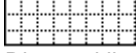





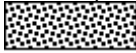

```
VcHistogram.RowBackColor = RGB(255, 0, 0)
```






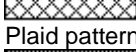


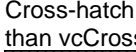
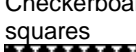

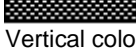
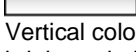
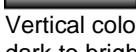




RowPattern

Property of VcHistogram

This property lets you set or retrieve the background pattern of the histogram.

	Data Type	Explanation
Property value	VcFillPattern Possible Values: .vc05PercentPattern... vc90PercentPattern 01 - 11 .vcAeroGlassPattern 44 .vcBDiagonalPattern 5 .vcCrossPattern 6 .vcDarkDownwardDiagonalPattern 2014 .vcDarkHorizontalPattern 2023	Pattern type Dots in foreground color on background color, the density of the foreground color increasing with the percentage  Vertical color gradient in the color of the fill pattern  Diagonal lines slanting from bottom left to top right  Cross-hatch pattern  Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width  Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 

.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width 
.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 
.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
.vcDashedHorizontalPattern 2026	Dashed horizontal lines 
.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 
.vcDashedVerticalPattern 2027	Dashed vertical lines 
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small 
.vcDiagonalBrickPattern 2032	Diagonal brick pattern 
.vcDivotPattern 2036	Divot pattern 
.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines 
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines 
.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right 
.vcHorizontalBrickPattern 2033	Horizontal brick pattern 
.vcHorizontalGradientPattern 52	Horizontal color gradient 
.vcHorizontalPattern 3	Horizontal lines 
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern 
.vcLargeConfettiPattern 2029	Confetti pattern, large 
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern 

.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern 
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern 
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern 
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern 
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern 
.vcNoPattern 1276	No fill pattern
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large 
.vcPlaidPattern 2035	Plaid pattern 
.vcShinglePattern 2039	Diagonal shingle pattern 
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern 
.vcSmallConfettiPattern 2028	Confetti pattern 
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern 
.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares 
.vcSpherePattern 2041	Checkerboard of spheres 
.vcTrellisPattern 2040	Trellis pattern 
.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright 
.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark 
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright 
.vcVerticalGradientPattern 62	Vertical color gradient 
.vcVerticalPattern 2	Vertical lines

.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark
.vcWavePattern 2031	Horizontal waves pattern
.vcWeavePattern 2034	Interwoven stripes pattern
.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern
.vcWideUpwardDiagonalPattern 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of vcBDiagonalPattern
.vcZigZagPattern 2030	Horizontal zig-zag lines

RowPatternColorAsARGB

Property of VcHistogram

This property lets you set or retrieve the pattern color of the histogram Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color. When casting an RGB value on an ARGB value, an alpha value of 255 has to be added.

Note:> The ribbon background of the numeric scale has to be transparent for the background to become visible.

	Data Type	Explanation
Property value	System.Drawing.Color	ARGB color values {{0...255},{0...255},{0...255},{0...255}}

Visible

Property of VcHistogram

This property lets you set or retrieve whether the histogram is visible.

	Data Type	Explanation
Property value	System.Boolean	Histogram visible (True)/ not visible (False)

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim histogram As VcHistogram

histogramCltn = VcGantt1.HistogramCollection
histogram = histogramCltn.Active
histogram.Visible = True
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcHistogram histogram = histogramCltn.Active;
histogram.Visible = true;
```

Methods

FitRangeIntoView

Method of VcHistogram

This method lets you match a section of the numeric scale into a window for display. The graduation will change correspondingly. The beginning and the end are set by the startValue and endValue parameters, respectively. The parameter gapAsNoOfTimeUnits is not used. To derive appropriate section limits from existing curves, see GetCurrentYValues(...).

To match histograms in a window please see **VcGantt.FitHistogramsIntoView**

	Data Type	Explanation
Parameter:		
⇒ startValue	System.Int32	Start value of the section to be matched
⇒ endValue	System.Int32	End value of the section to be matched
⇒ gapAsNoOfTimeUnits	System.Int32	Parameter is not used
Return value	System.Boolean	Area could/could not be matched.

GetActualScaleValues

Method of VcHistogram

This method lets you retrieve the actual minimum and maximum values of the histogram's numeric scale.

	Data Type	Explanation
Parameter:		
↩ minimumValue	System.Int32	Minimum actual value of the numeric scale
↩ maximumValue	System.Int32	Maximum actual value of the numeric scale
Return value	System.Boolean	High-low values could (True) / could not (False) be successfully retrieved.

GetCurrentYValues

Method of VcHistogram

This method lets you retrieve the minimum and maximum Y-value of all curves in the histogram. The result can contribute to defining the section of the numeric scale to be displayed (s. **FitRangeIntoView**).

	Data Type	Explanation
Parameter:		
↩ minValue	System.Int32	Minimum Y-value of all curves
↩ maxValue	System.Int32	Maximum Y-value of all curves
Return value	System.Boolean	High-low values could (True) / could not (False) be successfully retrieved.

PutInOrderAfter

Method of VcHistogram

This method lets you set the histogram behind a histogram specified by name, within the HistogramCollection. If you set the name to "", the histogram will be put in the first position. The order of the histograms determines the order by which they are displayed.

	Data Type	Explanation
Parameter:		
⇒ refName	System.String	Name of the histogram behind which the current histogram is to be put.

Return value	Void	
---------------------	------	--

Example Code VB.NET

```
Dim histgrCltn As VcHistogramCollection
Dim histgr1 As VcHistogram
Dim histgr2 As VcHistogram

histgrCltn = VcGantt1.HistogramCollection()
histgr1 = histgrCltn.Add("histgr1")
histgr2 = histgrCltn.Add("histgr2")
histgr1.PutInOrderAfter("histgr2")
histgrCltn.Update()
```

Example Code C#

```
VcHistogramCollection histgrCltn = vcGantt1.HistogramCollection;
VcHistogram histgr1 = histgrCltn.Add("histgr1");
VcHistogram histgr2 = histgrCltn.Add("histgr2");
histgr1.PutInOrderAfter("histgr2");
histgrCltn.Update();
```

ScrollToValue

Method of VcHistogram

This method allows you to scroll to a defined y value in the histogram and to specify whether that value should be displayed at the top, in the center or at the bottom of the screen.

	Data Type	Explanation
Parameter:		
⇒ value	System.Int32	Y value to be scrolled to
⇒ verAlignment	VcVerticalAlignment	Vertical alignment
	Possible Values: .vcBottomAligned 2 .vcTopAligned 1 .vcVerCenterAligned - 1	bottom aligned top aligned vertically centered
Return value	System.Boolean	Scrolling was/was not performed successfully.

Example Code VB.NET

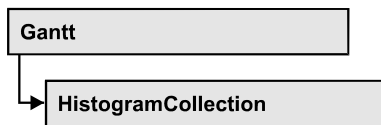
```
Dim histogramCltn As VcHistogramCollection
Dim histogram As VcHistogram

histogramCltn = VcGantt1.HistogramCollection
histogram = histogramCltn.HistogramByName("HISTOGRAM_1")
histogram.ScrollToValue(7, VcVerticalAlignment.vcVerCenterAligned)
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcHistogram histogram = histogramCltn.HistogramByName("HISTOGRAM_1");
histogram.ScrollToValue(7, VcVerticalAlignment.vcVerCenterAligned);
```

7.39 VcHistogramCollection



An object of the type `VcHistogramCollection` automatically contains all available histograms. You can access all objects in an iterative loop by **For Each histogram In HistogramCollection** or by the methods **First...** and **Next...**. You can access a single histogram using the method **HistogramByName**. The number of groups in the collection object can be retrieved by the property **Count**.

Properties

- Active
- Count

Methods

- CreateHistogram
- Delete
- FirstHistogram
- GetEnumerator
- HistogramByIndex
- HistogramByName
- NextHistogram

Properties

Active

Property of VcHistogramCollection

This property lets you set or retrieve the name of the histogram currently used.

The active histogram may be `NOTHING`, if user actions did not take place yet in the histogram area. A histogram can be activated for example by marking a curve.

	Data Type	Explanation
Property value	VcHistogram	Currently used histogram

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim histogram As VcHistogram

histogramCltn = VcGantt1.HistogramCollection
histogram = histogramCltn.Active
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcHistogram histogram = histogramCltn.Active;
```

Count

Read Only Property of VcHistogramCollection

This property lets you retrieve the number of histograms in the HistogramCollection object.

	Data Type	Explanation
Property value	System.Int32	Number of histograms

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim numberOfHistograms As Integer

histogramCltn = VcGantt1.HistogramCollection
numberOfHistograms = histogramCltn.Count
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
int numberOfHistograms = histogramCltn.Count;
```

Methods

CreateHistogram

Method of VcHistogramCollection

By this method you can create a histogram object, which automatically is a member of the HistogramCollection object. The histogram is a copy of the one previously created and therefore contains the same curves.

1034 API Reference: VcHistogramCollection

	Data Type	Explanation
Parameter: ⇒ histogramName	System.String	Name of the histogram to be created
Return value	VcHistogram	Histogram created

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim newHistogram As VcHistogram

histogramCltn = VcGantt1.HistogramCollection
newHistogram = histogramCltn.CreateHistogram("resourceHistogram")
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcHistogram newHistogram = histogramCltn.CreateHistogram("resourceHistogram");
```

Delete

Method of VcHistogramCollection

By this method you can delete a histogram.

	Data Type	Explanation

FirstHistogram

Method of VcHistogramCollection

This method can be used to access the initial value, i.e. the first histogram of a histogram collection, and then to continue in a forward iteration loop by the method **NextHistogram** for the histograms following. If there is no histogram in the histogram collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcHistogram	First histogram

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim histogram As VcHistogram

histogramCltn = VcGantt1.HistogramCollection
histogram = histogramCltn.FirstHistogram
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcHistogram histogram = histogramCltn.FirstHistogram();
```

GetEnumerator**Method of VcHistogramCollection**

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the histogram objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

HistogramByIndex**Method of VcHistogramCollection**

This method lets you access a histogram by its index. If a histogram does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the histogram
Return value	VcHistogram	Histogram object returned

HistogramByName**Method of VcHistogramCollection**

By this method you can retrieve a histogram by its name. If there is no histogram of this name, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ histogramName	System.String	Name of the histogram
Return value	VcHistogram	Histogram

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim histogram As VcHistogram

histogramCltn = VcGantt1.HistogramCollection
histogram = histogramCltn.HistogramByName("Histogram_2")
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcHistogram histogram = histogramCltn.HistogramByName("Histogram_2");
```

NextHistogram**Method of VcHistogramCollection**

This method can be used in a forward iteration loop to retrieve subsequent histograms from a histogram collection after initializing the loop by the method **FirstHistogram**. If there is no histogram left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcHistogram	Succeeding histogram

Example Code VB.NET

```
Dim histogramCltn As VcHistogramCollection
Dim histogram As VcHistogram

histogramCltn = VcGantt1.HistogramCollection
histogram = histogramCltn.FirstHistogram

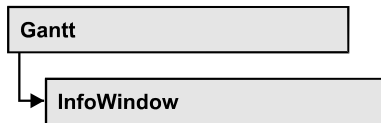
While Not histogram Is Nothing
    ListBox1.Items.Add(histogram.Name)
    histogram = histogramCltn.NextHistogram
End While
```

Example Code C#

```
VcHistogramCollection histogramCltn = vcGantt1.HistogramCollection;
VcHistogram histogram = histogramCltn.FirstHistogram();

while (histogram != null)
{
    listBox1.Items.Add(histogram.Name);
    histogram = histogramCltn.NextHistogram();
}
```

7.40 VcInfoWindow



An object of the type `VcInfoWindow` designates the information window of a node appearing in a Gantt chart when a node is created or modified.

Properties

- `OutputFormatForCenterDate`
- `OutputFormatForDuration`
- `OutputFormatForEndDate`
- `OutputFormatForStartDate`
- `ReferenceDate`
- `UseReferenceDate`
- `Visible`

Properties

OutputFormatForCenterDate

Property of `VcInfoWindow`

This property lets you set or retrieve the output format of a layer's center date (e.g. of a symbol layer) in information windows of nodes. To compose the date you can use the below codes:

- D: first letter of the day of the week (not adjustable)
- TD: Day of the Week (adjustable by using the event **VcTextEntrySupplying**)
- DD: two-digit figure for the day of the month: 01-31
- DDD: first three letters of the day of the week (not adjustable)
- M: first letter of the name of the month (not adjustable)
- TM: name of the month (adjustable by using the event **VcTextEntrySupplying**)
- MM: two-digit figure for the month: 01-12

MMM:	first three letters of the name of the month (not adjustable)
YY:	two-digit figure for the year
YYYY:	four-digit figure for the year
WW:	two-digit figure for the number of the calendar week: 01-53
TW:	text for "calendar week" (adjustable by using the event VcTextEntrySupplying)
Q:	one-digit figure for the quarter: 1-4
TQ:	name of quarter (adjustable by using the event VcTextEntrySupplying)
hh:	two-digit figure for the hour in 24 hours format: 00-23
HH:	two-digit figure for the hour in 12 hours format: 01-12
Th:	Text of "o' clock" (adjustable by using the event VcTextEntrySupplying)
TH:	"am" or "pm" (adjustable by using the event VcTextEntrySupplying)
mm	two-digit figure for the minute: 00-59
ss:	two-digit figure for the second: 00-59
TS:	short date format, as defined in the regional settings of the windows control panel
TL:	long date format, as defined in the regional settings of the windows control panel
TT:	time format, as defined in the regional settings of the windows control panel

Note: Characters which are not to be interpreted as part of the date should be preceded by a backslash '\'. '\\' for instance results in '\'. The special characters: ':, /, -' and **blank** don't need '\' as prefix.

This setting is valid for the table area and for layer annotations in the node area. This property also can be set on the **General** property page.

	Data Type	Explanation
Property value	System.String	String that holds the code of the format to be used; if an empty string is passed, the output format of the Gantt object will be used (see VcGantt.DateOutputFormat).

OutputFormatForDuration

Property of VcInfoWindow

This property lets you set or retrieve the output format of the duration in information windows of nodes. To compose the date you can use the below codes:

hh: two-digit figure for the hour in 24 hours format: 00-23

mm two-digit figure for the minute: 00-59

ss: two-digit figure for the second: 00-59

xC/XC: *The usage of this format requires a special setting in the .ini file. Please contact NETRONIC if necessary.* You can set a maximum ten-place, simple upward counting, for example "07:16:00", which equals 7 hours, 16 minutes, 0 seconds. The notation is: **xC22:C11:C00**. In written language: Show at least 2 digits for the counters 2...0. The separators are variable and can be replaced by other separators symbols. "x" means: Display a preceding "-" symbol if the value is negative, but no "+" symbol if it is positive. "X" means: Display a preceding "-" symbol if the value is negative and a "+" symbol for positive values.

Note: Characters which are not to be interpreted as part of the date should be preceded by a backslash '\'. '\\' for instance results in '\'. The special characters: ':, /, -' and **blank** don't need '\' as prefix.

This setting is valid for the table area and for layer annotations in the node area. This property also can be set on the **General** property page.

	Data Type	Explanation
Property value	System.String	String that holds the code of the format to be used; if an empty string is passed, the output format of the Gantt object will be used (see VcGantt.DateOutputFormat).

OutputFormatForEndDate

Property of VcInfoWindow

This property lets you set or retrieve the output format of a layer's end date in information windows of nodes. To compose the date you can use the below codes:

- D: first letter of the day of the week (not adjustable)
- TD: Day of the Week (adjustable by using the event **VcTextEntrySupplying**)
- DD: two-digit figure for the day of the month: 01-31
- DDD: first three letters of the day of the week (not adjustable)
- M: first letter of the name of the month (not adjustable)
- TM: name of the month (adjustable by using the event **VcTextEntrySupplying**)
- MM: two-digit figure for the month: 01-12
- MMM: first three letters of the name of the month (not adjustable)
- YY: two-digit figure for the year
- YYYY: four-digit figure for the year
- WW: two-digit figure for the number of the calendar week: 01-53
- TW: text for "calendar week" (adjustable by using the event **VcTextEntrySupplying**)
- Q: one-digit figure for the quarter: 1-4
- TQ: name of quarter (adjustable by using the event **VcTextEntrySupplying**)
- hh: two-digit figure for the hour in 24 hours format: 00-23
- HH: two-digit figure for the hour in 12 hours format: 01-12
- Th: Text of "o' clock" (adjustable by using the event **VcTextEntrySupplying**)
- TH: "am" or "pm" (adjustable by using the event **VcTextEntrySupplying**)
- mm two-digit figure for the minute: 00-59
- ss: two-digit figure for the second: 00-59
- TS: short date format, as defined in the regional settings of the windows control panel

- TL: long date format, as defined in the regional settings of the windows control panel
- TT: time format, as defined in the regional settings of the windows control panel

Note: Characters which are not to be interpreted as part of the date should be preceded by a backslash '\'. '\\' for instance results in '\'. The special characters: ':, /, -' and **blank** don't need '\' as prefix.

This setting is valid for the table area and for layer annotations in the node area. This property also can be set on the **General** property page.

	Data Type	Explanation
Property value	System.String	String that holds the code of the format to be used; if an empty string is passed, the output format of the Gantt object will be used (see VcGantt.DateOutputFormat).

OutputFormatForStartDate

Property of VcInfoWindow

This property lets you set or retrieve the output format of a layer's start date in the information windows of nodes. To compose the date you can use the below codes:

- D: first letter of the day of the week (not adjustable)
- TD: Day of the Week (adjustable by using the event **VcTextEntrySupplying**)
- DD: two-digit figure for the day of the month: 01-31
- DDD: first three letters of the day of the week (not adjustable)
- M: first letter of the name of the month (not adjustable)
- TM: name of the month (adjustable by using the event **VcTextEntrySupplying**)
- MM: two-digit figure for the month: 01-12
- MMM: first three letters of the name of the month (not adjustable)
- YY: two-digit figure for the year

YYYY:	four-digit figure for the year
WW:	two-digit figure for the number of the calendar week: 01-53
TW:	text for "calendar week" (adjustable by using the event VcTextEntrySupplying)
Q:	one-digit figure for the quarter: 1-4
TQ:	name of quarter (adjustable by using the event VcTextEntrySupplying)
hh:	two-digit figure for the hour in 24 hours format: 00-23
HH:	two-digit figure for the hour in 12 hours format: 01-12
Th:	Text of "o' clock" (adjustable by using the event VcTextEntrySupplying)
TH:	"am" or "pm" (adjustable by using the event VcTextEntrySupplying)
mm	two-digit figure for the minute: 00-59
ss:	two-digit figure for the second: 00-59
TS:	short date format, as defined in the regional settings of the windows control panel
TL:	long date format, as defined in the regional settings of the windows control panel
TT:	time format, as defined in the regional settings of the windows control panel

Note: Characters which are not to be interpreted as part of the date should be preceded by a backslash '\'. '\\' for instance results in '\'. The special characters: ':, /, -' and **blank** don't need '\' as prefix.

This setting is valid for the table area and for layer annotations in the node area. This property also can be set on the **General** property page.

	Data Type	Explanation
Property value	System.String	String that holds the code of the format to be used; if an empty string is passed, the output format of the Gantt object will be used (see VcGantt.DateOutputFormat).

ReferenceDate

Property of VcInfoWindow

This property lets you set or retrieve the reference date.

	Data Type	Explanation
Property value	System.DateTime	Reference date

UseReferenceDate

Property of VcInfoWindow

This property lets you set or retrieve whether the information window uses a reference date.

	Data Type	Explanation
Property value	System.Boolean	The information window uses (True) / does not use (False) reference date Default value: False

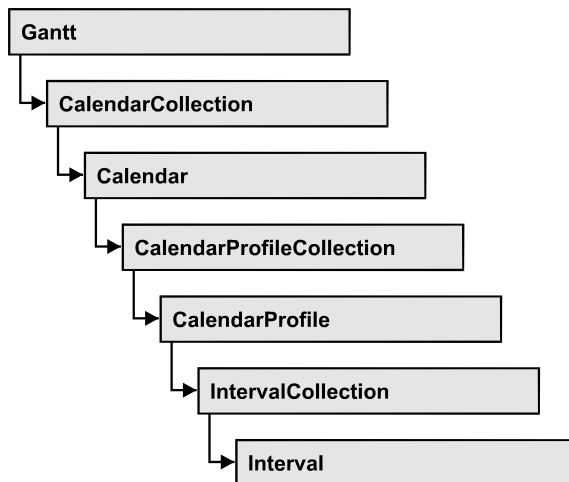
Visible

Property of VcInfoWindow

This property lets you set or retrieve whether the information window is visible.

	Data Type	Explanation
Property value	System.Boolean	Information window visible/invisible Default value: True

7.41 VcInterval



An object of the type **VcInterval** offers the possibility of defining time intervals that are interpreted as working or non-working time. The distinction between the two characteristics is made by the special settings <**WORK**> and <**NONWORK**> of the property **CalendarProfileName**. An interval may refer to other already defined calendar profiles by its property **CalendarProfileName**.

According to the current interval type (**vcCalendarInterval**, **vcDayProfileInterval**, **vcWeekProfileInterval**, **vcYearProfileInterval** oder **vcShiftProfileInterval**) which is not set explicitly but derives from the context of use, only certain properties of the object take effect.

The following table lists the interval types and their corresponding properties:

vcCalendar-Interval	vcYearProfile-Interval	vcWeekProfile-Interval	vcDayProfile-Interval	vcShift-Interval
StartDateTime	StartMonth	StartWeekday	StartTime	Duration
EndDateTime	EndMonth	EndWeekday	EndTime	TimeUnit
	DayInEndMonth			
	DayInStartMonth			

A **CalendarInterval** designates a non-recurring time span within a precisely defined period. Example: 5/5/2010 11:30 to 9/15/2010 5:00.

A **YearProfileInterval** allows to define a yearly recurring day or time span. Example: 5/1 or 12/24 to 12/26.

A **WeekProfileInterval** applies to single or several days in succession of a week. Example: Saturday or Monday to Friday.

A **DayProfileInterval** specifies certain time spans during a day. Example: 8:00 to 5.00

A **ShiftProfile** designates a time span within the specified unit **vcDay**, **vcHours**, **vcMinute** or **vcSeconds** without referring to a date. Example: 4 hours.

Properties

- BackgroundColor
- CalendarProfileName
- DayInEndMonth
- DayInStartMonth
- Duration
- EndDateTime
- EndMonth
- EndTime
- EndWeekday
- LineColor
- LineThickness
- LineType
- Name
- Pattern
- PatternColor
- Specification
- StartDateTime
- StartMonth
- StartTime
- StartWeekday
- Text
- TimeUnit
- Type
- UseGraphicalAttributes

Methods

- PutInOrderAfter

Properties

BackgroundColor

Property of VcInterval

This property lets you set or retrieve the background color of the interval's calendar grid. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color. When casting an RGB value on an ARGB value, an alpha value of 255 has to be added.

The background color can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	System.DrawingColor	ARGB color values {0...255},{0...255},{0...255},{0...255}) Default value: &hFFD8D8D8 (gray)

CalendarProfileName

Property of VcInterval

This property lets you assign a calendar profile to the interval or retrieve the one currently used. This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	System.String	Name of the calendar profile

DayInEndMonth

Property of VcInterval

This property returns or sets the day in the end month of this interval object (for profiles of the type **vcYearProfile** only). This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	System.Int16	Day of last month

DayInStartMonth

Property of VcInterval

This property returns or sets the day in the start month of this interval object (for profiles of the type **vcYearProfile** only). This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	System.Int16	Day of first month

Duration

Property of VcInterval

This property lets you set or retrieve the duration for the interval *only for calendar profiles of the type **vcShiftProfile***. The duration can also be set in the **Edit Shift Calendar** dialog. This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	System.Int32	Last weekday of interval

EndDateTime

Property of VcInterval

This property returns or sets the end date and time of this interval object (for profiles of the type **vcCalendar** only). This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	System.DateTime	End date and time of interval

EndMonth

Property of VcInterval

This property returns or sets the end month of this interval object (for profiles of the type **vcYearProfile** only). This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	VcMonth	End month of interval
	Possible Values: .vcApril 4 .vcAugust 8 .vcDecember 12 .vcFebruary 2 .vcJanuary 1 .vcJuly 7 .vcJune 6 .vcMarch 3 .vcMay 5 .vcNovember 11 .vcOktober 10 .vcSeptember 9	April August December February Januar July une March May November October September

EndTime

Property of VcInterval

This property returns or sets the end time of this interval object (for profiles of the type **vcDayProfile** only). This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	System.DateTime	End time of interval

EndWeekday

Property of VcInterval

This property returns or sets the last weekday of this interval object (for profiles of the type **vcWeekProfile** only). This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	VcWeekday	Last weekday of interval
	Possible Values:	

.vcFriday 5	Week day Friday
.vcMonday 1	Week day Monday
.vcSaturday 6	Week day Saturday
.vcSunday 7	Week day Sunday
.vcThursday 4	Week day Thursday
.vcTuesday 2	Week day Tuesday
.vcWednesday 3	Week day Wednesday

LineColor

Read Only Property of VcInterval

This property lets you set or retrieve the line color of an interval and can also be set in the **Administrate Intervals** dialog. This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {(0...255},{0...255},{0...255)}

LineThickness

Read Only Property of VcInterval

This property lets you set or retrieve the line thickness of the interval's calendar grid lines.

If you set this property to values between 1 and 4, an absolute line thickness is defined in pixels. Irrespective of the zoom factor a line will always show the same line thickness in pixels. When printing though, the line thickness is adapted for the sake of legibility and becomes dependent of the zoom factor:

Value	Points	mm
1	1/2 point	0.09 mm
2	1 point	0.18 mm
3	3/2 points	0.26 mm
4	2 points	0.35 mm

A point equals 1/72 inch and represents the unit of the font size.

If you set this property to values between 5 and 1,000, the line thickness is defined in 1/100 mm, so the lines will be displayed in a true thickness in pixels that depends on the zoom factor.

This property also can be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	System.Int16	Line thickness LineType {1...4}: line thickness in pixels LineType {5...1000}: line thickness in 1/100 mm Default value: As defined in the dialog

LineType

Read Only Property of VcInterval

This property lets you set or retrieve the line type of the interval's calendar grid. It can also be set in the **Administrate Intervals** dialog. This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	VcLineType	Line type ({0...255},{0...255},{0...255})
	Possible Values: .vcDashed 4 .vcDashed 4 .vcDashedDotted 5 .vcDashedDotted 5 .vcDotted 3 .vcDotted 3 .vcLineType0 100 .vcLineType1 101 .vcLineType10 110 .vcLineType11 111 .vcLineType12 112 .vcLineType13 113 .vcLineType14 114 .vcLineType15 115 .vcLineType16 116	Line dashed Line dashed Line dashed-dotted Line dashed-dotted Line dotted Line dotted Line Type 0 Line Type 1 Line Type 10 Line Type 11 Line Type 12 Line Type 13 Line Type 14 Line Type 15 Line Type 16

.vcLineType17 117	Line Type 17 -----
.vcLineType18 118	Line Type 18 -----
.vcLineType2 102	Line Type 2 -----
.vcLineType3 103	Line Type 3 -----
.vcLineType4 104	Line Type 4 -----
.vcLineType5 105	Line Type 5 -----
.vcLineType6 106	Line Type 6 -----
.vcLineType7 107	Line Type 7 -----
.vcLineType8 108	Line Type 8 -----
.vcLineType9 109	Line Type 9 -----
.vcNone 1	No line type assigned
.vcNone 1	No line type
.vcSolid 2	Line solid
.vcSolid 2	Line solid

Name

Read Only Property of VcInterval

This property lets you retrieve the name of an interval. This feature can also be set in the **Administrate Intervals** dialog.









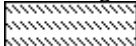

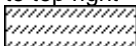




	Data Type	Explanation
Property value	System.String	Name of the interval


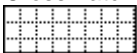
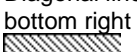
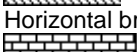
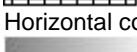
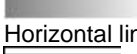
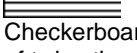


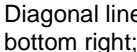

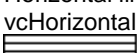
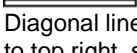

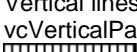
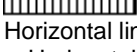


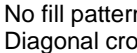
Pattern

Property of VcInterval

This property lets you set or retrieve the pattern of the interval's calendar grid. The pattern can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	VcFillPattern	Pattern type Default value: As defined in the dialog
	Possible Values:	

.vc05PercentPattern... vc90PercentPattern 01 - 11	Dots in foreground color on background color, the density of the foreground color increasing with the percentage 
.vcAeroGlassPattern 44	Vertical color gradient in the color of the fill pattern 
.vcBDDiagonalPattern 5	Diagonal lines slanting from bottom left to top right 
.vcCrossPattern 6	Cross-hatch pattern 
.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDDiagonalPattern and of twice the line width 
.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 
.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDDiagonalPattern and of twice the line width 
.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 
.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
.vcDashedHorizontalPattern 2026	Dashed horizontal lines 
.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 
.vcDashedVerticalPattern 2027	Dashed vertical lines 
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small 
.vcDiagonalBrickPattern 2032	Diagonal brick pattern 
.vcDivotPattern 2036	Divot pattern 

.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines 
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines 
.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right 
.vcHorizontalBrickPattern 2033	Horizontal brick pattern 
.vcHorizontalGradientPattern 52	Horizontal color gradient 
.vcHorizontalPattern 3	Horizontal lines 
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern 
.vcLargeConfettiPattern 2029	Confetti pattern, large 
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern 
.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern 
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern 
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern 
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern 
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern 
.vcNoPattern 1276	No fill pattern 
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large 
.vcPlaidPattern 2035	Plaid pattern 
.vcShinglePattern 2039	Diagonal shingle pattern 
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern 

.vcSmallConfettiPattern 2028	Confetti pattern
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern
.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares
.vcSpherePattern 2041	Checkerboard of spheres
.vcTrellisPattern 2040	Trellis pattern
.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright
.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright
.vcVerticalGradientPattern 62	Vertical color gradient
.vcVerticalPattern 2	Vertical lines
.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark
.vcWavePattern 2031	Horizontal waves pattern
.vcWeavePattern 2034	Interwoven stripes pattern
.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern
.vcWideUpwardDiagonalPattern 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of vcBDiagonalPattern
.vcZigZagPattern 2030	Horizontal zig-zag lines

PatternColor

Read Only Property of VcInterval

This property lets you set or retrieve the pattern color of the interval's calendar grid. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color. When casting an RGB value on an ARGB value, an alpha value of 255 has to be added.

The pattern color can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255})

Specification

Read Only Property of VcInterval

This property lets you retrieve the specification of an interval. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored smoothly to text files or data bases. This allows for persistency. A specification can be used to create an interval by the method **VcInterval-Collection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the interval

StartDateTime

Property of VcInterval

This property returns or sets the start date and time of this interval object (for profiles of the type **vcCalendar** only). This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	System.DateTime	Start date and time of interval

StartMonth

Property of VcInterval

This property returns or sets the start month of this interval object (for profiles of the type **vcYearProfile** only). This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	VcMonth	Start month of interval
	Possible Values: .vcApril 4 .vcAugust 8 .vcDecember 12 .vcFebruary 2 .vcJanuary 1 .vcJuly 7 .vcJune 6 .vcMarch 3 .vcMay 5 .vcNovember 11 .vcOktober 10 .vcSeptember 9	April August December February Januar July une March May November October September

StartTime

Property of VcInterval

This property returns or sets the start time of this interval object (for profiles of the type **vcDayProfile** only). This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	System.DateTime	Start time of interval

StartWeekday

Property of VcInterval

This property returns or sets the first weekday of this interval object (for profiles of the type **vcWeekProfile** only). This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	VcWeekday	Start weekday of interval
	Possible Values:	

.vcFriday 5	Week day Friday
.vcMonday 1	Week day Monday
.vcSaturday 6	Week day Saturday
.vcSunday 7	Week day Sunday
.vcThursday 4	Week day Thursday
.vcTuesday 2	Week day Tuesday
.vcWednesday 3	Week day Wednesday

Text

Property of VcInterval

This property lets you set or retrieve the text of the time ribbon *only for calendar profiles of the type **vcShiftProfile***. The text can also be set in the **Edit Shift Calendar** dialog.

	Data Type	Explanation
Property value	System.String	Annotation text of the time ribbon

TimeUnit

Property of VcInterval

This property lets you set or retrieve the time unit for the interval *only for calendar profiles of the type **vcVariableProfile***. The time unit can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	VcTimeUnit	Time unit Default value: vcDay
	Possible Values: .vcDay 5 .vcHour 6 .vcMinute 7 .vcSecond 8	Time unit day Time unit hour Time unit minute Time unit second

Type


Read Only Property of VcInterval

This property lets you enquire the type of the interval. This feature can also be set in the **Administrate Intervals** dialog.

	Data Type	Explanation
Property value	VcIntervalType Possible Values: .vcCalendarInterval 139 .vcDayProfileInterval 4 .vcIntervalProfileInterval 5 .vcWeekProfileInterval 3 .vcYearProfileInterval 2	Type of the interval

UseGraphicalAttributes

Read Only Property of VcInterval

This property lets you set or retrieve whether the graphical attributes that have been set for this interval shall be used. This feature can be also set in the dialog **Administrative Intervals** (which you reach by clicking  in the **Administrative Calendar Profiles** dialog). If they are to be used, the property **VcCalendarGrid.UseGraphicalAttributesOfIntervals** needs to have been set to **True**.

	Data Type	Explanation
Property value	System.Boolean	Graphical attributes of the interval are displayed (True)/are not displayed (False)

Methods

PutInOrderAfter

Method of VcInterval

This method lets you set the interval behind an interval specified by name, within the IntervalCollection. If you set the name to "", the interval will be put in the first position. The order of the intervals within the collection determines the order by which they apply to the calendars.

	Data Type	Explanation
Parameter: refName	System.String	Name of the interval behind which the current interval is to be put.
Return value	Void	

Example Code VB.NET

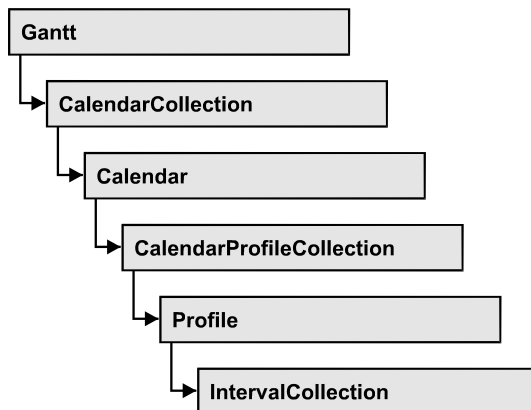
```
Dim intvlCltn As VcIntervalCollection
Dim intvl1 As VcInterval
Dim intvl2 As VcInterval

intvlCltn = VcGantt1.IntervalCollection()
intvl1 = intvlCltn.Add("intvl1")
intvl2 = intvlCltn.Add("intvl2")
intvl1.PutInOrderAfter("intvl2")
intvlCltn.Update()
```

Example Code C#

```
VcIntervalCollection intvlCltn = vcGantt1.IntervalCollection;
VcInterval intvl1 = intvlCltn.Add("intvl1");
VcInterval intvl2 = intvlCltn.Add("intvl2");
intvl1.PutInOrderAfter("intvl2");
intvlCltn.Update();
```


7.42 VcIntervalCollection



The VcIntervalCollection object contains all intervals available. You can access all objects in an iterative loop by **For Each Interval In BoxFormatCollection** or by the methods **First...** and **Next...**. You can access a single interval by the methods **IntervalByName** and **IntervalByIndex**. The number of intervals in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the intervals in the corresponding way.

Properties

- Count

Methods

- Add
- AddBySpecification
- Copy
- FirstInterval
- IntervalByIndex
- IntervalByName
- NextInterval
- Remove
- Update

Properties

Count

Read Only Property of VcIntervalCollection

This property lets you retrieve the number of intervals in the interval collection.

	Data Type	Explanation
Property value	System.Int32	Number of Interval objects

Methods

Add

Method of VcIntervalCollection

By this method you can create an interval as a member of the IntervalCollection. If the name has not been used before, the new interval object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ intervalName	System.String	Interval name
Return value	VcInterval	New interval object

AddBySpecification

Method of VcIntervalCollection

This method lets you create an interval by using an interval specification. This way of creating allows interval objects to become persistent. The specification of an interval can be saved and re-loaded (see VcInterval property **Specification**). In a subsequent session the interval can be created again from the specification including its former name.

	Data Type	Explanation
Parameter: ⇒ Specification	System.String	Interval specification
Return value	VcInterval	New Interval object

Copy

Method of VcIntervalCollection

By this method you can copy an interval. If the interval that is to be copied exists, and if the name for the new interval does not yet exist, the new interval object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ intervalName	System.String	Name of the interval to be copied
⇒ newIntervalName	System.String	Name of the new interval
Return value	VcInterval	interval object

FirstInterval

Method of VcIntervalCollection

This method can be used to access the initial value, i.e. the first interval of an interval collection, and then to continue in a forward iteration loop by the method **NextInterval** for the intervals following. If there is no interval in the FilterCollection object, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcInterval	First interval object

IntervalByIndex

Method of VcIntervalCollection

This method lets you access an interval by its index. If no interval of the specified index does exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ Index	System.Int16	Index of the interval
Return value	VcInterval	Interval object returned

IntervalByName

Method of VcIntervalCollection

By this method you can retrieve an interval by its name. If no interval of the specified name does exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ intervalName	System.String	Name of the interval object
Return value	VcInterval	interval object returned

NextInterval

Method of VcIntervalCollection

This method can be used in a forward iteration loop to retrieve subsequent intervals from an interval collection after initializing the loop by the method **FirstInterval**. If there is no interval left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcInterval	Subsequent interval object

Remove

Method of VcIntervalCollection

This method lets you delete an interval. If the interval is used in another object, it cannot be deleted. Then False will be returned, otherwise True.

	Data Type	Explanation
Parameter: ⇒ intervalName	System.String	interval name
Return value	System.Boolean	interval deleted (True)/not deleted (False)

Update

Method of VcIntervalCollection

This method lets you update an interval collection after having modified it.

	Data Type	Explanation
Return value	System.Boolean	update successful (True)/ not successful (False)

7.43 VcLayer



A layer is the graphical representation of a date (symbol layer) or a set of two dates (rectangle layer) within a node. A layer can be customized by a lot of attributes (shape, color, height, offset, contents of annotation fields, font).

Properties

- BackgroundColor
- BackgroundColorDataFieldIndex
- BackgroundColorMapName
- CompletionDataFieldIndex
- DurationDataFieldIndex
- EndDataFieldIndex
- EndSnapTarget
- FilterName
- Format
- GraphicsFileName
- GraphicsFileNameDataFieldIndex
- GraphicsFileNameMapName
- Height
- HeightDataFieldIndex
- HeightMapName
- HorizontalOffset
- LabelSizeDependence
- LegendText
- LineColor
- LineColorDataFieldIndex
- LineColorMapName
- MaximumEndDataFieldIndex
- MinimumStartDataFieldIndex
- Movable
- Name
- NonWorkIntervalBackgroundColor
- NonWorkIntervalBackgroundColorDataFieldIndex

- NonWorkIntervalBackgroundColorMapName
- NonWorkIntervalLineColor
- NonWorkIntervalLineColorDataFieldIndex
- NonWorkIntervalLineColorMapName
- NonWorkIntervalLineThickness
- NonWorkIntervalLineType
- NonWorkIntervalPattern
- NonWorkIntervalPatternColor
- NonWorkIntervalPatternColorDataFieldIndex
- NonWorkIntervalPatternColorMapName
- NonWorkIntervalPatternDataFieldIndex
- NonWorkIntervalPatternMapName
- NonWorkIntervalShape
- ObjectDrawEventsEnabled
- Pattern
- PatternColor
- PatternColorDataFieldIndex
- PatternColorMapName
- PatternDataFieldIndex
- PatternMapName
- Shape
- Sizeable
- Specification
- StartDataFieldIndex
- StartSnapTarget
- ThreeDEffect
- UsedAsOverlapLayer
- VerticalOffset
- VerticalOffsetDataFieldIndex
- VerticalOffsetMapName
- Visible
- VisibleInLegend

Methods

- CalculateCurrentWidth
- PutInOrderAfter

Properties

BackgroundColor

Property of VcLayer

This property lets you set retrieve the background color of the layer. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color.

If in the property **BackgroundColorMapName** a map is specified, the map will set the background color in dependence on data.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255})

BackgroundColorDataFieldIndex

Property of VcLayer

This property lets you set or retrieve the data field index to be used with the property **BackgroundColorMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection
Dim dataRec1 As VcDataRecord
Dim layer As VcLayer
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection
dataRec1 = dataRecCltn.Add("1;Node 1;01.01.14;;8;Red")

VcGantt1.EndLoading()

mapCltn = VcGantt1.MapCollection
map = mapCltn.Add("MapColor")

map.Type = VcMapType.vcColorMap
mapEntry = map.CreateEntry
mapEntry.DataFieldValue = "Green"
mapEntry.Color = System.Drawing.Color.Green
mapEntry = map.CreateEntry
mapEntry.DataFieldValue = "Red"
mapEntry.Color = System.Drawing.Color.Red
mapCltn.Update()

layer = VcGantt1.LayerCollection.LayerByIndex(0)
layer.BackgroundColorMapName = "MapColor"
layer.BackgroundColorDataFieldIndex = 5
VcGantt1.LayerCollection.Update()
```

Example Code C#

```
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

VcDataRecord dataRec1 = dataRecCltn.Add("1;Node 1;02.01.14;;8;Red");
vcGantt1.EndLoading();

VcMapCollection mapCltn = vcGantt1.MapCollection;
VcMap map = mapCltn.Add("MapColor");
map.Type = VcMapType.vcColorMap;

VcMapEntry mapEntry = map.CreateEntry();
mapEntry.DataFieldValue = "Green";
mapEntry.Color = System.Drawing.Color.Green;
mapEntry = map.CreateEntry();
mapEntry.DataFieldValue = "Red";
mapEntry.Color = System.Drawing.Color.Red;
mapCltn.Update();

VcLayer layer = vcGantt1.LayerCollection.LayerByIndex(0);
layer.BackgroundColorMapName = "MapColor";
layer.BackgroundColorDataFieldIndex = 5;
vcGantt1.LayerCollection.Update()
```

BackgroundColorMapName

Property of VcLayer

This property lets you set or retrieve the name of a color map (type vcColorMap). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **BackgroundColorDataFieldIndex**, then the background color is controlled by the map. If no data field entry applies, the background color of the layer that is specified in the property **BackgroundColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection
Dim dataRec1 As VcDataRecord
Dim layer As VcLayer
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection
dataRec1 = dataRecCltn.Add("1;Node 1;01.01.14;;8;Red")

VcGantt1.EndLoading()

mapCltn = VcGantt1.MapCollection
map = mapCltn.Add("MapColor")

map.Type = VcMapType.vcColorMap
mapEntry = map.CreateEntry
mapEntry.DataFieldValue = "Green"
mapEntry.Color = System.Drawing.Color.Green
mapEntry = map.CreateEntry
mapEntry.DataFieldValue = "Red"
mapEntry.Color = System.Drawing.Color.Red
mapCltn.Update()

layer = VcGantt1.LayerCollection.LayerByIndex(0)
layer.BackgroundColorMapName = "MapColor"
layer.BackgroundColorDataFieldIndex = 5
VcGantt1.LayerCollection.Update()
```

Example Code C#

```

VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

VcDataRecord dataRec1 = dataRecCltn.Add("1;Node 1;02.01.14;;8;Red");
vcGantt1.EndLoading();

VcMapCollection mapCltn = vcGantt1.MapCollection;
VcMap map = mapCltn.Add("MapColor");
map.Type = VcMapType.vcColorMap;

VcMapEntry mapEntry = map.CreateEntry();
mapEntry.DataFieldValue = "Green";
mapEntry.Color = System.Drawing.Color.Green;
mapEntry = map.CreateEntry();
mapEntry.DataFieldValue = "Red";
mapEntry.Color = System.Drawing.Color.Red;
mapCltn.Update();

VcLayer layer = vcGantt1.LayerCollection.LayerByIndex(0);
layer.BackgroundColorMapName = "MapColor";
layer.BackgroundColorDataFieldIndex = 5;
vcGantt1.LayerCollection.Update()

```

CompletionDataFieldIndex**Property of VcLayer**

This property lets you set or retrieve the data field that contains the percentage degree of completion of the layer.

The end date visualized by the layer is calculated from the start date field, the end date field or the duration respectively and the percent complete value. The data of the activity will not be changed.

This property is not available for symbol and bitmap layers.

	Data Type	Explanation
Property value	System.String	Index of the data field that contains the degree of completion

DurationDataFieldIndex**Property of VcLayer**

This property lets you set or retrieve the data field that contains the duration of the layer.

The unit of the duration will be interpreted in dependency on the time unit specified on the **General** property page.

This property is not available for symbol and bitmap layers.

	Data Type	Explanation
Property value	System.Int32	Index of the data field that contains the duration

EndDataFieldIndex

Property of VcLayer

This property lets you set or retrieve the data field that contains the end value of the layer, e.g. Early Start, Late Start, Scheduled Start.

To define a rectangle or line layer you need to specify a start and end field or a duration. If both an end field and a duration are specified, the duration entry overrides the end field entry. When an interaction occurs, not only the

duration field will be updated, but also the end field.

This property is not available for symbol and bitmap layers.

	Data Type	Explanation
Property value	System.Int16	Index of the data field that contains the end value

EndSnapTarget

Read Only Property of VcLayer

This property lets you set or retrieve whether the end date of this layer is to define as snap target.

	Data Type	Explanation
Property value	System.Boolean	End date of this layer is/is not defined as snap target

FilterName

Property of VcLayer

This property lets you specify the name of the filter that defines what activities the layer is to apply to.

	Data Type	Explanation
Property value	System.String	Filter name

Format

Read Only Property of VcLayer

This property lets you retrieve the format of the layer.

	Data Type	Explanation
Property value	VcLayerFormat	Layer format

GraphicsFileName

Property of VcLayer

This property lets you set or retrieve the name of a graphics file the content of which is displayed in the layer. The graphics file name has to denote an existing graphics file. *Available formats:*

- *.BMP (Microsoft Windows Bitmap)
- *.EMF (Enhanced Metafile or Enhanced Metafile Plus)
- *.GIF (Graphics Interchange Format)
- *.JPG (Joint Photographic Experts Group)
- *.PNG (Portable Network Graphics)
- *.TIF (Tagged Image File Format)
- *.VMF (Viewer Metafile)
- *.WMF (Microsoft Windows Metafile, probably with EMF included)

EMF, EMF+, VMF and WMF are vector formats that allow to store a file independent of pixel resolution. All other formats are pixel-oriented and confined to a limited resolution.

The VMF format basically has been deprecated, but it will still be supported for some time to maintain compatibility with existing applications.

For the graphics file to be displayed, independent of the format set here, the property **LayerShape** has to be set to **vcBitmapLayer**.

	Data Type	Explanation
Property value	System.String	Name of the graphics file

Example Code VB.NET

```

Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection
Dim dataRec1 As VcDataRecord
Dim layer As VcLayer
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection
dataRec1 = dataRecCltn.Add("1;Node 1;03.01.14;;8;Pic1.bmp")
VcGantt1.EndLoading()

mapCltn = VcGantt1.MapCollection
map = mapCltn.Add("MapGraphic")

map.Type = VcMapType.vcGraphicsFileMap
mapEntry = map.CreateEntry
mapEntry.GraphicsFileName = "c:\Pic1.bmp"
mapCltn.Update()

layer = VcGantt1.LayerCollection.LayerByIndex(0)
layer.Shape = VcLayerShape.vcBitmapLayer
layer.GraphicsFileName = "c:\Pic1.bmp"
layer.GraphicsFileNameMapName = "MapGraphic"
layer.GraphicsFileNameDataFieldIndex = 5
VcGantt1.LayerCollection.Update()

```

Example Code C#

```

VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

VcDataRecord dataRec1 = dataRecCltn.Add("1;Node 1;02.03.14;;8;Pic1.bmp");
vcGantt1.EndLoading();

VcMapCollection mapCltn = vcGantt1.MapCollection;
VcMap map = mapCltn.Add("MapGraphic");
map.Type = VcMapType.vcGraphicsFileMap;

VcMapEntry mapEntry = map.CreateEntry();
mapEntry.GraphicsFileName = @"c:\Pic1.bmp";
mapCltn.Update();

VcLayer layer = vcGantt1.LayerCollection.LayerByIndex(0);
layer.Shape = VcLayerShape.vcBitmapLayer;
layer.GraphicsFileName = "Pic1.bmp";
layer.GraphicsFileNameMapName = "MapGraphic";
layer.GraphicsFileNameDataFieldIndex = 5;
vcGantt1.LayerCollection.Update();

```

GraphicsFileNameDataFieldIndex**Property of VcLayer**

This property lets you set or retrieve the data field index that has to be specified if the property **GraphicsFileNameMapName** is used. If a valid data field index but no map is specified, the graphics file name will be loaded from the data field specified.

For the graphics file to be displayed, the property **LayerShape** has to be set to **vcBitmapLayer**.

	Data Type	Explanation
Property value	System.Int16	Index of the data field

Example Code VB.NET

```

Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection
Dim dataRec1 As VcDataRecord
Dim layer As VcLayer
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection
dataRec1 = dataRecCltn.Add("1;Node 1;03.01.14;;8;Pic1.bmp")
VcGantt1.EndLoading()

mapCltn = VcGantt1.MapCollection
map = mapCltn.Add("MapGraphic")

map.Type = VcMapType.vcGraphicsFileMap
mapEntry = map.CreateEntry
mapEntry.GraphicsFileName = "c:\Pic1.bmp"
mapCltn.Update()

layer = VcGantt1.LayerCollection.LayerByIndex(0)
layer.Shape = VcLayerShape.vcBitmapLayer
layer.GraphicsFileName = "c:\Pic1.bmp"
layer.GraphicsFileNameMapName = "MapGraphic"
layer.GraphicsFileNameDataFieldIndex = 5
VcGantt1.LayerCollection.Update()

```

Example Code C#

```

VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

VcDataRecord dataRec1 = dataRecCltn.Add("1;Node 1;02.03.14;;8;Pic1.bmp");
vcGantt1.EndLoading();

VcMapCollection mapCltn = vcGantt1.MapCollection;
VcMap map = mapCltn.Add("MapGraphic");
map.Type = VcMapType.vcGraphicsFileMap;

VcMapEntry mapEntry = map.CreateEntry();
mapEntry.GraphicsFileName = @"c:\Pic1.bmp";
mapCltn.Update();

VcLayer layer = vcGantt1.LayerCollection.LayerByIndex(0);
layer.Shape = VcLayerShape.vcBitmapLayer;
layer.GraphicsFileName = "Pic1.bmp";
layer.GraphicsFileNameMapName = "MapGraphic";
layer.GraphicsFileNameDataFieldIndex = 5;
vcGantt1.LayerCollection.Update();

```

GraphicsFileNameMapName**Property of VcLayer**

This property lets you set or retrieve the name of a map of the type **vcGraphicsFileMap** or **""**. Only if a name and a data field index are specified in the property **GraphicsFileNameDataFieldIndex**, the graphics

will be controlled by the map. If no data field entry applies, the graphics specified in the property **GraphicsFileName** will be displayed.

For the graphics file to be displayed, the property **LayerShape** has to be set to **vcBitmapLayer**.

	Data Type	Explanation
Property value	System.String	Name of the graphics map

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection
Dim dataRec1 As VcDataRecord
Dim layer As VcLayer
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection
dataRec1 = dataRecCltn.Add("1;Node 1;03.01.14;;8;Pic1.bmp")
VcGantt1.EndLoading()

mapCltn = VcGantt1.MapCollection
map = mapCltn.Add("MapGraphic")

map.Type = VcMapType.vcGraphicsFileMap
mapEntry = map.CreateEntry
mapEntry.GraphicsFileName = "c:\Pic1.bmp"
mapCltn.Update()

layer = VcGantt1.LayerCollection.LayerByIndex(0)
layer.Shape = VcLayerShape.vcBitmapLayer
layer.GraphicsFileName = "c:\Pic1.bmp"
layer.GraphicsFileNameMapName = "MapGraphic"
layer.GraphicsFileNameDataFieldIndex = 5
VcGantt1.LayerCollection.Update()
```

Example Code C#

```

VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

VcDataRecord dataRec1 = dataRecCltn.Add("1;Node 1;02.03.14;;8;Pic1.bmp");
vcGantt1.EndLoading();

VcMapCollection mapCltn = vcGantt1.MapCollection;
VcMap map = mapCltn.Add("MapGraphic");
map.Type = VcMapType.vcGraphicsFileMap;

VcMapEntry mapEntry = map.CreateEntry();
mapEntry.GraphicsFileName = @"c:\Pic1.bmp";
mapCltn.Update();

VcLayer layer = vcGantt1.LayerCollection.LayerByIndex(0);
layer.Shape = VcLayerShape.vcBitmapLayer;
layer.GraphicsFileName = "Pic1.bmp";
layer.GraphicsFileNameMapName = "MapGraphic";
layer.GraphicsFileNameDataFieldIndex = 5;
vcGantt1.LayerCollection.Update();

```

Height**Property of VcLayer**

This property lets you set or retrieve the height of the layer.

	Data Type	Explanation
Property value	System.Int32	Height in 1/100 mm

HeightDataFieldIndex**Property of VcLayer**

This property lets you set or retrieve the data field index that has to be specified if the property **HeightMapName** is used. If you set this property to **-1**, no map will be used.

This property will only become effective after the layer collection was updated by the method **Vc.LayerCollection.Update()**.

	Data Type	Explanation
Property value	System.Int32	Data field index

Example Code VB.NET

```

Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim layer As VcLayer

layer = VcGantt1.LayerCollection.FirstLayer
mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcMillimeterMap)
map = mapCltn.FirstMap
layer.HeightMapName = map.Name
ayer.HeightDataFieldIndex = VcGantt1.DetectFieldIndex("Maindata", "LayerHeight")
VcGantt1.LayerCollection.Update()

```

Example Code C#

```

VcLayer layer = vcGantt1.LayerCollection.FirstLayer();
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcMillimeterMap);
VcMap map = mapCltn.FirstMap();
layer.HeightMapName = map.Name;
layer.HeightDataFieldIndex = vcGantt1.DetectFieldIndex("Maindata",
"LayerHeight");
vcGantt1.LayerCollection.Update();

```

HeightMapName**Property of VcLayer**

This property lets you set or retrieve the name of a millimeter map (type `vcMillimeterMap`). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **HeightDataFieldIndex**, then the height is controlled by the map. If no data field entry applies, the height of the layer that is specified in the property **Height** will be used.

This property will only become effective after the layer collection was updated by the method **vcGantt1.LayerCollection.Update()**.

	Data Type	Explanation
Property value	System.String	Name of the millimetre map

Example Code VB.NET

```

Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim layer As VcLayer

layer = VcGantt1.LayerCollection.FirstLayer
mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcMillimeterMap)
map = mapCltn.FirstMap
layer.HeightMapName = map.Name
ayer.HeightDataFieldIndex = VcGantt1.DetectFieldIndex("Maindata", "LayerHeight")
VcGantt1.LayerCollection.Update()

```

Example Code C#

```

VcLayer layer = vcGantt1.LayerCollection.FirstLayer();
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcMillimeterMap);
VcMap map = mapCltn.FirstMap();
layer.HeightMapName = map.Name;
layer.HeightDataFieldIndex = vcGantt1.DetectFieldIndex("Maindata",
"LayerHeight");
vcGantt1.LayerCollection.Update();

```

HorizontalOffset**Property of VcLayer**

This property lets you set or retrieve the horizontal offset of the layer. This is only possible for symbol or bitmap layers. If you set an offset for other layer shapes, this will be without effect.

	Data Type	Explanation
Property value	System.Int16	Horizontal offset in % -50 ... 50

LabelSizeDependence**Property of VcLayer**

This property lets you set or retrieve, whether and how the size of the label is to be dependent on the size of the layer.

	Data Type	Explanation
Property value	VcLabelSizeDependence Possible Values: .vcFixedToBar 1 .vcTextHeightAndWidthIndependent 79 .vcTextHeightIndependent 39 .vcTextWidthIndependent 40	Dependence of the label on the layer size restricted by layer siz independent on text height and width independent on text height independent on text width

Example Code VB.NET

```

Dim layerCltn As VcLayerCollection
Dim layer As VcLayer

layerCltn = VcGantt1.LayerCollection
layer = layerCltn.LayerByName("Start End")
layer.LabelSizeDependence = VcLabelSizeDependence.vcFixedToBar

```

Example Code C#

```
VcLayerCollection layerCltn = vcGantt1.LayerCollection;
VcLayer layer = layerCltn.LayerByName("Start-End");
layer.LabelSizeDependence = VcLabelSizeDependence.vcFixedToBar;
```

LegendText**Property of VcLayer**

This property lets you set or retrieve the legend text of a layer. When set to "", the layer name (property **Name**) will be displayed.

	Data Type	Explanation
Property value	System.String	Legend text of the layer Default value: " " (content of the property Name)

LineColor**Property of VcLayer**

This property lets you set or retrieve the color of the (border) line of the layer.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255})

LineColorDataFieldIndex**Property of VcLayer**

This property lets you set or retrieve the data field index to be used with a map specified by the property **LineColorMapName**. If you set this property to -1, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

LineColorMapName

Property of VcLayer

This property lets you set or retrieve the name of a map for the line color. If set to "" or if the property **LineColorDataFieldIndex** is set to **-1**, then no map will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

MaximumEndDataFieldIndex

Property of VcLayer

If this property is set to a valid field index, the date and time of the corresponding field are considered as upper limit for the end time of the layer when a layer or a node is moved interactively .

This property can also be set in the **Edit Layer** dialog.

	Data Type	Explanation

MinimumStartDataFieldIndex

Read Only Property of VcLayer

If this property is set to a valid field index, the date and time of the corresponding field are considered as lower limit for the start time of the layer when a layer or a node is moved interactively .

This property can also be set in the **Edit Layer** dialog.

	Data Type	Explanation
Property value	System.Int32	Data field index for earliest start time Default value: -1

Movable

Property of VcLayer

This property lets you set or retrieve whether a layer can be moved interactively.

	Data Type	Explanation
Property value	System.Boolean	Movable (True)/ not Movable (False) Default value: True

Example Code VB.NET

```
Dim layer As VcLayer

layer = VcGantt1.LayerCollection.LayerByName("layer1")
layer.Movable = False
```

Example Code C#

```
VcLayer VcLayer = vcGantt1.LayerCollection.LayerByName("layer1");
layer.Movable = false;
```

Name

Read Only Property of VcLayer

This property lets you retrieve the name of a layer.

	Data Type	Explanation
Property value	System.String	Name of the layer

Example Code VB.NET

```
Dim layerCltn As VcLayerCollection
Dim layer As VcLayer

layerCltn = VcGantt1.LayerCollection
For Each layer In layerCltn
    ListBox1.Items.Add(layer.Name)
Next
```

Example Code C#

```
VcLayerCollection layerCltn = vcGantt1.LayerCollection;
foreach(VcLayer layer in layerCltn)
    listBox1.Items.Add(layer.Name);
```

NonWorkIntervalBackgroundColor

Property of VcLayer

This property lets you set retrieve the background color of the layer. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color.

If in the property **NonWorkIntervalBackgroundColorMapName** a map is specified, the map will set the background color in dependence on data.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255})

NonWorkIntervalBackgroundColorDataFieldIndex

Property of VcLayer

This property lets you set or retrieve the data field index to be used with the property **NonWorkIntervalBackgroundColorMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

NonWorkIntervalBackgroundColorMapName

Property of VcLayer

This property lets you set or retrieve the name of a color map (type vcColorMap). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **NonWorkIntervalBackgroundColorDataFieldIndex**, then the background color is controlled by the map. If no data field entry applies, the background color of the layer that is specified in the property **NonWorkIntervalBackgroundColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

NonWorkIntervalLineColor

Property of VcLayer

This property lets you set or retrieve the color of the (border) line of the layer.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255})

NonWorkIntervalLineColorDataFieldIndex

Property of VcLayer

This property lets you set or retrieve the data field index to be used with a map specified by the property **NonWorkIntervalLineColorMapName**. If you set this property to -1, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

NonWorkIntervalLineColorMapName

Property of VcLayer

This property lets you set or retrieve the name of a map for the line color. If set to "" or if the property **NonWorkIntervalLineColorDataFieldIndex** is set to **-1**, then no map will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

NonWorkIntervalLineThickness

Property of VcLayer

This property lets you set or retrieve the thickness of the (border) line of the layer.

If you set this property to values between 1 and 4, an absolute line thickness is defined in pixels. Irrespective of the zoom factor a line will always show the same line thickness in pixels. When printing though, the line thickness is adapted for the sake of legibility and becomes dependent of the zoom factor:

Value	Points	mm
1	1/2 point	0.09 mm
2	1 point	0.18 mm
3	3/2 points	0.26 mm
4	2 points	0.35 mm

A point equals 1/72 inch and represents the unit of the font size.

If you set this property to values between 5 and 1,000, the line thickness is defined in 1/100 mm, so the lines will be displayed in a true thickness in pixels that depends on the zoom factor.

	Data Type	Explanation
Property value	System.Int16	Line thickness LineType {1...4}: line thickness in pixels LineType {5...1000}: line thickness in 1/100 mm Default value: As defined in the dialog

NonWorkIntervalLineType

Property of VcLayer

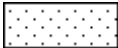







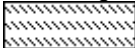
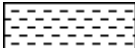
This property lets you set or retrieve the type of the (border) line of the layer.




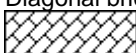
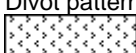

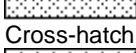
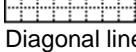


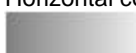
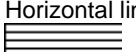
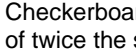


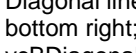

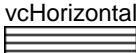
	Data Type	Explanation

NonWorkIntervalPattern

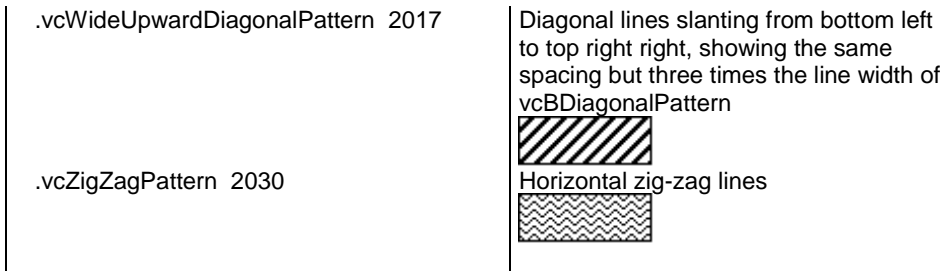
Property of VcLayer

This property lets you set or retrieve the pattern of the layer. If in the property **NonWorkIntervalPatternMapName** a map is specified, this map will control the pattern dependent on the data.

	Data Type	Explanation
Property value	VcFillPattern	Pattern type Default value: As defined in the dialog
	Possible Values: .vc05PercentPattern... .vc90PercentPattern 01 - 11	Dots in foreground color on background color, the density of the foreground color increasing with the percentage 
	.vcAeroGlassPattern 44	Vertical color gradient in the color of the fill pattern 
	.vcBDiagonalPattern 5	Diagonal lines slanting from bottom left to top right 
	.vcCrossPattern 6	Cross-hatch pattern 
	.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width 
	.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 
	.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width 
	.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 
	.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
	.vcDashedHorizontalPattern 2026	Dashed horizontal lines 

.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 
.vcDashedVerticalPattern 2027	Dashed vertical lines 
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small 
.vcDiagonalBrickPattern 2032	Diagonal brick pattern 
.vcDivotPattern 2036	Divot pattern 
.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines 
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines 
.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right 
.vcHorizontalBrickPattern 2033	Horizontal brick pattern 
.vcHorizontalGradientPattern 52	Horizontal color gradient 
.vcHorizontalPattern 3	Horizontal lines 
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern 
.vcLargeConfettiPattern 2029	Confetti pattern, large 
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern 
.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern 
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern 
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern 
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern 

.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern
.vcNoPattern 1276	No fill pattern
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large
.vcPlaidPattern 2035	Plaid pattern
.vcShinglePattern 2039	Diagonal shingle pattern
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern
.vcSmallConfettiPattern 2028	Confetti pattern
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern
.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares
.vcSpherePattern 2041	Checkerboard of spheres
.vcTrellisPattern 2040	Trellis pattern
.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright
.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright
.vcVerticalGradientPattern 62	Vertical color gradient
.vcVerticalPattern 2	Vertical lines
.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark
.vcWavePattern 2031	Horizontal waves pattern
.vcWeavePattern 2034	Interwoven stripes pattern
.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern



NonWorkIntervalPatternColor

Property of VcLayer

This property lets you set or retrieve the pattern color of the layer. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color.

If by the property **NonWorkIntervalPatternColorMapName** a map was specified, the map will set the pattern in dependence of data.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255})

NonWorkIntervalPatternColorDataFieldIndex

Property of VcLayer

This property lets you set or retrieve the data field index that has to be specified if the property **NonWorkIntervalPatternColorMapName** is used. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

NonWorkIntervalPatternColorMapName

Property of VcLayer

This property lets you set or retrieve the name of a color map (type `vcColorMap`). If set to "", no map will be used. Only if a map name and a data field index are specified in the property **NonWorkIntervalPatternColorDataFieldIndex**, the pattern color is controlled by the map. If no data field entry applies, the pattern color of the layer that is specified in the property **NonWorkIntervalPatternColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

NonWorkIntervalPatternDataFieldIndex

Property of VcLayer

This property lets you set or retrieve the data field index to be used together with the property `NonWorkIntervalPatternMapName`. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

NonWorkIntervalPatternMapName

Property of VcLayer

This property lets you set or retrieve the name of a pattern map (type `vcPatternMap`). If set to "", no map will be used. Only if a map name and additionally a data field index are specified in the property **PatternDataFieldIndex**, the pattern is controlled by the map. If no data field entry applies, the pattern of the layer that is specified in the property **Pattern** will be used.

	Data Type	Explanation

NonWorkIntervalShape

Property of VcLayer

This property lets you set or retrieve the form of non work intervals in rectangle layers. It can also be set in the **Edit layer** dialog.

	Data Type	Explanation
Property value	VcNonWorkIntervalShape Possible Values: .vcEmptyArea 2 .vcLine 1 .vcNo 0 .vcRectangle 112	Form of non work intervals in rectangle layers work free intervals are displayed as empty area work free intervals are displayed as line work free intervals are not displayed work free intervals are displayed as rectangle

ObjectDrawEventsEnabled

Property of VcLayer


If this property is set to **true**, the events **VcObjectDrawn** and **VcObjectDrawing** are enabled for nodes which are drawn with this layer or for annotation ribbons.








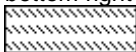
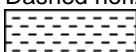



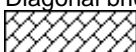
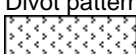

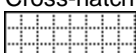
	Data Type	Explanation
Property value	System.Boolean	ObjectDraw events enabled (True) or disabled (False) Default value: False


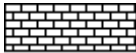
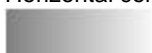
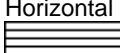
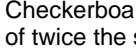


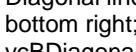

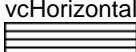
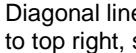

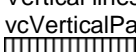
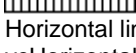


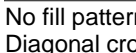

Pattern

Property of VcLayer

This property lets you set or retrieve the pattern of the layer. If in the property **PatternMapName** a map is specified, this map will control the pattern dependent on the data.

	Data Type	Explanation
Property value	VcFillPattern Possible Values: .vc05PercentPattern... vc90PercentPattern 01 - 11	Pattern type Default value: As defined in the dialog Dots in foreground color on background color, the density of the foreground color increasing with the percentage 

.vcAeroGlassPattern 44	Vertical color gradient in the color of the fill pattern 
.vcBDiagonalPattern 5	Diagonal lines slanting from bottom left to top right 
.vcCrossPattern 6	Cross-hatch pattern 
.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width 
.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 
.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width 
.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 
.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
.vcDashedHorizontalPattern 2026	Dashed horizontal lines 
.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 
.vcDashedVerticalPattern 2027	Dashed vertical lines 
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small 
.vcDiagonalBrickPattern 2032	Diagonal brick pattern 
.vcDivotPattern 2036	Divot pattern 
.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines 
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines 

.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right 
.vcHorizontalBrickPattern 2033	Horizontal brick pattern 
.vcHorizontalGradientPattern 52	Horizontal color gradient 
.vcHorizontalPattern 3	Horizontal lines 
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern 
.vcLargeConfettiPattern 2029	Confetti pattern, large 
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcBDiagonalPattern 
.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern 
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern 
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern 
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern 
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern 
.vcNoPattern 1276	No fill pattern
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large 
.vcPlaidPattern 2035	Plaid pattern 
.vcShinglePattern 2039	Diagonal shingle pattern 
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern 
.vcSmallConfettiPattern 2028	Confetti pattern 
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern 

<code>.vcSolidDiamondPattern</code> 2046	Checkerboard pattern showing diagonal squares
<code>.vcSpherePattern</code> 2041	Checkerboard of spheres
<code>.vcTrellisPattern</code> 2040	Trellis pattern
<code>.vcVerticalBottomLightedConvexPattern</code> 43	Vertical color gradient from dark to bright
<code>.vcVerticalConcavePattern</code> 40	Vertical color gradient from dark to bright to dark
<code>.vcVerticalConvexPattern</code> 41	Vertical color gradient from bright to dark to bright
<code>.vcVerticalGradientPattern</code> 62	Vertical color gradient
<code>.vcVerticalPattern</code> 2	Vertical lines
<code>.vcVerticalTopLightedConvexPattern</code> 42	Vertical color gradient from bright to dark
<code>.vcWavePattern</code> 2031	Horizontal waves pattern
<code>.vcWeavePattern</code> 2034	Interwoven stripes pattern
<code>.vcWideDownwardDiagonalPattern</code> 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of <code>vcF-DiagonalPattern</code>
<code>.vcWideUpwardDiagonalPattern</code> 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of <code>vcBDiagonalPattern</code>
<code>.vcZigZagPattern</code> 2030	Horizontal zig-zag lines

PatternColor

Property of VcLayer

This property lets you set or retrieve the pattern color of the layer. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An

alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color.

If by the property **PatternColorMapName** a map was specified, the map will set the pattern in dependence of data.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255})

PatternColorDataFieldIndex

Property of VcLayer

This property lets you set or retrieve the data field index that has to be specified if the property **PatternColorMapName** is used. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

PatternColorMapName

Property of VcLayer

This property lets you set or retrieve the name of a color map (type vcColorMap). If set to "", no map will be used. Only if a map name and a data field index are specified in the property **PatternColorDataFieldIndex**, the pattern color is controlled by the map. If no data field entry applies, the pattern color of the layer that is specified in the property **PatternColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

PatternDataFieldIndex

Property of VcLayer

This property lets you set or retrieve the data field index to be used together with the property **PatternMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection
Dim dataRec1 As VcDataRecord
Dim layer As VcLayer
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection
dataRec1 = dataRecCltn.Add("1;Node 1;01.01.14;;8;Horizontal")
VcGantt1.EndLoading()

mapCltn = VcGantt1.MapCollection
map = mapCltn.Add("MapPattern")

map.Type = VcMapType.vcPatternMap
mapEntry = map.CreateEntry
mapEntry.DataFieldValue = "Diagonal"
mapEntry.Pattern = VcFillPattern.vcFDiagonalPattern
mapEntry = map.CreateEntry
mapEntry.DataFieldValue = "Horizontal"
mapEntry.Pattern = VcFillPattern.vcHorizontalPattern
mapCltn.Update()

layer = VcGantt1.LayerCollection.LayerByIndex(0)
layer.PatternMapName = "MapPattern"
layer.PatternDataFieldIndex = 5
VcGantt1.LayerCollection.Update()
```

Example Code C#

```

VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

VcDataRecord dataRec1 = dataRecCltn.Add("1;Node 1;02.01.14;;8;Horizontal");
vcGantt1.EndLoading();

VcMapCollection mapCltn = vcGantt1.MapCollection;
VcMap map = mapCltn.Add("MapPattern");
map.Type = VcMapType.vcPatternMap;

VcMapEntry mapEntry = map.CreateEntry();
mapEntry.DataFieldValue = "Diagonal";
mapEntry.Pattern = VcFillPattern.vcFDDiagonalPattern;
mapEntry = map.CreateEntry();
mapEntry.DataFieldValue = "Horizontal";
mapEntry.Pattern = VcFillPattern.vcHDDiagonalPattern;
mapCltn.Update();

VcLayer layer = vcGantt1.LayerCollection.LayerByIndex(0);
layer.PatternMapName = "MapPattern";
layer.PatternDataFieldIndex = 5;
vcGantt1.LayerCollection.Update();

```

PatternMapName**Property of VcLayer**

This property lets you set or retrieve the name of a pattern map (type `vcPatternMap`). If set to "", no map will be used. Only if a map name and additionally a data field index are specified in the property **PatternDataFieldIndex**, the pattern is controlled by the map. If no data field entry applies, the pattern of the layer that is specified in the property **Pattern** will be used.

	Data Type	Explanation
Parameter: ⇒ Rückgabewert	System.String	Name of the pattern map
Property value	System.String	Name of the pattern map

Example Code VB.NET

```
Dim dataTable As VcDataTable
Dim dataRecCltn As VcDataRecordCollection
Dim dataRec1 As VcDataRecord
Dim layer As VcLayer
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

dataTable = VcGantt1.DataTableCollection.DataTableByName("Maindata")
dataRecCltn = dataTable.DataRecordCollection
dataRec1 = dataRecCltn.Add("1;Node 1;01.01.14;;8;Horizontal")
VcGantt1.EndLoading()

mapCltn = VcGantt1.MapCollection
map = mapCltn.Add("MapPattern")

map.Type = VcMapType.vcPatternMap
mapEntry = map.CreateEntry
mapEntry.DataFieldValue = "Diagonal"
mapEntry.Pattern = VcFillPattern.vcFDDiagonalPattern
mapEntry = map.CreateEntry
mapEntry.DataFieldValue = "Horizontal"
mapEntry.Pattern = VcFillPattern.vcHDDiagonalPattern
mapCltn.Update()

layer = VcGantt1.LayerCollection.LayerByIndex(0)
layer.PatternMapName = "MapPattern"
layer.PatternDataFieldIndex = 5
VcGantt1.LayerCollection.Update()
```

Example Code C#

```
VcDataTable dataTable =
vcGantt1.DataTableCollection.DataTableByName("Maindata");
VcDataRecordCollection dataRecCltn = dataTable.DataRecordCollection;

VcDataRecord dataRec1 = dataRecCltn.Add("1;Node 1;02.01.14;;8;Horizontal");
vcGantt1.EndLoading();

VcMapCollection mapCltn = vcGantt1.MapCollection;
VcMap map = mapCltn.Add("MapPattern");
map.Type = VcMapType.vcPatternMap;

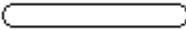


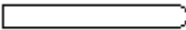
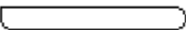
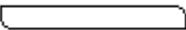

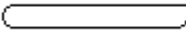

VcMapEntry mapEntry = map.CreateEntry();
mapEntry.DataFieldValue = "Diagonal";
mapEntry.Pattern = VcFillPattern.vcFDDiagonalPattern;
mapEntry = map.CreateEntry();
mapEntry.DataFieldValue = "Horizontal";
mapEntry.Pattern = VcFillPattern.vcHDDiagonalPattern;
mapCltn.Update();


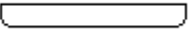
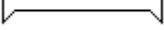
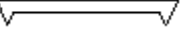
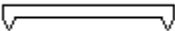
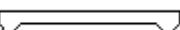















VcLayer layer = vcGantt1.LayerCollection.LayerByIndex(0);
layer.PatternMapName = "MapPattern";
layer.PatternDataFieldIndex = 5;
vcGantt1.LayerCollection.Update();
```

Shape

Property of VcLayer

This property lets you set or retrieve the shape of the layer. In the symbols below, black sections can be color-coded (please see **BackgroundColor-AsARGB**, **Pattern** and **PatternColor**).

Property value	Data Type	Explanation
	VcLayerShape	Layer shape
	Possible Values:	
	.vcAllRoundedRectangleLayer 61441	All corners rounded 
	.vcBitmapLayer 103007	Layer form bitmap <Bitmap-Layer>
	.vcInvisibleSymbolLayer 101000	Layer invisible <unsichtbares Symbol>
	.vcLineLayer 2	Layershape line 
	.vcNAndSERoundedRectangleLayer 45057	Rounded top left, top right and bottom right corner 
	.vcNAndSWRoundedRectangleLayer 28673	Rounded top left, top right and bottom left corner 
	.vcNEAndSERoundedRectangleLayer 40961	Rounded top right and bottom right corner 
	.vcNEAndSRoundedRectangleLayer 57345	Rounded bottom left, bottom right and top right corner 
	.vcNEAndSWRoundedRectangleLayer 24577	Rounded bottom left and top right corner 
	.vcNERoundedRectangleLayer 8193	Rounded top right corner 
	.vcNRoundedRectangleLayer 12289	Rounded top right and top left corner 
	.vcNWAndSERoundedRectangleLayer 36865	Rounded top left and bottom right corner 
	.vcNWAndSRoundedRectangleLayer 52349	Rounded top left, bottom left and bottom right corner 
	.vcNWAndSWRoundedRectangleLayer 20481	Rounded bottom left and top left corner 
	.vcNWRoundedRectangleLayer 4097	Rounded top left corner 
	.vcRectangleLayer 1	

.vcSERoundedRectangleLayer 32769	Rounded bottom right corner 
.vcSRoundedRectangleLayer 49153	Rounded bottom right and bottom left corner 
.vcSummaryBar1 1858	Summary bar 
.vcSummaryBar2 1859	Summary bar 
.vcSummaryBar3 1860	Summary bar 
.vcSummaryBar4 1861	Summary bar 
.vcSWRoundedRectangleLayer 16385	Rounded bottom left corner 
.vcSymbolLayer1 101001	Arrow pointing downward (black can be replaced by color) 
.vcSymbolLayer10 101010	Square (black can be replaced by color) 
.vcSymbolLayer11 101032	Circle 
.vcSymbolLayer12 101033	Arrow down in circle 
.vcSymbolLayer13 101034	Triangle in circle, tip pointing down 
.vcSymbolLayer14 101035	Pointed bracket in circle, right one 
.vcSymbolLayer15 101036	Narrow triangle in circle, tip pointing up 
.vcSymbolLayer16 101037	Triangle in circle, tip pointing right 
.vcSymbolLayer17 101038	Triangle in circle, tip pointing left 
.vcSymbolLayer18 101039	Square sitting on tip in circle 
.vcSymbolLayer19 101040	Two narrow triangles in circle, position horizontal, tips pointing to center 
.vcSymbolLayer2 101002	Triangle, tip pointing downward 
.vcSymbolLayer20 101041	Narrow triangle in circle, pointing down 
.vcSymbolLayer21 101042	Square in circle 

.vcSymbolLayer22 103001	Circle
.vcSymbolLayer23 102031	Arrow up
.vcSymbolLayer24 102034	Triangle, tip up
.vcSymbolLayer25 102016	Pointed bracket, left one
.vcSymbolLayer26 102051	Arrow up in circle
.vcSymbolLayer27 102054	Triangle in circle, tip pointing up
.vcSymbolLayer3 101003	Right pointed bracket
.vcSymbolLayer4 101004	Narrow triangle, tip pointing up
.vcSymbolLayer5 101005	Triangle, tip pointing right
.vcSymbolLayer6 101006	Triangle, tip pointing left
.vcSymbolLayer7 101007	Square sitting on tip
.vcSymbolLayer8 101008	Two narrow triangles, position horizontal, tips pointing to center
.vcSymbolLayer9 101009	Narrow triangle, tip pointing down
.vcTriangleBottomLeftLayer 1566	Triangle layer, tip pointing to the left
.vcTriangleBottomRightLayer 1564	Triangle layer, tip pointing to the right

Sizeable

Property of VcLayer

This property lets you set or retrieve whether the layer size can be changed interactively.

	Data Type	Explanation
Property value	VcLayerSizeability	Mode of layer sizeability
		Default value: True
	Possible Values:	
	.vcSizeableLeft 1	
	.vcSizeableLeftRight 3	
	.vcSizeableNone 0	

.vcSizeableRight 2

Example Code VB.NET

```
Dim layer As VcLayer

layer = vcGantt1.LayerCollection.LayerByName("layer1")
layer.Sizeable = VcLayerSizeability.vcSizeableLeftRight
```

Example Code C#

```
VcLayer VcLayer = VcGantt1.LayerCollection.LayerByName("layer1");
layer.Sizeable = VcLayerSizeability.vcSizeableLeftRight;
```

Specification

Read Only Property of VcLayer

This property lets you retrieve the specification of a layer. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored without problems to text files or data bases. This allows for persistency. A specification can be used to create a layer by the method **VcLayerCollection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the layer

StartDataFieldIndex

Property of VcLayer

This property lets you set or retrieve the data field that contains the start value of the layer, e.g. Early Start, Late Start, Scheduled Start.

	Data Type	Explanation
Property value	System.Int16	Index of the data field that contains the start value

StartSnapTarget

Read Only Property of VcLayer

This property lets you set or retrieve whether the start date of this layer is to define as snap target.

	Data Type	Explanation
Property value	System.Boolean	Start date of this layer is/is not defined as snap target

ThreeDEffect

Property of VcLayer

This property lets you set or retrieve whether the layer will be highlighted by a 3D effect.

	Data Type	Explanation
Property value	System.Boolean	3D effect switched on (True)/switched off (False) Default value: False

UsedAsOverlapLayer

Property of VcLayer

This property lets you set or retrieve whether this layer is to be used as an overlap layer. Overlap layers occur to indicate whether two different nodes overlap. They grow and shrink correspondingly to the size of the overlapping parts and therefore indicate the degree of hiding. (Cf. also **VcGantt.OverlapLayerEnabled** and **VcGantt.OverlapLayerName**).

	Data Type	Explanation
Property value	System.Boolean	True: layer is used as an overlap layer; False: layer is not used as an overlap layer. Default value: False

Example Code VB.NET

```
Dim layer As VcLayer

layer = VcGantt1.LayerCollection.LayerByName("layer1")
layer.UsedAsOverlapLayer = False
```

Example Code C#

```
VcLayer VcLayer = vcGantt1.LayerCollection.LayerByName("layer1");
layer.UsedAsOverlapLayer = false;
```

VerticalOffset

Property of VcLayer

This property lets you set or retrieve the vertical offset of the layer. If in the property **VerticalOffsetMapName** a map is specified, this map will control the vertical offset dependent on the data.

	Data Type	Explanation
Property value	System.Int32	Vertical offset. Unit: 1/100 mm

VerticalOffsetDataFieldIndex

Property of VcLayer

This property lets you set or retrieve the data field index

that has to be specified if the property **VerticalOffsetMapName** is used. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

VerticalOffsetMapName

Property of VcLayer

This property lets you set or retrieve the name of a millimeter map (type vcMillimeterMap). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **VerticalOffsetDataFieldIndex**, then the vertical offset is controlled by the map. If no data field entry applies, the vertical offset of the layer that is specified in the property **VerticalOffset** will be used.

	Data Type	Explanation
Property value	System.String	Name of the millimetre map

Visible

Property of VcLayer

This property lets you set or retrieve whether a layer is visible.

	Data Type	Explanation
Property value	System.Boolean	Layer visible/invisible Default value: True

Example Code VB.NET

```
Dim layerCltn As VcLayerCollection
Dim layer As VcLayer

layerCltn = VcGantt1.LayerCollection
layer = layerCltn.LayerByName("Start End")
layer.Visible = False
```

Example Code C#

```
VcLayerCollection layerCltn = vcGantt1.LayerCollection;
VcLayer layer = layerCltn.LayerByName("Start-End");
layer.Visible = false;
```

VisibleInLegend

Property of VcLayer

This property lets you set or retrieve whether a layer object is to be visible in the legend. This property also can be set by the **Specify Bar Appearance** dialog.

	Data Type	Explanation
Property value	System.Boolean	Layer visible in legend (True)/ invisible in legend (False) Default value: True

Example Code VB.NET

```
Dim layerCltn As VcLayerCollection
Dim layer As VcLayer

layerCltn = VcGantt1.LayerCollection
layer = layerCltn.LayerByName("Start End")
layer.VisibleInLegend = False
```

Example Code C#

```
VcLayerCollection layerCltn = vcGantt1.LayerCollection;
VcLayer layer = layerCltn.LayerByName("Start-End");
layer.VisibleInLegend = false;
```

Methods

CalculateCurrentWidth

Method of VcLayer

This method calculates the current width of the layer which belongs to the layer definition of the node specified. The width unit is 1/100 mm. If no layer in the layer definition of the node is visible, for example due to filter conditions, **-1** will be returned.

	Data Type	Explanation
Parameter: ⇨ node	VcNode	Node, in the layer definition of which the layer is looked for.
Return value	System.Int32	Width of the layer in 1/100 mm

PutInOrderAfter

Method of VcLayer

This method lets you set the layer behind a layer specified by name, within the LayerCollection. If you set the name to "", the layer will be put in the first position. The order of the layers determines the order by which they are displayed.

	Data Type	Explanation
Parameter: ⇨ refName	System.String	Name of the layer behind which the current layer is to be put.
Return value	Void	

Example Code VB.NET

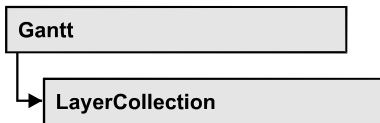
```
Dim layerCltn As VcLayerCollection
Dim layer1 As VcLayer
Dim layer2 As VcLayer

layerCltn = VcGantt1.LayerCollection()
layer1 = layerCltn.Add("layer1")
layer2 = layerCltn.Add("layer2")
layer1.PutInOrderAfter("layer2")
layerCltn.Update()
```

Example Code C#

```
VcLayerCollection layerCltn = vcGantt1.LayerCollection;  
VcLayer layer1 = layerCltn.Add("layer1");  
VcLayer layer2 = layerCltn.Add("layer2");  
layer1.PutInOrderAfter("layer2");  
layerCltn.Update();
```


7.44 VcLayerCollection



The LayerCollection object automatically contains all available layers . You can access all objects in an iterative loop by **For Each layer In LayerCollection** or by the methods **First...** and **Next...**. You can access a single layer using the methods **LayerByName** and **LayerByIndex**. The number of layers in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the layers in the corresponding way.

Properties

- Count

Methods

- Add
- AddBySpecification
- Copy
- FirstLayer
- GetEnumerator
- LayerByIndex
- LayerByName
- NextLayer
- Remove
- Update

Properties

Count

Read Only Property of VcLayerCollection

This property lets you retrieve the number of layers in the layer collection.

	Data Type	Explanation
Property value	System.Int32	Number of layers

Example Code VB.NET

```
Dim numberOfLayers As Integer

numberOfLayers = VcGantt1.LayerCollection.Count
```

Example Code C#

```
int numberOfLayers = vcGantt1.LayerCollection.Count;
```

Methods

Add

Method of VcLayerCollection

By this method you can create a layer as a member of the LayerCollection. If the name has not been used before, the new layer object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ layerName	System.String	Layer name
Return value	VcLayer	New layer object

Example Code VB.NET

```
newlayer = VcGantt1.LayerCollection.Add("test1")
```

Example Code C#

```
VcLayer newLayer = vcGantt1.LayerCollection.Add("test1");
```

AddBySpecification

Method of VcLayerCollection

This method lets you create a layer by using a layer specification. This way of creating allows layer objects to become persistent. The specification of a layer can be saved and re-loaded (see VcLayer property **Specification**). In a subsequent session the layer can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter: ⇒ specification	System.String	Specification of the layer

Return value	VcLayer	New layer object
---------------------	---------	------------------

Copy

Method of VcLayerCollection

By this method you can copy a layer. If the layer that is to be copied exists, and if the name for the new layer does not yet exist, the new layer object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter:		
⇒ layerName	System.String	Name of the layer to be copied
⇒ newLayerName	System.String	Name of the new layer
Return value	VcLayer	Layer object

FirstLayer

Method of VcLayerCollection

This method can be used to access the initial value, i.e. the first layer of a layer collection and then to continue in a forward iteration loop by the method **NextLayer** for the layers following. If there is no layer in the layer collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcLayer	First Layer

Example Code VB.NET

```
Dim layerCltn As VcLayerCollection
Dim layer As VcLayer

layerCltn = VcGantt1.LayerCollection
layer = layerCltn.FirstLayer
```

Example Code C#

```
VcLayerCollection layerCltn = vcGantt1.LayerCollection;
VcLayer layer = layerCltn.FirstLayer();
```

GetEnumerator

Method of VcLayerCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the layer objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

LayerByIndex

Method of VcLayerCollection

This method lets you access a layer by its index. If a layer does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the layer
Return value	VcLayer	Layer object returned

LayerByName

Method of VcLayerCollection

This method retrieves a layer by its name. If a layer of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ layerName	System.String	Name of layer
Return value	VcLayer	Layer

Example Code VB.NET

```
Dim layerCltn As VcLayerCollection
Dim layer As VcLayer

layerCltn = VcGantt1.LayerCollection
layer = layerCltn.LayerByName("Start-End")
```

Example Code C#

```
VcLayerCollection layerCltn = vcGantt1.LayerCollection;
VcLayer layer = layerCltn.LayerByName("Start-End");
```

NextLayer**Method of VcLayerCollection**

This method can be used in a forward iteration loop to retrieve subsequent layers from a layer collection after initializing the loop by the method **FirstLayer**. If there is no layer left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcLayer	Next Layer

Example Code VB.NET

```
Dim layerCltn As VcLayerCollection
Dim layer As VcLayer

layerCltn = VcGantt1.LayerCollection
layer = layerCltn.FirstLayer
While Not layer Is Nothing
    ListBox1.Items.Add(layer.Name)
    layer = layerCltn.NextLayer
End While
```

Example Code C#

```
VcLayerCollection layerCltn = vcGantt1.LayerCollection;
VcLayer layer = layerCltn.FirstLayer();

while (layer != null)
{
    listBox1.Items.Add(layer.Name);
    layer = layerCltn.NextLayer();
}
```

Remove**Method of VcLayerCollection**

This method lets you delete a layer. If it is used in another object, it cannot be deleted. Then False will be returned, otherwise True.

	Data Type	Explanation
Parameter: ⇒ layerName	System.String	Layer name
Return value	System.Boolean	Layer deleted (True)/not deleted (False)

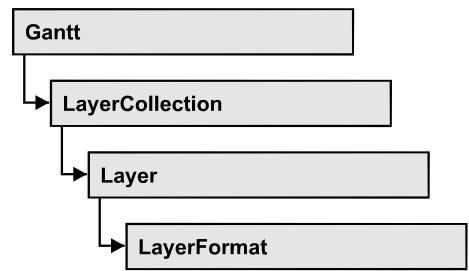
Update

Method of VcLayerCollection

This method lets you update a layer collection after having modified it.

	Data Type	Explanation
Return value	System.Boolean	Update successful (True)/ not successful (False)

7.45 VcLayerFormat



A layer format specifies the annotation of layers. With **For Each formatfield In LayerFormat** you can retrieve all layers.

Properties

- FormatField
- FormatFieldCount

Methods

- CopyFormatField
- GetEnumerator
- RemoveFormatField

Properties

FormatField

Read Only Property of VcLayerFormat

This property gives access to a VcLayerFormatField object by its index. The index has to be in the range from 0 to FormatFieldCount-1.

The property FormatField is an Indexed Property, which in C# is addressed by the methods set_FormatField (index, pvn) and get_FormatField (index) .

	Data Type	Explanation
Parameter: index	System.Int16	Index of the layer format field 0 ... FormatFieldCount-1
Property value	VcLayerFormatField	Layer format field

FormatFieldCount

Read Only Property of VcLayerFormat

This property gives access to the number of fields in a layer format.

	Data Type	Explanation
Property value	System.Int16	Number of fields of the layer format

Methods

CopyFormatField

Method of VcLayerFormat

This method allows to copy a layer format field. The new VcLayerFormatField object is returned. It is given automatically the next index not used before.

	Data Type	Explanation
Parameter: ⇨ position	VcFormatFieldPosition Possible Values: .vcAbove 1 .vcBelow 3 .vcLeftOf 0 .vcOutsideAbove 9 .vcOutsideBelow 11 .vcOutsideLeftOf 8 .vcOutsideRightOf 12 .vcRightOf 4	Position of the new layer format field above below left of outside, above outside, below outside, left of outside, right of right of
⇨ refIndex	System.Int16	Index of the reference layer format field
Return value	VcLayerFormatField	Layer format field object

GetEnumerator

Method of VcLayerFormat

This method returns an Enumerator object. It is implied in the For...Each construct of Visual Basic and C#. It supports the iteration by language specific elements. This object allows to iterate over the layer objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

RemoveFormatField

Method of VcLayerFormat

This method lets you remove a layer format field by its index. After that, the program will set all layer format field indexes newly in order to number them consecutively.

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the layer format field to be deleted

7.46 VcLayerFormatField

An object of the type **VcLayerFormatField** represents a field of a VcLayerFormat-Object. A layer format field does not have a name, as many other objects do, but it has an index that defines its position in the layer format.

Properties

- Alignment
- BottomMargin
- ConstantText
- FormatName
- Index
- LeftMargin
- MinimumWidth
- Priority
- RightMargin
- TextDataFieldIndex
- TextFont
- TextFontColor
- TextFontColorDataFieldIndex
- TextFontColorMapName
- TextFontDataFieldIndex
- TextFontMapName
- TextLineCount
- TextLineCountDataFieldIndex
- TextLineCountMapName
- TopMargin
- TruncatedTextSuppressed

Methods

- CalculateLineCount

Properties

Alignment

Property of VcLayerFormatField

This property lets you set or retrieve the alignment of the content of the layer format field.

	Data Type	Explanation
Property value	VcFormatFieldAlignment Possible Values: .vcFFABottom 28 .vcFFABottomLeft 27 .vcFFABottomRight 29 .vcFFACenter 25 .vcFFALeft 24 .vcFFARight 26 .vcFFATop 22 .vcFFATopLeft 21 .vcFFATopRight 23	Alignment of the field content Bottom Bottom left Bottom right Center Left Right Top Top left Top right

BottomMargin

Property of VcLayerFormatField

This property lets you set or retrieve the width (in mm) of the bottom margin of the layer format field. It can also be set in the **Edit Layer Format** dialog box.

	Data Type	Explanation

ConstantText

Property of VcLayerFormatField

This property allows the layer format field to display a constant text, if the property **TextDataFieldIndex** was set to **-1**.

	Data Type	Explanation
Property value	System.String	Constant text

FormatName

Read Only Property of VcLayerFormatField

This property lets you retrieve the name of the layer format to which this field belongs.

	Data Type	Explanation
Property value	System.String	Name of the layer format

Index

Read Only Property of VcLayerFormatField

This property lets you retrieve the index of the layer format field in the associated layer format.

	Data Type	Explanation
Property value	System.Int16	Index of the layer format field

LeftMargin

Property of VcLayerFormatField

This property lets you set or retrieve the width (in mm) of the left margin of the layer format field. It can also be set in the **Edit Layer Format** dialog box.

	Data Type	Explanation

MinimumWidth

Property of VcLayerFormatField

This property lets you set or retrieve the minimum width of the layer format field in mm if the label size dependence allows it.

	Data Type	Explanation
Property value	System.Int16	Minimum width (in mm) of the layer format field 0 ... 99

Priority

Property of VcLayerFormatField

This property lets you set or enquire the priority of the layer format field. By the priority you can influence the allocation of the available space in the field. The higher the priority, the greater the probability to get the space necessary.

	Data Type	Explanation
Property value	System.Int16	priority of the layer format field {-9...9}

RightMargin

Property of VcLayerFormatField

This property lets you set or retrieve the width (in mm) of the right margin of the layer format field. It can also be set in the **Edit Layer Format** dialog box.

	Data Type	Explanation

TextDataFieldIndex

Property of VcLayerFormatField

This property lets you set or retrieve the index of the data field, the content of which is to be displayed in the layer format field. If the value of the index equals **-1**, the content of the property **ConstantText** will be returned instead.

	Data Type	Explanation
Property value	System.Int16	Index of the data field

TextFont

Property of VcLayerFormatField

This property lets you set or retrieve the font of the layer format field. If in the property **TextFontMapName** a map was set, the map will control the text font color in dependence of the data.

	Data Type	Explanation
Property value	System.Drawing.Font	Font type of the layer format

TextFontColor

Property of VcLayerFormatField

This property lets you set or retrieve the font color of the layer format field. If a map was set by the property **TextFontMapName**, the map will control the text font color in dependence of the data.

	Data Type	Explanation
Property value	System.Drawing.Color	Font color of the layer format Default value: -1

TextFontColorDataFieldIndex

Property of VcLayerFormatField

This property lets you set or retrieve the data field index to be used with a font color map specified by the property **TextFontColorMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

TextFontColorMapName

Property of VcLayerFormatField

This property lets you set or retrieve the name of a color map (type vcColorMap) for the font color, if the format field is of the type **vcFFTText**. If the name of the color map is set to "", no map will be used. If a map name

and a data field index are specified by the property **TextFontColorDataFieldIndex**, the font color will be selected from the map. If no map entry applies, the font color specified by the property **TextFontColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the font color map

TextFontDataFieldIndex

Property of VcLayerFormatField

This property lets you set or retrieve the data field index to be used with a font map specified by the property **TextFontMapName**. If you set this property to -1, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

TextFontMapName

Property of VcLayerFormatField

This property lets you set or retrieve the name of a font map (type `vcFontMap`). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **TextFontDataFieldIndex**, the font will be selected from the map. If no data field entry applies, the font that is specified by the property **TextFont** will be used.

	Data Type	Explanation
Property value	System.String	Name of the font map

TextLineCount

Property of VcLayerFormatField

This property lets you enquire or set the line count, if the label size dependence allows it

	Data Type	Explanation
Property value	System.Int16	Number of lines

TextLineCountDataFieldIndex

Property of VcLayerFormatField

This property lets you set or retrieve the data field index to be used together with a font map specified by the property **TextLineCountMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

TextLineCountMapName

Property of VcLayerFormatField

This property lets you set or retrieve the name of a numeric map for the number of text lines. If set to "", no map will be used. If a map name and additionally a data field index is specified by the property **TextLineCountDataFieldIndex**, the corresponding number of text lines will be selected from the map. If no data field entry applies, the number of text lines specified by the property **TextLineCount** will be used.

	Data Type	Explanation
Property value	System.String	Name of the numeric map

TopMargin

Property of VcLayerFormatField

This property lets you set or retrieve the width (in mm) of the top margin of the layer format field. It can also be set in the **Edit Layer Format** dialog box.

	Data Type	Explanation

TruncatedTextSuppressed

Property of VcLayerFormatField

This property lets you set or retrieve, whether text which does not fit completely in the layer format field is to be suppressed or clipped.

	Data Type	Explanation
Property value	System.Boolean	Property active (True)/ not active (False)

Methods

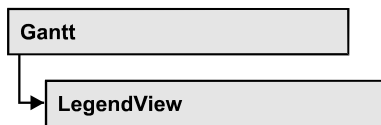
CalculateLineCount

Method of VcLayerFormatField

For outside fields of a layer only: This method calculates the number of text lines in the layer format field of the designated node, considering the current sizes of the layer and of the font. If inside fields are passed, -1 will be returned. The result of the method can be stored to a data field of the node to control the number of lines displayed (See dialog **Edit layer format -> Line count**).

	Data Type	Explanation
Parameter: ⇒ node	VcNode	Node
Return value	System.Int32	Calculated number of lines

7.47 VcLegendView



An object of the type **VcLegendView** designates the legend view window.

Properties

- Border
- BorderColor
- Height
- HeightActualValue
- Left
- LeftActualValue
- ScrollBarMode
- Top
- TopActualValue
- Visible
- Width
- WidthActualValue
- WindowMode

Methods

- Update

Properties

Border

Property of VcLegendView

This property lets you set or retrieve whether the world view has a frame (not in **vcPopupWindow** mode). The color of the frame is **Color.Black**. This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Boolean	Legend view with a border line (True)/without border line (False) Default value: True

Example Code VB.NET

```
VcGantt1.LegendView.Mode = VcLegendViewMode.vcNotFixed
VcGantt1.LegendView.Border = True
```

Example Code C#

```
vcGantt1.LegendView.Mode = VcLegendViewMode.vcNotFixed;
vcGantt1.LegendView.Border = true;
```

BorderColor

Property of VcLegendView

This property lets you set/retrieve the color of the frame that may be visible.

	Data Type	Explanation
Property value	System.Drawing.Color RGB ({0...255},{0...255},{0...255})	RGB color values ({0...255},{0...255},{0...255}) Default value: 0,0,0

Height

Property of VcLegendView

This property lets you retrieve the vertical extension of the legend view. It can also be set in the modes **vcFixedAtTop** and **vcFixedAtBottom**.

This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Int32	Height of the legend view Default value: 100

Example Code VB.NET

```
VcGantt1.LegendView.Height = 100
```

Example Code C#

```
vcGantt1.LegendView.Height = 100;
```

HeightActualValue

Read Only Property of VcLegendView

This property lets you retrieve the vertical extension of the legend view which actually is displayed. In the modes **b!vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** the actual value may differ from the one that was set because in these modes either the height or the width is preset.

	Data Type	Explanation
Property value	System.Int32	Actual height of the legend view {0, ...}

Example Code VB.NET

```
VcGantt1.LegendView.Height = 300
```

Example Code C#

```
vcGantt1.LegendView.Height = 100;
```

Left

Property of VcLegendView

This property lets you retrieve the left position of the world view. It can also be set in the modes **vcNotFixed** and **vcPopupWindow**.

This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Int32	Left position of the legend view Default value: 0

Example Code VB.NET

```
VcGantt1.LegendView.Left = 200
```

Example Code C#

```
vcGantt1.LegendView.Left = 200;
```

LeftActualValue

Read Only Property of VcLegendView

This property lets you retrieve the left position of the legend view which actually is displayed. In the modes **b!vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** the actual value may differ from the one that was set because in these modes either the height or the width is preset.

	Data Type	Explanation
Property value	System.Int32	Actual left position of the legend view {0, ...}

Example Code VB.NET

```
VcGantt1.LegendView.LeftActualValue = 150
```

Example Code C#

```
vcGantt1.LegendView.LeftActualValue = 150;
```

ScrollBarMode

Property of VcLegendView

This property lets you set or retrieve the scroll bar mode of the world view. This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	VcLegendViewScrollBarMode	Scrollbarmode Default value: NoScrollBar
	Possible Values: .vcAutomaticScrollBar 3 .vcHorizontalScrollBar 1 .vcNoScrollBar 0 .vcVerticalScrollBar 2	Display of a horizontal or vertical scrollbar if required. Display of a horizontal scrollbar if required. The chart is always displayed completely without scrollbars. Display of a vertical scrollbar if required.

Example Code VB.NET

```
VcGantt1.LegendView.ScrollBarMode = vcAutomaticScrollbar
```

Example Code C#

```
vcGantt1.LegendView.ScrollBarMode = vcAutomaticScrollBar;
```

Top

Property of VcLegendView

This property lets you retrieve the top position of the legend view. It can also be set in the modes **vcNotFixed** and **vcPopupWindow**.

This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Int32	Top position of the legend view Default value: 0

Example Code VB.NET

```
VcGantt1.LegendView.Top = 20
```

Example Code C#

```
vcGantt1.LegendView.Top = 20;
```

TopActualValue

Read Only Property of VcLegendView

This property lets you enquire the top position of the legend view which actually is displayed. In the modes **b!vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** the actual value may differ from the one that was set because in these modes either the height or the width is preset.

	Data Type	Explanation
Property value	System.Int32	Actual top position of the legend view {0, ...}

Example Code VB.NET

```
VcGantt1.LegendView.TopActualValue = 40
```

Example Code C#

```
vcGantt1.LegendView.TopActualValue = 40;
```

Visible

Property of VcLegendView

This property lets you enquire/set whether the legend view is visible or not. This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Boolean	Legend view visible (True)/not visible (False) Default value: False

Example Code VB.NET

```
VcGantt1.LegendView.Visible = True
```

Example Code C#

```
vcGantt1.LegendView.Visible = true;
```

Width

Property of VcLegendView

This property lets you retrieve the horizontal extent of the world view. It can also be set in the modes **vcFixedAtLeft**, **vcFixedAtRight**, **vcNotFixed** and **vcPopupWindow**.

This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Int32	Horizontal extension of the legend view Default value: 100

Example Code VB.NET

```
VcGantt1.LegendView.Width = 200
```

Example Code C#

```
vcGantt1.LegendView.Width = 200;
```

WidthActualValue

Read Only Property of VcLegendView

This property lets you retrieve the horizontal extent of the world view which actually is displayed. In the modes **b!vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** the actual value

may differ from the one that was set because in these modes either the height or width is preset.

	Data Type	Explanation
Property value	System.Int32	Actual horizontal extension of the legend view {0, ...}

Example Code VB.NET

```
VcGantt1.LegendView.WidthActualValue = 600
```

Example Code C#

```
vcGantt1.LegendView.WidthActualValue = 600;
```

WindowMode

Property of VcLegendView

This property lets you set or retrieve the legend view mode. This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	VcLegendViewWindowMode	Mode of the legend view Default value: vcPopupWindow
	Possible Values: .vcFixedAtBottom 4	The Legend view is displayed on the bottom of the VARCHART .NET control window. Then the height can be specified, whereas the position and the width are fixed.
	.vcFixedAtLeft 1	The Legend view is displayed on the left side of the VARCHART .NET control window. Then the width can be specified, whereas the position and the height are fixed.
	.vcFixedAtRight 2	The Legend view is displayed on the right side of the VARCHART .NET control window. Then the width can be specified, whereas the position and the height are fixed.
	.vcFixedAtTop 3	The Legend view is displayed on the top of the VARCHART .NET control window. Then the height can be specified, whereas the position and the width are fixed.
	.vcPopupWindow 6	The Legend view is a popup window with its own frame. The user can modify its position and extension, open it via the default context menu, and close it via the Close button in the frame.

Example Code VB.NET

```
VcGantt1.LegendView.Mode = VcLegendViewMode.vcNotFixed
```


Example Code C#

```
vcGantt1.LegendView.Mode = VcLegendViewMode.vcNotFixed;
```

Methods

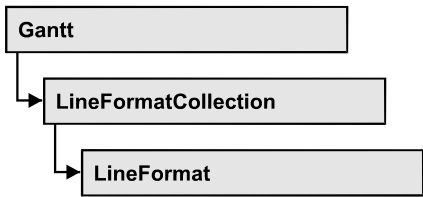
Update

Method of VcLegendView

This method lets you update the legend.

	Data Type	Explanation

7.48 VcLineFormat



An object of the type VcLineFormat defines the contents and the appearance of lines, for example in a date line grid.

Properties

- FormatField
- FormatFieldCount
- Name
- Specification

Methods

- CopyFormatField
- RemoveFormatField

Properties

FormatField

Read Only Property of VcLineFormat

This property lets you retrieve a VcLineFormatField object by its index. The index has to be in the range from 0 to FormatFieldCount-1.

	Data Type	Explanation
Parameter: index	System.Int16	Index of the line format field
Property value	VcNodeFormatField	Line format field

FormatFieldCount

Read Only Property of VcLineFormat

This property allows to determine the number of fields in a line format.

	Data Type	Explanation
Property value	System.Int16	Number of fields of the line format

Example Code VB.NET

```
Dim lineFormat As VcLineFormat

lineFormat = VcGantt1.LineFormatCollection.FirstFormat
MsgBox(lineFormat.FormatFieldCount)
```

Example Code C#

```
VcLineFormat nodeFormat = vcGantt1.LineFormatCollection.FirstFormat();
MessageBox.Show(lineFormat.FormatFieldCount.ToString());
```

Name

Property of VcLineFormat

This property lets you set or retrieve the name of the line format.

	Data Type	Explanation
Property value	System.String	Name of the line format

Example Code VB.NET

```
Dim lineFormat As VcLineFormat

lineFormat = VcGantt1.LineFormatCollection.FirstFormat
MsgBox(lineFormat.Name)
```

Example Code C#

```
VcLineFormat lineFormat = vcGantt1.LineFormatCollection.FirstFormat();
MessageBox.Show(lineFormat.Name);
```

Specification

Read Only Property of VcLineFormat

This property lets you retrieve the specification of a line format. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can smoothly be stored to text files or data bases. This allows for persistency. A specification can be used to create a line format by the method **VcLineFormatCollection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the line format

Methods

CopyFormatField

Method of VcLineFormat

This method allows to copy a line format field, returning the new VcLineFormatField object. It contains the next consecutive unused index.

	Data Type	Explanation
Parameter: ⇒ position	VcFormatFieldPosition Possible Values: .vcAbove 1 .vcBelow 3 .vcLeftOf 0 .vcOutsideAbove 9 .vcOutsideBelow 11 .vcOutsideLeftOf 8 .vcOutsideRightOf 12 .vcRightOf 4	Position of the new line format field above below left of outside, above outside, below outside, left of outside, right of right of
⇒ refIndex	System.Int16	Index of the reference line format field
Return value	VcNodeFormatField	Line format field object

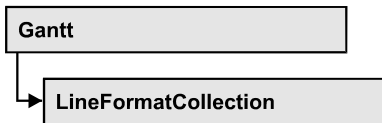
RemoveFormatField

Method of VcLineFormat

This method lets you remove a line format field by its index. After that, the program will re-set all node format field indexes in order to number them consecutively.

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the line format field to be deleted

7.49 VcLineFormatCollection



An object of the type VcLineFormatCollection automatically contains all line formats available to lines. You can access all objects in an iterative loop by **For Each lineFormat In LineFormatCollection** or by the methods **First...** and **Next...**. You can access a single line format by the methods **FormatByName** and **FormatByIndex**. The number of lines in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the line formats in the corresponding way.

Properties

- Count

Methods

- Add
- AddBySpecification
- Copy
- FirstFormat
- FormatByIndex
- FormatByName
- NextFormat
- Remove

Properties

Count

Read Only Property of VcLineFormatCollection

This property lets you retrieve the number of line formats in the line format collection.

	Data Type	Explanation
Property value	System.Int32	Number of line formats

Example Code VB.NET

```
Dim lineFormatCltn As VcLineFormatCollection
Dim numberOfLineformats As Integer

lineFormatCltn = VcGantt1.LineFormatCollection
numberOfLineformats = lineFormatCltn.Count
```

Example Code C#

```
VcLineFormatCollection lineFormatCltn = vcGantt1.LineFormatCollection;
int numberOfLineformats = lineFormatCltn.Count;
```

Methods

Add

Method of VcLineFormatCollection

By this method you can create a line format as a member of the LineFormatCollection. If the name has not been used before, the new line object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ formatName	System.String	Line format name
Return value	VcLineFormat	New line format object

Example Code VB.NET

```
Dim newLineFormat = VcGantt1.LineFormatCollection.Add("lineFormat1")
```

Example Code C#

```
newLineFormat = vcGantt1.LineFormatCollection.Add("lineFormat1");
```

AddBySpecification

Method of VcLineFormatCollection

This method lets you create a line format by using a line format specification. This way of creating allows line format objects to become persistent. The specification of a line format can be saved and re-loaded (see VcLineFormat property **Specification**). In a subsequent session the line format can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter: ⇒ formatSpecification	System.String	Line format specification
Return value	VcLineFormat	New line format object

Copy

Method of VcLineFormatCollection

By this method you can copy a line format. If the line format to be copied exists, and if the name for the new line format does not yet exist, the new line format object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ newFormatName	System.String	Name of the new line format
Return value	VcLineFormat	Line format object

Example Code VB.NET

```
Dim lineFormatCltn As VcLineFormatCollection
Dim lineFormat As VcLineFormat

lineFormatCltn = VcGantt1.LineFormatCollection
lineFormat = lineFormatCltn.Copy("CurrentLineFormat", "NewLineFormat")
```

Example Code C#

```
VcLineFormatCollection lineFormatCltn = vcGantt1.LineFormatCollection;
VcLineFormat lineFormat = lineFormatCltn.Copy("CurrentLineFormat",
"NewLineFormat");
```

FirstFormat

Method of VcLineFormatCollection

This method can be used to access the initial value, i.e. the first line format of a line format collection and then to continue in a forward iteration loop by the method **NextFormat** for the line formats following. If there is no line format in the line format collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcLineFormat	First line format

Example Code VB.NET

```
Dim format As VcLineFormat

format = VcGantt1.LineFormatCollection.FirstFormat
```

Example Code C#

```
VcLineFormat format = vcGantt1.LineFormatCollection.FirstFormat();
```

FormatByIndex

Method of VcLineFormatCollection

This method lets you access a line format by its index. If a line format does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcLineFormat	Line format object returned

Example Code VB.NET

```
Dim formatLineCltn As VcLineFormatCollection
Dim formatLine As VcLineFormat

formatLineCltn = VcGantt1.LineFormatCollection
formatLine = formatLineCltn.FormatByIndex(2)
```

Example Code C#

```
VcLineFormatCollection lineFormatCltn = vcGantt1.LineFormatCollection;
VcLineFormat format = lineFormatCltn.FormatByIndex(2);
```

FormatByName

Method of VcLineFormatCollection

By this method you can retrieve a line format by its name. If a line format of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ formatName	System.String	Name of the line format
Return value	VcLineFormat	Line format

Example Code VB.NET

```
Dim formatLineCltn As VcLineFormatCollection
Dim formatLine As VcLineFormat

formatLineCltn = VcGantt1.LineFormatCollection
formatLine = formatLineCltn.FormatByName("Standard")
```

Example Code C#

```
VcLineFormatCollection lineFormatCltn = vcGantt1.LineFormatCollection;
VcLineFormat format = lineFormatCltn.FormatByName("Standard");
```

NextFormat**Method of VcLineFormatCollection**

This method can be used in a forward iteration loop to retrieve subsequent line formats from a line format collection after initializing the loop by the method **FirstFormat**. If there is no format left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcLineFormat	Subsequent line format

Example Code VB.NET

```
Dim formatLineCltn As VcLineFormatCollection
Dim formatLine As VcLineFormat

formatLineCltn = VcGantt1.LineFormatCollection
formatLine = formatLineCltn.FirstFormat

While Not formatLine Is Nothing
    ListLine1.Items.Add(formatLine.Name)
    formatLine = formatLineCltn.NextFormat
End While
```

Example Code C#

```
VcLineFormatCollection lineFormatCltn = vcGantt1.LineFormatCollection;
VcLineFormat lineFormat = lineFormatCltn.FirstFormat();

while (lineFormat != null)
{
    ListLine.Items.Add(lineFormat.Name);
    lineFormat = lineFormatCltn.NextFormat();
}
```

Remove**Method of VcLineFormatCollection**

This method lets you delete a line format. If the line format is used by another object, it cannot be deleted. Then **False** will be returned, otherwise **True**.

	Data Type	Explanation
Parameter: ⇒ FormatName	System.String	Line format name
Return value	System.Boolean	Line format deleted (True) / not deleted (False)

Example Code VB.NET

```

Dim lineFormatCltn As VcLineFormatCollection
Dim lineFormat As VcLineFormat

lineFormatCltn = VcGantt1.LineFormatCollection
lineFormat = lineFormatCltn.FormatByIndex(1)
lineFormatCltn.Remove(lineFormat.Name)

```

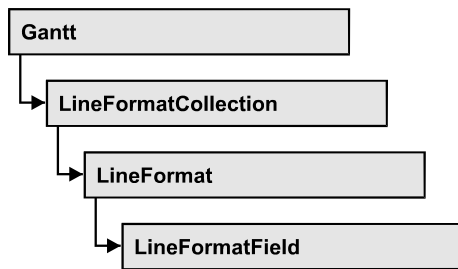
Example Code C#

```

VcLineFormatCollection lineFormatCltn = vcGantt1.LineFormatCollection;
VcLineFormat lineFormat = lineFormatCltn.FormatByIndex(1);
lineFormatCltn.Remove(lineFormat.Name);

```

7.50 VcLineFormatField



An object of the type `VcLineFormatField` represents a field of a `VcLineFormat`-Object. A line format field does not have a name as many other objects, but it has an index that defines its position in the line format.

Properties

- Alignment
- ConstantText
- DateOutputFormat
- FormatName
- Index
- PatternBackgroundColorAsARGB
- PatternBackgroundColorDataFieldIndex
- PatternBackgroundColorMapName
- PatternColorAsARGB
- PatternColorMapName
- PatternEx
- PatternExDataFieldIndex
- PatternExMapName
- TextDataFieldIndex
- TextFontColor
- TextFontColorDataFieldIndex
- TextFontColorMapName
- TextFontDataFieldIndex
- TextFontMapName
- TextLineCount

Properties

Alignment

Property of VcLineFormatField

This property lets you set or retrieve the alignment of the content of the line format field.

	Data Type	Explanation
Property value	VcFormatFieldAlignment	Alignment of the field content
	Possible Values: .vcFFABottom 28 .vcFFABottomLeft 27 .vcFFABottomRight 29 .vcFFACenter 25 .vcFFALeft 24 .vcFFARight 26 .vcFFATop 22 .vcFFATopLeft 21 .vcFFATopRight 23	Bottom Bottom left Bottom right Center Left Right Top Top left Top right

ConstantText

Property of VcLineFormatField

This property allows the line format field to display a constant text, if the line format field is of the type **vcFFTText** and if the property **TextDataField-Index** was set to -1.

	Data Type	Explanation
Property value	System.String	Constant text

DateOutputFormat

Property of VcLineFormatField

This property lets you set or retrieve the date output format. To compose the date you can use the below codes:

- D: first character of the day of the week (not adjustable)
- TD: Day of the Week (adjustable by using the event **VcTextEntrySupplying**)

DD:	two-digit figure for the day of the month: 01-31
DDD:	three initial characters of the day of the week (not adjustable)
M:	first character of the name of the month (not adjustable)
TM:	name of the month (adjustable by using the event VcTextEntrySupplying)
MM:	two-digit figure for the month: 01-12
MMM:	three initial characters of the name of the month (not adjustable)
YY:	two-digit figure for the year
YYYY:	four-digit figure for the year
WW:	two-digit figure for the number of the calendar week: 01-53
TW:	text for "calendar week" (adjustable by using the event VcTextEntrySupplying)
Q:	one-digit figure for the quarter: 1-4
TQ:	name of quarter (adjustable by using the event VcTextEntrySupplying)
hh	two-digit figure for the hour in 24 hours format: 00-23
HH:	two-digit figure for the hour in 12 hours format: 01-12
Th:	Text of "o' clock" (adjustable by using the event VcTextEntrySupplying)
TH:	"am" or "pm" (adjustable by using the event VcTextEntrySupplying)
mm	two-digit figure for the minute: 00-59
ss:	two-digit figure for the second: 00-59
TS:	short date format, as defined in the regional settings of the windows control panel
TL:	long date format, as defined in the regional settings of the windows control panel
TT:	time format, as defined in the regional settings of the windows control panel

Note: Characters which are not to be interpreted as part of the date should be preceded by a backslash '\'. '\\' for instance results in '\'. The special characters: ':, /, -' and **blank** don't need '\' as a prefix.

	Data Type	Explanation
Property value	System.String {DMYhms:;}/}	Date

Example Code VB.NET

```
VcGantt1.DateOutputFormat = "DD.MM.YY"
```

Example Code C#

```
vcGantt1.DateOutputFormat = "DD.MM.YY";
```

FormatName

Read Only Property of VcLineFormatField

This property lets you retrieve the name of the line format to which this field belongs.

	Data Type	Explanation
Property value	System.String	Name of the line format

Index

Read Only Property of VcLineFormatField

This property lets you retrieve the index of the line format field in the associated line format.

	Data Type	Explanation
Property value	System.Int16	Index of the line format field

PatternBackgroundColorAsARGB

Property of VcLineFormatField

This property lets you set or retrieve the background color of the line format field. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color.

If the line format field shall have the color of the line format, select the value -1.

If by the property **PatternBackgroundColorMapName** a map was specified, the map will set the background color in dependence on data.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255}) Default value: -1

PatternBackgroundColorDataFieldIndex

Property of VcLineFormatField

This property lets you set or retrieve the data field index to be used with a color map specified by the property **PatternBackgroundColorMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

PatternBackgroundColorMapName

Property of VcLineFormatField

This property lets you set or retrieve the name of a color map (type vcColorMap). If set to "", no map will be used. If you specify a map name and in addition set a data field index by the property **PatternBackgroundColorDataFieldIndex**, then the background color will be set by the map. If none of the map entries applies, the background color specified by the property **BackgroundColor** will apply.

	Data Type	Explanation
Property value	System.String	Name of the color map

PatternColorAsARGB

Property of VcLineFormatField

This property lets you set or retrieve the pattern color of the line format field. Color values have a transparency or alpha value, followed by a value for a

red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color. When casting an RGB value on an ARGB value, an alpha value of 255 has to be added.

If the box format field shall have the background color of the line format, select the value **-1**.

	Data Type	Explanation
Property value	System.Drawing.Color	Pattern color of the line format field

PatternColorMapName

Property of VcLineFormatField

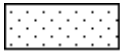
This property lets you set or retrieve the name of a color map (type `vcColorMap`). If set to "", no map will be used. Only if a map name and a data field index are specified in the property **PatternColorDataFieldIndex**, the pattern color is controlled by the map. If no data field entry applies, the pattern color of the calendar grid that is specified in the property **PatternColor** will be used.








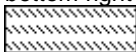
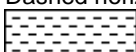



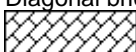
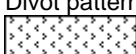

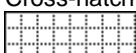
	Data Type	Explanation
Property value	System.String	Name of the color map




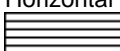
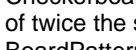


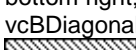
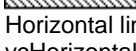
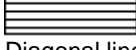
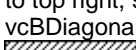
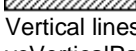


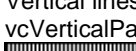
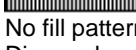


PatternEx

Property of VcLineFormatField

This property lets you set or retrieve the pattern of the field background of the line format field.

	Data Type	Explanation
Property value	VcFillPattern Possible Values: .vc05PercentPattern... vc90PercentPattern 01 - 11	Pattern type Dots in foreground color on background color, the density of the foreground color increasing with the percentage 

.vcAeroGlassPattern 44	Vertical color gradient in the color of the fill pattern 
.vcBDiagonalPattern 5	Diagonal lines slanting from bottom left to top right 
.vcCrossPattern 6	Cross-hatch pattern 
.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width 
.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 
.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width 
.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 
.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
.vcDashedHorizontalPattern 2026	Dashed horizontal lines 
.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 
.vcDashedVerticalPattern 2027	Dashed vertical lines 
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small 
.vcDiagonalBrickPattern 2032	Diagonal brick pattern 
.vcDivotPattern 2036	Divot pattern 
.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines 
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines 

.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right 
.vcHorizontalBrickPattern 2033	Horizontal brick pattern 
.vcHorizontalGradientPattern 52	Horizontal color gradient 
.vcHorizontalPattern 3	Horizontal lines 
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern 
.vcLargeConfettiPattern 2029	Confetti pattern, large 
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern 
.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern 
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern 
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern 
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern 
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern 
.vcNoPattern 1276	No fill pattern
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large 
.vcPlaidPattern 2035	Plaid pattern 
.vcShinglePattern 2039	Diagonal shingle pattern 
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern 
.vcSmallConfettiPattern 2028	Confetti pattern 
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern 

.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares
.vcSpherePattern 2041	Checkerboard of spheres
.vcTrellisPattern 2040	Trellis pattern
.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright
.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright
.vcVerticalGradientPattern 62	Vertical color gradient
.vcVerticalPattern 2	Vertical lines
.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark
.vcWavePattern 2031	Horizontal waves pattern
.vcWeavePattern 2034	Interwoven stripes pattern
.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern
.vcWideUpwardDiagonalPattern 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of vcBDiagonalPattern
.vcZigZagPattern 2030	Horizontal zig-zag lines

PatternExDataFieldIndex

Read Only Property of VcLineFormatField

This property lets you set or retrieve the data field index to be used together with the property **PatternExMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Index of the data field

PatternExMapName

Read Only Property of VcLineFormatField

This property lets you set or retrieve the name of a font map (type `vcPatternMap`). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **PatternExDataFieldIndex**, then the pattern is controlled by the map. If no data field entry applies, the pattern that is specified in the property **PatternEx** will be used.

	Data Type	Explanation
Property value	System.String	Name of the pattern map

TextDataFieldIndex

Property of VcLineFormatField

This property lets you set or retrieve the index of the data field, the content of which is to be displayed in the table format field. This property only works if the type of the data field is **vcFFTText**. If you set the value of the index to -1, the content of the property **ConstantText** will be returned instead.

	Data Type	Explanation
Property value	System.Int16	Index of the data field

TextFontColor

Property of VcLineFormatField

This property lets you set or retrieve the font color of the line format field, if it is of the type **vcFFTText**. If a map was set by the property **TextFontMapName**, the map will control the text font color in dependence on the data.

	Data Type	Explanation
Property value	System.Drawing.Color {True}	Font color of the line format

TextFontColorDataFieldIndex

Property of VcLineFormatField

This property lets you set or retrieve the data field index to be used with a font color map specified by the property **TextFontColorMapName**. If you set this property to -1, no map will be used.

	Data Type	Explanation
Property value	System.Int16 {True}	Data field index

TextFontColorMapName

Property of VcLineFormatField

This property lets you set or retrieve the name of a color map (type vcColorMap) for the font color, if the format field is of the type **vcFFTText**. If the name of the color map is set to "", no map will be used. If a map name and a data field index are specified by the property **TextFontColorDataFieldIndex**, the font color will be controlled by the map. If no map entry applies, the font color specified by the property **TextFontColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the font color map

TextFontDataFieldIndex

Property of VcLineFormatField

This property lets you set or retrieve the data field index to be used with a font map specified by the property **TextFontMapName**. If you set this property to -1, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

TextFontMapName

Property of VcLineFormatField

This property lets you set or retrieve the name of a font map (type `vcFontMap`). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **TextFontDataFieldIndex**, then the font is controlled by the map. If no data field entry applies, the font that is specified by the property **TextFont** will be used.

	Data Type	Explanation
Property value	System.String	Name of the font map

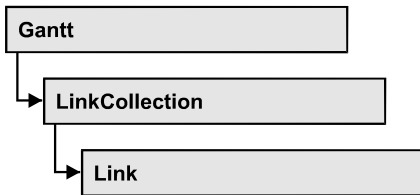
TextLineCount

Property of VcLineFormatField

This property lets you set or retrieve the number of lines, if the size of the annotation field allows for it

	Data Type	Explanation
Property value	System.Int16	Number of lines

7.51 VcLink



A VcLink object represents the logical and graphical link between two nodes. On the **Link** property page you can specify via a tick box **Show links** whether links should be displayed. Even if they are not displayed, they will be used for scheduling.

Properties

- AllData
- DataField
- ID
- PredecessorNode
- SuccessorNode

Methods

- DataRecord
- Delete
- RelatedDataRecord
- Update

Properties

AllData

Property of VcLink

This property lets you set or retrieve all data fields of a link. When setting the data, you can specify a CSV string (using semicolons as separators) or a data field. When retrieving the data, a character string will be returned. (See also **InsertLinkRecord**.)

	Data Type	Explanation
Property value	System.String	All data of the link

Example Code VB.NET

```

Dim linkCltn As VcLinkCollection
Dim link As VcLink
Dim allDataOfLink As String

linkCltn = VcGantt1.LinkCollection
link = linkCltn.FirstLink
allDataOfLink = link.AllData

```

Example Code C#

```

VcLinkCollection linkCltn = vcGantt1.LinkCollection;
VcLink link = linkCltn.FirstLink();
string allDataOfLink = link.AllData.ToString();

```

DataField**Property of VcLink**

This property lets you set or retrieve a specific data field of a link. The values which identify the predecessor and the successor nodes must not be changed.

The property DataField is an Indexed Property, which in C# is addressed by the methods set_DataField (index, pvn) and get_DataField (index).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the data field
Property value	System.Object	Content of data field

Example Code VB.NET

```

Dim linkCltn As VcLinkCollection
Dim link As VcLink
Dim message As String

linkCltn = VcGantt1.LinkCollection
For Each link In linkCltn
    message = "Delete link from " + link.DataField(1) + " to " +
link.DataField(2) + " ?"
    If MsgBox(message, MsgBoxStyle.OKCancel, "Delete Link") = MsgBoxResult.OK
Then
        link.Delete()
    End If
Next

```


Example Code C#

```
VcLinkCollection linkCltn = vcGantt1.LinkCollection;

foreach (VcLink link in linkCltn)
{
    DialogResult retVal = MessageBox.Show("Delete link from " +
link.get_DataField(1) + " to " + link.get_DataField(2) + " ?", "Deleting curve
point", MessageBoxButtons.OKCancel);
    if (retVal == DialogResult.OK)
        link.Delete();
}
```

ID**Read Only Property of VcLink**

By this property you can retrieve the ID of a link.

	Data Type	Explanation
Property value	System.String	Link ID

PredecessorNode**Read Only Property of VcLink**

This method lets you identify the predecessor node of a link.

	Data Type	Explanation
Property value	VcNode	Predecessor node

Example Code VB.NET

```
Dim linkCltn As VcLinkCollection
Dim link As VcLink
Dim node As VcNode
Dim nodeName As String

linkCltn = vcGantt1.LinkCollection
link = linkCltn.FirstLink
node = link.PredecessorNode
nodeName = node.DataField(1)
```

Example Code C#

```
VcLinkCollection linkCltn = vcGantt1.LinkCollection;
VcLink link = linkCltn.FirstLink();
VcNode node = link.PredecessorNode;
string nodeName = node.get_DataField(1).ToString();
```

SuccessorNode

Read Only Property of VcLink

This method lets you identify the successor node of a link.

	Data Type	Explanation
Property value	VcNode	Successor node

Example Code VB.NET

```
Dim linkCltn As VcLinkCollection
Dim link As VcLink
Dim node As VcNode
Dim nodeName As String

linkCltn = VcGantt1.LinkCollection
link = linkCltn.FirstLink
node = link.SuccessorNode
nodeName = node.DataField(1)
```

Example Code C#

```
VcLinkCollection linkCltn = vcGantt1.LinkCollection;
VcLink link = linkCltn.FirstLink();

VcNode node = link.SuccessorNode;
string nodeName = node.get_DataField(1).ToString();
```

Methods

DataRecord

Method of VcLink

This property lets you retrieve the link as a data record object. The properties of the data record object give access to the corresponding data table and the data table collection.

	Data Type	Explanation
Return value	VcDataRecord	Data record returned

Delete

Method of VcLink

By this method you can delete a link.

	Data Type	Explanation
Return value	System.Boolean	Link was/was not successfully deleted

Example Code VB.NET

```
Private Sub VcGantt1_VcLinksRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcLinksClickingEventArgs) Handles VcGantt1.VcLinksRightClicking
    Dim message As String
    message = "Delete link: " + e.LinkCollection.FirstLink.AllData

    If MsgBox(message, MsgBoxStyle.OKCancel, "Delete link") = MsgBoxResult.OK Then
        e.LinkCollection.FirstLink.Delete()
    End If
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup
End Sub
```

Example Code C#

```
private void vcGantt1_VcLinksRightClicking(object sender,
NETRONIC.XGantt.VcLinksClickingEventArgs e)
{
    string message = "Delete link: " + e.LinkCollection.FirstLink().AllData;
    DialogResult retVal = MessageBox.Show(message, "Deleting link",
    MessageBoxButtons.OKCancel);
    if (retVal == DialogResult.OK)
        e.LinkCollection.FirstLink().Delete();
    else
        e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

RelatedDataRecord**Method of VcLink**

This method lets you retrieve a data record from a data table that is related to the link data table. The index passed by the parameter denotes the field in the data record that holds the key of the related data record.

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of data field that holds the key
Return value	VcDataRecord	Related data record returned

Update**Method of VcLink**

When a data field of a link was edited by the **DataField** property, you can update the diagram by the **Update** method.

	Data Type	Explanation
Return value	System.Boolean	Link was/was not successfully updated

Example Code VB.NET

```

Dim linkCltn As VcLinkCollection
Dim link As VcLink

linkCltn = VcGantt1.LinkCollection
link = linkCltn.FirstLink
link.DataField(2) = 10
link.Update()

```

Example Code C#

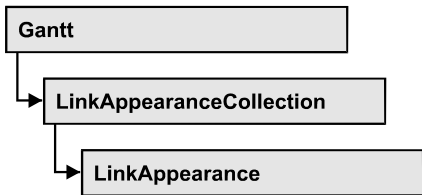
```

VcLinkCollection linkCltn = vcGantt1.LinkCollection;
VcLink link = linkCltn.FirstLink();

link.set_DataField(2, 10);
link.Update();

```

7.52 VcLinkAppearance



A VcLinkAppearance object defines the appearance of a link, if the node data comply with the conditions defined by the filters assigned. Different link appearances can be set on the **Link** property page in the table.

Properties

- FilterName
- LineColor
- LineThickness
- LineType
- Name
- PredecessorLayerName
- PredecessorPortSymbol
- RoutingType
- SuccessorLayerName
- SuccessorPortSymbol
- Visible

Methods

- PutInOrderAfter

Properties

FilterName

Read Only Property of VcLinkAppearance

This property lets you retrieve the filter that is used for a link appearance. This property can be also set on the **Link** property page.

	Data Type	Explanation
Property value	System.String	Filter name

Example Code VB.NET

```

Dim linkAppearanceCltn As VcLinkAppearanceCollection
Dim linkAppearance As VcLinkAppearance
Dim filterOfLinkApp As String

linkAppearanceCltn = VcGantt1.LinkAppearanceCollection
linkAppearance = linkAppearanceCltn.LinkAppearanceByName("Blue")
filterOfLinkApp = linkAppearance.FilterName

```

Example Code C#

```

VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkAppearance =
linkAppearanceCltn.LinkAppearanceByName("Blue");
string filterOfLinkApp = linkAppearance.FilterName;

```

LineColor

Property of VcLinkAppearance

This property lets you set or retrieve the line color of a LinkAppearance object.

This property can be also set in the **Line Attributes** dialog box that can be opened by the **Link** property page.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values

Example Code VB.NET

```

Dim linkAppearanceCltn As VcLinkAppearanceCollection
Dim linkAppearance As VcLinkAppearance

linkAppearanceCltn = VcGantt1.LinkAppearanceCollection
linkAppearance = linkAppearanceCltn.LinkAppearanceByName("Blue")
linkAppearance.LineColor = Color.Blue

```

Example Code C#

```

VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkAppearance =
linkAppearanceCltn.LinkAppearanceByName("Blue");
linkAppearance.LineColor = Color.LightSteelBlue;

```

LineThickness

Property of VcLinkAppearance

This property lets you set or retrieve the line thickness of a LinkAppearance object.

If you set this property to values between 1 and 4, an absolute line thickness is defined in pixels. Irrespective of the zoom factor a line will always show the same line thickness in pixels. When printing though, the line thickness is adapted for the sake of legibility and becomes dependent of the zoom factor:

Value	Points	mm
1	1/2 point	0.09 mm
2	1 point	0.18 mm
3	3/2 points	0.26 mm
4	2 points	0.35 mm

A point equals 1/72 inch and represents the unit of the font size.

If you set this property to values between 5 and 1,000, the line thickness is defined in 1/100 mm, so the lines will be displayed in a true thickness in pixels that depends on the zoom factor.

This property can be also set in the **Line Attributes** dialog box that can be opened by the **Link** property page.

	Data Type	Explanation
Property value	System.Int32	Line thickness LineType {1...4}: line thickness in pixels LineType {5...1000}: line thickness in 1/100 mm Default value: As defined on property page

Example Code VB.NET

```
Dim linkAppearanceCltn As VcLinkAppearanceCollection
Dim linkAppearance As VcLinkAppearance

linkAppearanceCltn = VcGantt1.LinkAppearanceCollection
linkAppearance = linkAppearanceCltn.LinkAppearanceByName("Standard")
linkAppearance.LineThickness = 4
```

Example Code C#

```
VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkAppearance =
linkAppearanceCltn.LinkAppearanceByName("Standard");
linkAppearance.LineThickness = 4;
```

LineType

Property of VcLinkAppearance

This property lets you set or retrieve the line type of a LinkAppearance object. This property can be also be set in the **Line attributes** dialog, that you can get to by the **Link** property page.

Property value	Data Type	Explanation
	VcLineType	Line type Default value: vcSolid
	Possible Values:	
	.vcDashed 4	Line dashed
	.vcDashed 4	Line dashed
	.vcDashedDotted 5	Line dashed-dotted
	.vcDashedDotted 5	Line dashed-dotted
	.vcDotted 3	Line dotted
	.vcDotted 3	Line dotted
	.vcLineType0 100	Line Type 0
	.vcLineType1 101	Line Type 1
	.vcLineType10 110	Line Type 10
	.vcLineType11 111	Line Type 11
	.vcLineType12 112	Line Type 12
	.vcLineType13 113	Line Type 13
	.vcLineType14 114	Line Type 14
	.vcLineType15 115	Line Type 15
	.vcLineType16 116	Line Type 16
	.vcLineType17 117	Line Type 17
	.vcLineType18 118	Line Type 18
	.vcLineType2 102	Line Type 2
	.vcLineType3 103	Line Type 3
	.vcLineType4 104	Line Type 4
	.vcLineType5 105	Line Type 5
	.vcLineType6 106	Line Type 6
	.vcLineType7 107	Line Type 7
	.vcLineType8 108	Line Type 8
	.vcLineType9 109	Line Type 9
	.vcNone 1	No line type assigned
	.vcNone 1	No line type
	.vcSolid 2	Line solid

	.vcSolid 2	Line solid
--	------------	------------

Example Code VB.NET

```
Dim linkAppearanceCltn As VcLinkAppearanceCollection
Dim linkAppearance As VcLinkAppearance

linkAppearanceCltn = VcGantt1.LinkAppearanceCollection
linkAppearance = linkAppearanceCltn.LinkAppearanceByName("Blue")
linkAppearance.LineType = 5
```

Example Code C#

```
VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkAppearance =
linkAppearanceCltn.LinkAppearanceByName("Blue");
linkAppearance.LineType = VcLineType.vcLineType5;
```

Name

Read Only Property of VcLinkAppearance

This property lets you retrieve the name of a LinkAppearance object.

This property can also be set on the **Links** property page.

	Data Type	Explanation
Property value	System.String	Name of the link appearance

Example Code VB.NET

```
Dim linkAppearanceCltn As VcLinkAppearanceCollection
Dim linkAppearance As VcLinkAppearance
Dim nameLinkApp As String

linkAppearanceCltn = VcGantt1.LinkAppearanceCollection
linkAppearance = linkAppearanceCltn.FirstLinkAppearance
nameLinkApp = linkAppearance.Name
```

Example Code C#

```
VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkAppearance = linkAppearanceCltn.FirstLinkAppearance();
string nameLinkApp = linkAppearance.Name;
```

PredecessorLayerName

Property of VcLinkAppearance

This property lets you specify or retrieve to which layer of the predecessor node a link is to be drawn. If you enter "" (default), the link will be drawn to the first visible layer of this node.

This property can also be set on the **Links** property page.

	Data Type	Explanation
Property value	System.String	Character string that passes the layer name

PredecessorPortSymbol

Property of VcLinkAppearance

This property lets you assign/retrieve a port symbol to/from a link, that visually accentuates the junction of the link and the predecessor node.

This property can also be set on the **Links** property page.

	Data Type	Explanation
Property value	VcLinkPredecessorPortSymbol	Symbol on the predecessor node Default value: vcLPSNone

Example Code VB.NET

```
Dim linkAppearanceCltn As VcLinkAppearanceCollection
Dim linkAppearance As VcLinkAppearance

linkAppearanceCltn = VcGantt1.LinkAppearanceCollection
linkAppearance = linkAppearanceCltn.FirstLinkAppearance
linkAppearance.PredecessorPortSymbol =
VcLinkPredecessorPortSymbol.vcLPSDoubleSemiCircle
```

Example Code C#

```
VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkAppearance = linkAppearanceCltn.FirstLinkAppearance();
linkAppearance.PredecessorPortSymbol =
VcLinkPredecessorPortSymbol.vcLPSFilledDoubleSemiCircle;
```

RoutingType

Property of VcLinkAppearance

This property lets you set or retrieve, whether the links of the diagram should be drawn horizontally and vertically only (and therefore show orthogonal shapes), or if they are allowed to lead directly to their aim, probably on an oblique route, allowing to cut through objects.

This property can also be set on the **Links** property page.

	Data Type	Explanation
Property value	VcRoutingType	Routing type
	Possible Values: .vcLRTOrthogonal 1	Default value: vcLRTOrthogonal Links run horizontally and vertically only and show an orthogonal shape.
	.vcLRTOrthogonalDistinguishable 2	Links run horizontally and vertically only and show an orthogonal shape. By displaying bends with appropriate slants, the links can be better distinguished and their direction more easily tracked. The height of the chart will be increased according to the number of horizontal line parts which are generated.
	.vcLRTStraightLined 4	Links have a straight shape and may run obliquely.

SuccessorLayerName

Property of VcLinkAppearance

This property lets you specify or retrieve to which layer of the successor node a link is to be drawn. If you enter "" (default), the link will be drawn to the first visible layer of this node.

This property can also be set on the **Links** property page.

	Data Type	Explanation
Property value	System.String	Character string that passes the layer name

SuccessorPortSymbol

Property of VcLinkAppearance

This property lets you assign/retrieve a port symbol to a link, that accentuates the intersection of the link and the successor node.

This property can also be set on the **Links** property page.

	Data Type	Explanation
Property value	VcLinkSuccessorPortSymbol	Symbol on the successor node Default value: vcLSSNone

Example Code VB.NET

```
VcLinkAppearanceCollection linkAppearanceCltn =
VcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkAppearance = linkAppearanceCltn.FirstLinkAppearance();
linkAppearance.SuccessorPortSymbol =
VcLinkSuccessorPortSymbol.vcLSSFilledDoubleArrow;
```

Example Code C#

```
VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkAppearance = linkAppearanceCltn.FirstLinkAppearance();
linkAppearance.SuccessorPortSymbol =
VcLinkSuccessorPortSymbol.vcLSSFilledDoubleArrow;
```

Visible

Property of VcLinkAppearance

This property lets you set or retrieve whether the link is to be visible or not, taking no effect, however, on the phantom lines for links while dragging.

This property can also be set on the **Links** property page, but here also applying to the phantom lines.

	Data Type	Explanation
Property value	System.Boolean	Property active/not active Default value: True

Example Code VB.NET

```
Dim linkAppearanceCltn As VcLinkAppearanceCollection
Dim linkAppearance As VcLinkAppearance

linkAppearanceCltn = VcGantt1.LinkAppearanceCollection
linkAppearance = linkAppearanceCltn.FirstLinkAppearance
linkAppearance.Visible = False
```

Example Code C#

```
VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkAppearance = linkAppearanceCltn.FirstLinkAppearance();
linkAppearance.Visible = false;
```

Methods
PutInOrderAfter**Method of VcLinkAppearance**

This method lets you set the link appearance behind a link appearance specified by name, within the LinkAppearanceCollection. If you set the name to "", the link appearance will be put in the first position. The order of the link appearances within the collection determines the order by which they apply to the links.

	Data Type	Explanation
Parameter: refLinkAppearanceName	System.String	Name of the link appearance behind which the current link appearance is to be put.

Example Code VB.NET

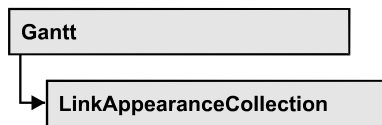
```
Dim linkAppCltn As VcLinkAppearanceCollection
Dim linkApp1 As VcLinkAppearance
Dim linkApp2 As VcLinkAppearance

linkAppCltn = vcGantt1.LinkAppearanceCollection()
linkApp1 = linkAppCltn.Add("linkApp1")
linkApp2 = linkAppCltn.Add("linkApp2")
linkApp1.PutInOrderAfter("linkApp2")
linkAppCltn.Update()
```

Example Code C#

```
VcLinkAppearanceCollection linkAppCltn = vcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkApp1 = linkAppCltn.Add("linkApp1");
VcLinkAppearance linkApp2 = linkAppCltn.Add("linkApp2");
linkApp1.PutInOrderAfter("linkApp2");
linkAppCltn.Update();
```

7.53 VcLinkAppearanceCollection



An object of the type `VcLinkAppearanceCollection` automatically contains all available link appearances. You can access all objects in an iterative loop by **For Each linkAppearance In LinkAppearanceCollection** or by the methods **First...** and **Next...**. You can access a single line format by the methods **LinkAppearanceByName** and **LinkAppearanceByIndex**. The number of link appearances in the collection object can be retrieved by the property **Count**.

Properties

- Count

Methods

- Add
- AddBySpecification
- Copy
- FirstLinkAppearance
- GetEnumerator
- LinkAppearanceByIndex
- LinkAppearanceByName
- NextLinkAppearance
- Remove
- Update

Properties

Count

Read Only Property of VcLinkAppearanceCollection

This property lets you retrieve the number of link appearances in the `LinkAppearanceCollection` object.

1170 API Reference: VcLinkAppearanceCollection

	Data Type	Explanation
Property value	System.Int32	Number of link appearance objects

Example Code VB.NET

```
Dim linkAppearanceCltn As VcLinkAppearanceCollection
Dim numberOfLinkAppearance As Integer

linkAppearanceCltn = VcGantt1.LinkAppearanceCollection
numberOfLinkAppearance = linkAppearanceCltn.Count
```

Example Code C#

```
VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
int numberOfLinkAppearance = linkAppearanceCltn.Count;
```

Methods

Add

Method of VcLinkAppearanceCollection

By this method you can create a new link appearance as a member of the LinkAppearanceCollection. If the name was not used before, the new link appearance object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned. All attributes of the new link appearance by default are set to transparent.

	Data Type	Explanation
Parameter: ⇒ newName	System.String	Link appearance name
Return value	VcLinkAppearance	New link appearance object

Example Code VB.NET

```
newLinkAppearance = VcGantt1.LinkAppearanceCollection.Add("linkapp1")
```

Example Code C#

```
newLinkAppearance = vcGantt1.LinkAppearanceCollection.Add("linkapp1");
```

AddBySpecification

Method of VcLinkAppearanceCollection

This method lets you create a link appearance by using a link appearance specification. This way of creating allows link appearance objects to become persistent. The specification of a link appearance can be saved and re-loaded (see VcLinkAppearance property **Specification**). In a subsequent session the link appearance can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter:		
⇒ linkAppearanceSpecification	System.String	Link appearance specification
Return value	VcLinkAppearance	New link appearance object

Copy

Method of VcLinkAppearanceCollection

By this method you can copy a link appearance. When the link appearance has come into existence and if the name for the new link appearance did not yet exist, the new link appearance object will be returned. Otherwise "Nothing" (Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter:		
⇒ fromName	System.String	Name of the link appearance to be copied
⇒ newName	System.String	Name of the new link appearance
Return value	VcLinkAppearance	Link appearance object

FirstLinkAppearance

Method of VcLinkAppearanceCollection

This method can be used to access the initial value, i.e. the first link appearance of a link appearance collection and then to continue in a forward iteration loop by the method **NextLinkAppearance** for the link appearances following. If there is no link appearance in the link appearance collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcLinkAppearance	First linkAppearance object

Example Code VB.NET

```
Dim linkAppearanceCltn As VcLinkAppearanceCollection
Dim linkAppearance As VcLinkAppearance

linkAppearanceCltn = VcGantt1.LinkAppearanceCollection
linkAppearance = linkAppearanceCltn.FirstLinkAppearance
```

Example Code C#

```
VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkAppearance = linkAppearanceCltn.FirstLinkAppearance();
```

GetEnumerator**Method of VcLinkAppearanceCollection**

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the link appearance objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

LinkAppearanceByIndex**Method of VcLinkAppearanceCollection**

This method lets you access a link appearance object by its index. If a linkAppearance object does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the link appearance object
Return value	VcLinkAppearance	LinkAppearance object returned

LinkAppearanceByName

Method of VcLinkAppearanceCollection

This method retrieves a link appearance object by its name.

	Data Type	Explanation
Parameter: ⇒ linkAppearanceName	System.String	Name of the link appearance object
Return value	VcLinkAppearance	LinkAppearance object

Example Code VB.NET

```
Dim linkAppearanceCltn As VcLinkAppearanceCollection
Dim linkAppearance As VcLinkAppearance
linkAppearanceCltn = VcGantt1.LinkAppearanceCollection
linkAppearance = linkAppearanceCltn.LinkAppearanceByName("Standard")
```

Example Code C#

```
VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkAppearance =
linkAppearanceCltn.LinkAppearanceByName("Standard");
```

NextLinkAppearance

Method of VcLinkAppearanceCollection

This method can be used in a forward iteration loop to retrieve subsequent link appearances from a link appearance collection after initializing the loop by the method **FirstLinkAppearance**. If there is no link appearance left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcLinkAppearance	Succeeding linkAppearance object

Example Code VB.NET

```
Dim linkAppearanceCltn As VcLinkAppearanceCollection
Dim linkAppearance As VcLinkAppearance

linkAppearanceCltn = VcGantt1.LinkAppearanceCollection
linkAppearance = linkAppearanceCltn.FirstLinkAppearance
While Not linkAppearance Is Nothing
    linkAppearance.Visible = False
    ListBox1.Items.Add("Name: " + linkAppearance.Name)
    linkAppearance = linkAppearanceCltn.NextLinkAppearance
End While
```

Example Code C#

```

VcLinkAppearanceCollection linkAppearanceCltn =
vcGantt1.LinkAppearanceCollection;
VcLinkAppearance linkAppearance = linkAppearanceCltn.FirstLinkAppearance();
while (linkAppearance != null)
{
    linkAppearance.Visible = false;
    listBox1.Items.Add("Name: " + linkAppearance.Name);
    linkAppearance = linkAppearanceCltn.NextLinkAppearance();
}

```

Remove**Method of VcLinkAppearanceCollection**

This method lets you delete a link appearance. If the link appearance is being used in a different object, it cannot be deleted. Then **False** will be returned, otherwise **True**.

	Data Type	Explanation
Parameter:		
⇒ name	System.String	Name of the link appearance
Return value	System.Boolean	Link appearance deleted (True)/not deleted (False)

Update**Method of VcLinkAppearanceCollection**

This method lets you update a link appearance collection after having modified it.

	Data Type	Explanation
Return value	System.Boolean	Link appearance collection was/was not successfully updated

Example Code VB.NET

```

Dim linkAppearanceCltn As VcLinkAppearanceCollection
Dim linkAppearance As VcLinkAppearance

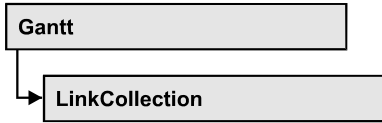
linkAppearanceCltn = vcGantt1.LinkAppearanceCollection
linkAppearance = linkAppearanceCltn.LinkAppearanceByIndex(0)
linkAppearanceCltn.Remove(linkAppearance.Name)
linkAppearanceCltn.Update()

```

Example Code C#

```
VcLinkAppearanceCollection linkAppearanceCltn =  
vcGantt1.LinkAppearanceCollection;  
VcLinkAppearance linkAppearance = linkAppearanceCltn.LinkAppearanceByIndex(0);  
linkAppearanceCltn.Remove(linkAppearance.Name);  
linkAppearanceCltn.Update();
```

7.54 VcLinkCollection



An object of the type VcLinkCollection contains all available links. You can access all objects in an iterative loop by **For Each link In LinkCollection** or by the methods **First...** and **Next...**. The number of links in the collection object can be retrieved by the property **Count**.

Properties

- Count

Methods

- FirstLink
- GetEnumerator
- NextLink
- SelectLinks

Properties

Count

Read Only Property of VcLinkCollection

This property lets you retrieve the number of links in the link collection.

	Data Type	Explanation
Property value	System.Int32	Number of links

Example Code VB.NET

```
Dim linkCltn As VcLinkCollection
Dim numberOfLinks As Integer

linkCltn = VcGantt1.LinkCollection
numberOfLinks = linkCltn.Count
```

Example Code C#

```
VcLinkCollection linkCltn = vcGantt1.LinkCollection;
int numberOfLinks = linkCltn.Count;
```

Methods

FirstLink

Method of VcLinkCollection

This method can be used to access the initial value, i.e. the first link of a link collection, and to continue in a forward iteration loop by the method **NextLink** for the links following. If there is no link in the link collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcLink	First link

Example Code VB.NET

```
Dim linkCltn As VcLinkCollection
Dim link As VcLink

linkCltn = VcGantt1.LinkCollection
link = linkCltn.FirstLink
```

Example Code C#

```
VcLinkCollection linkCltn = vcGantt1.LinkCollection;
VcLink link = linkCltn.FirstLink();
```

GetEnumerator

Method of VcLinkCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the link objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

NextLink

Method of VcLinkCollection

This method can be used in a forward iteration loop to retrieve subsequent links from a link collection after initializing the loop by the method **FirstLink**. If there is no link left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcLink	Succeeding link

Example Code VB.NET

```

Dim linkCltn As VcLinkCollection
Dim link As VcLink

linkCltn = VcGantt1.LinkCollection
link = linkCltn.FirstLink
While Not link Is Nothing
    ListBox1.Items.Add(link.AllData)
    link = linkCltn.NextLink
End While

```

Example Code C#

```

VcLinkCollection linkCltn = vcGantt1.LinkCollection;
VcLink link = linkCltn.FirstLink();

while (link != null)
{
    listBox1.Items.Add(link.AllData);
    link = linkCltn.NextLink();
}

```

SelectLinks

Method of VcLinkCollection

This method lets you specify the links that the link collection is to contain.

	Data Type	Explanation
Parameter: ⇒ selectionType	VcSelectionType Possible Values: .vcAll 0 .vcAllLinksCausingCycles 7 .vcAllLinksInCycles 6 .vcAllVisible 1 .vcSelected 2	Links to be selected All objects in the diagram will be selected If this selection type is chosen, the link collection will contain all links that cause the existence of cycles. If these links are deleted, cycles will cease to exist in this chart. If this selection type is chosen, the link collection will contain all links that participate in forming cycles. Cycles are chains of nodes and links of which the beginning and end join. All visible objects will be selected All marked objects will be selected
Return value	System.Int32	Number of links selected

Example Code VB.NET

```

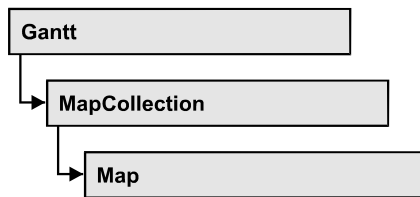
Dim linkCltn As VcLinkCollection
linkCltn = VcGantt1.LinkCollection
linkCltn.SelectGroups (vcAllMarked)

```

Example Code C#

```
VcLinkCollection linkCltn = vcGantt1.LinkCollection;  
linkCltn.SelectGroups (vcAllMarked);
```


7.55 VcMap



Maps define certain properties of nodes by data field entries, for example their background color which is based on the data of the node record.

In a map you can specify 150 map entries at maximum. By the call **For Each mapEntry In Map** you can retrieve all data field entries in an iterative loop.

Properties

- ConsiderFilterEntries
- Count
- GetEnumerator
- Name
- Specification
- Type

Methods

- CreateEntry
- DeleteEntry
- FirstMapEntry
- GetMapEntry
- NextMapEntry

Properties

ConsiderFilterEntries

Read Only Property of VcMap

This property lets you set/retrieve whether filters are considered when a map is assigned to data field entries so that ranges of values can also be specified as keys.

	Data Type	Explanation

Count

Read Only Property of VcMap

This property lets you retrieve the number of map entries in a map.

	Data Type	Explanation
Property value	System.Int32	Number of map entries

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim numberOfEntries As Integer

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
numberOfEntries = map.Count
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
int numberOfEntries = map.Count;
```

GetEnumerator

Read Only Property of VcMap

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the map entries included.

	Data Type	Explanation
Property value	VcObject	Reference object

Name

Read Only Property of VcMap

This property lets you retrieve the name of a map.

	Data Type	Explanation
Property value	System.String	Name of the map

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapName As String

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.FirstMap
mapName = map.Name
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.FirstMap();
string mapName = map.Name;
```

Specification**Read Only Property of VcMap**

This property lets you retrieve the specification of a map. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored without problems to text files or data bases. This allows for persistency. A specification can be used to create a map by the method **VcMapCollection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the map

Example Code VB.NET

```
Dim boxCltn As VcBoxCollection
Dim box As VcBox

boxCltn = VcGantt1.BoxCollection
box = boxCltn.FirstBox
MsgBox(box.Specification)
```

Example Code C#

```
VcBoxCollection boxCltn = vcGantt1.BoxCollection;
VcBox box = boxCltn.FirstBox();
MessageBox.Show(box.Specification);
```

Type

Property of VcMap

This property lets you enquire/set the map type.

	Data Type	Explanation
Property value	VcMapType Possible Values: .vcAnyMap 0 .vcColorMap 1 .vcFontMap 8 .vcGraphicsFileMap 7 .vcMillimeterMap 9 .vcNumberMap 10 .vcPatternMap 3 .vcTextMap 6	Map type any (used only for selecting) Colors Fonts Graphics file Millimeters Numbers Patterns Text

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
map.Type = VcMapType.vcPatternMap
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
map.Type = VcMapType.vcPatternMap;
```

Methods

CreateEntry

Method of VcMap

This method lets you create a new entry (a new row) for a map. To make the entry work, the method **MapCollection.Update()** should be invoked after creating.

	Data Type	Explanation
Return value	VcMapEntry	Map entry

Example Code VB.NET

```

Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.CreateEntry
mapCltn.Update

```

Example Code C#

```

VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.CreateEntry();
mapCltn.Update;

```

DeleteEntry**Method of VcMap**

This method lets you delete an entry (a row) of the map. To make the deletion work, the method **MapCollection.Update()** should be invoked after deleting.

	Data Type	Explanation
Parameter: ⇒ mapEntry	VcMapEntry	Map entry
Return value	System.Boolean	Map entry was/was not deleted successfully

Example Code VB.NET

```

Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
map.DeleteEntry(mapEntry)
mapCltn.Update

```

Example Code C#

```

VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
map.DeleteEntry(mapEntry);
mapCltn.Update;

```

FirstMapEntry

Method of VcMap

This method can be used to access the initial value, i.e. the first entry of a map object and then to continue in a forward iteration loop by the method **NextMapEntry** for the entries following. If there is no entry in the map, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcMapEntry	First map entry

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)

map = mapCltn.FirstMap
mapEntry = map.FirstMapEntry
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.FirstMap();
VcMapEntry mapEntry = map.FirstMapEntry();
```

GetMapEntry

Method of VcMap

This method returns the corresponding map entry for the given data field value.

	Data Type	Explanation
Return value	System.String	Map entry according to field value

NextMapEntry

Method of VcMap

This method can be used in a forward iteration loop to retrieve subsequent entries (rows) from a map object after initializing the loop by the method **FirstMapEntry**. If there is no map entry left, a **none** object will be returned (**Nothing** in Visual Basic).

1186 API Reference: VcMap

	Data Type	Explanation
Return value	VcMapEntry	Succeeding map entry

Example Code VB.NET

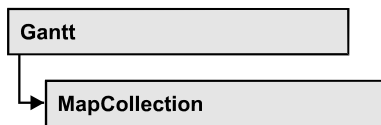
```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.FirstMap
mapEntry = map.FirstMapEntry
While Not mapEntry Is Nothing
    ListBox1.Items.Add(mapEntry.LegendText)
    mapEntry = map.NextMapEntry
End While
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.FirstMap();
VcMapEntry mapEntry = map.FirstMapEntry()
while (mapEntry != null)
{
    listBox1.Items.Add(mapEntry.LegendText);
    mapEntry= map.NextMapEntry();
}
```

7.56 VcMapCollection



An object of the type VcMapCollection contain the maps, which were assigned to the collection by the method **SelectMaps**. You can access all objects in an iterative loop by **For Each map In MapCollection** or by the methods **First...** and **Next...**. You can access a single map using the methods **MapByName** and **MapByIndex**. The number of maps in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the maps in the corresponding way.

Properties

- Count

Methods

- Add
- AddBySpecification
- Copy
- FirstMap
- GetEnumerator
- MapByIndex
- MapByName
- NextMap
- Remove
- SelectMaps
- Update

Properties

Count

Read Only Property of VcMapCollection

This property lets you retrieve the number of maps in the MapCollection object.

	Data Type	Explanation
Property value	System.Int32	Number of maps

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim numberOfMaps As Integer

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
numberOfMaps = mapCltn.Count
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
int numberOfMaps = mapCltn.Count;
```

Methods

Add

Method of VcMapCollection

By this method you can create a map as a member of the MapCollection. If the name has not been used before, the new map object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ mapName	System.String	Map name
Return value	VcMap	New map object

Example Code VB.NET

```
newMap = VcGantt1.MapCollection.Add("Map1")
```

Example Code C#

```
VcMap newMap = vcGantt1.MapCollection.Add("Map1");
```

AddBySpecification

Method of VcMapCollection

This method lets you create a map by using a map specification. This way of creating allows map objects to become persistent. The specification of a map

can be saved and re-loaded (see VcMap property **Specification**). In a subsequent session the map can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter: ⇒ specification	System.String	Map specification
Return value	VcMap	New map object

Copy

Method of VcMapCollection

By this method you can copy a map. If the map that is to be copied exists, and if the name for the new map does not yet exist, the new map object is returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ mapName	System.String	Name of the map to be copied
⇒ newMapName	System.String	Name of the new map
Return value	VcMap	Map object

FirstMap

Method of VcMapCollection

This method can be used to access the initial value, i.e. the first map of a map collection and then to continue in a forward iteration loop by the method **NextMap** for the maps following. If there is no map in the MapCollection, a **none** object will be returned (**Nothing** in Visual Basic). Beforehand, you have to specify a set of maps by the method **SelectMaps**.

	Data Type	Explanation
Return value	VcMap	First map

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.FirstMap
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.FirstMap();
```

GetEnumerator

Method of VcMapCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the map objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

MapByIndex

Method of VcMapCollection

This method lets you access a map by its index. If a map does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the map
Return value	VcMap	Map object returned

MapByName

Method of VcMapCollection

By this method you can get a map by its name. Beforehand, you have to specify a set of maps by the method **SelectMaps**. If a map of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ mapName	System.String	Name of the map
Return value	VcMap	Map

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
```

NextMap**Method of VcMapCollection**

This method can be used in a forward iteration loop to retrieve subsequent maps from a map collection after initializing the loop by the method **FirstMap**. If there is no map left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcMap	Succeeding map

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.FirstMap
While Not map Is Nothing
    ListBox1.Items.Add(map.Name)
    map = mapCltn.NextMap
End While
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.FirstMap();
while (map != null)
{
    listBox1.Items.Add(map.Name);
    map = mapCltn.NextMap();
}
```

Remove

Method of VcMapCollection

This method lets you delete a map. If the map is used in another object, it cannot be deleted. Then False will be returned, otherwise True.

	Data Type	Explanation
Parameter: ⇒ mapName	System.String	Map name
Return value	System.Boolean	Map deleted (True)/not deleted (False)

SelectMaps

Method of VcMapCollection

This method lets you specify which map types your map collection should contain.

	Data Type	Explanation
Parameter: ⇒ selectionType	VcMapType Possible Values: .vcAnyMap 0 .vcColorMap 1 .vcFontMap 8 .vcGraphicsFileMap 7 .vcMillimeterMap 9 .vcNumberMap 10 .vcPatternMap 3 .vcTextMap 6	Map type to be selected any (used only for selecting) Colors Fonts Graphics file Millimeters Numbers Patterns Text
Return value	System.Int32	Number of maps selected

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
```

Update

Method of VcMapCollection

This method has to be used when map modifications have been made and you want to updates all objects that are concerned by the maps you have edited. You should call this method at the end of the code that defines the maps and the map collection. Otherwise the update will be processed before all map definitions are processed.

	Data Type	Explanation
Return value	System.Boolean	Update successful (True)/ not successful (False)

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
While Not mapEntry.DataFieldValue = "A"
    mapEntry = map.NextMapEntry
End While

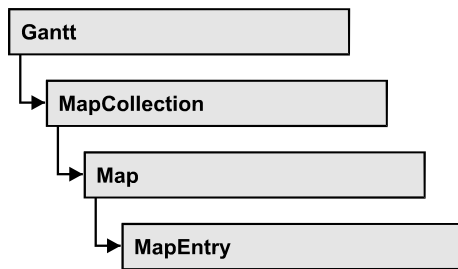
mapEntry.Color = Color.Blue
mapCltn.Update()
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
while (mapEntry.DataFieldValue != "A")
    mapEntry = map.NextMapEntry();

mapEntry.Color = Color.LightSteelBlue;
mapCltn.Update();
```

7.57 VcMapEntry



An object of the type VcMapEntry is a map entry and therefore an element of a map. A map entry is defined by the combination of a data field content of the node's record, a color or graphics file and a legend text.

In each map you can specify up to a maximum of 150 map entries. If you need further map entries, please specify a new map, e. g. as a copy of the current one.

Properties

- Color
- DataFieldValue
- FontBody
- FontName
- FontSize
- GraphicsFileName
- LegendText
- Millimeter
- Number
- Pattern

Properties

Color

Property of VcMapEntry

For Color Maps: This property lets you set or retrieve the color value of a map entry. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255})

Example Code VB.NET

```

Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim colorOfMapEntry As Color

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcColorMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
colorOfMapEntry = mapEntry.Color

```

Example Code C#

```

VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcColorMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
Color colorOfMapEntry = mapEntry.Color;

```

DataFieldValue**Property of VcMapEntry**

This property lets you set or retrieve the content of a data of each map entry.

	Data Type	Explanation
Property value	System.String	Content of the data field

Example Code VB.NET

```

Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim dataFieldValue As String

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcAnyMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry

dataFieldValue = mapEntry.DataFieldValue

```

Example Code C#

```

VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcAnyMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
string dataFieldValue = mapEntry.DataFieldValue;

```


FontBody

Property of VcMapEntry

for Font Maps: This property lets you set or retrieve the font body of the map entry.

	Data Type	Explanation
Property value	VcFontBody Possible Values: .vcBold 2 .vcBoldItalic 4 .vcItalic 3 .vcRegular 1	Font body bold bold and italic italic regular

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim fontBodyOfMapEntry As VcFontBody

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcFontMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
fontBodyOfMapEntry = VcFontBody.vcBold
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcFontMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
VcFontBody fontBodyOfMapEntry = VcFontBody.vcBold;
```

FontName

Property of VcMapEntry

for Font Maps: This property lets you set or retrieve the font name of the map entry.

	Data Type	Explanation
Property value	System.String	Font type

Example Code VB.NET

```

Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim fontNameOfMapEntry As String

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcFontMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
fontNameOfMapEntry = "Arial"

```

Example Code C#

```

VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcFontMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
string fontNameOfMapEntry = "Arial";

```

FontSize

Property of VcMapEntry

for Font Maps: This property lets you set or retrieve the font name of the map entry.

	Data Type	Explanation
Property value	System.Int32	Font size

Example Code VB.NET

```

Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim fontSizeOfMapEntry As Integer

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcFontMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
fontSizeOfMapEntry = 14

```

Example Code C#

```

VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcFontMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
int fontSizeOfMapEntry = 14;

```

GraphicsFileName

Property of VcMapEntry

For Graphic File Maps: This property lets you set or retrieve the graphics file name of a map entry. *Available formats:*

- *.BMP (Microsoft Windows Bitmap)
- *.EMF (Enhanced Metafile oder Enhanced Metafile Plus)
- *.GIF (Graphics Interchange Format)
- *.JPG (Joint Photographic Experts Group)
- *.PNG (Portable Network Graphics)
- *.TIF (Tagged Image File Format)
- *.VMF (Viewer Metafile)
- *.WMF (Microsoft Windows Metafile, probably with EMF included)

EMF, EMF+, VMF and WMF are vector formats that allow to store a file independent of pixel resolution. All other formats are pixel-oriented and confined to a limited resolution.

The VMF format basically has been deprecated, but it will still be supported for some time to maintain compatibility with existing applications.

	Data Type	Explanation
Property value	System.String	Name of the graphics file

Example Code VB.NET

```

Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim exeName As String
Dim exeDir As String

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcGraphicsFileMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry

exeName = System.Environment.GetCommandLineArgs(0)
exeDir = System.IO.Path.GetDirectoryName(exeName)
mapEntry.GraphicsFileName = exeDir + "\Bitmaps\picture1.bmp"

```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcGraphicsFileMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();

String exeName = Environment.GetCommandLineArgs()[0];
mapEntry.GraphicsFileName = System.IO.Path.GetDirectoryName(exeName) +
@"\..\Bitmaps\picture1.bmp";
```

LegendText**Property of VcMapEntry**

This property lets you set or retrieve the legend text of a map entry.

	Data Type	Explanation
Property value	System.String	Legend text

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim legendOfMapEntry As String

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcFontMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
legendOfMapEntry = "1. activity"
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcFontMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
string legendOfMapEntry = "1. activity";
```

Millimeter**Property of VcMapEntry**

for Millimeter Maps: This property lets you set or retrieve the millimetre value of a map entry.

	Data Type	Explanation
Property value	System.Int32	1/100 units

Example Code VB.NET

```
Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim millimeterOfMapEntry As Integer

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcMillimeterMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
millimeterOfMapEntry = 3
```

Example Code C#

```
VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcMillimeterMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
int millimeterOfMapEntry = 3;
```

Number

Property of VcMapEntry

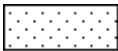

For numeric maps: This property lets you set or retrieve the numeric value of a map entry.

	Data Type	Explanation
Property value	System.Int32	Numeric value













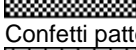
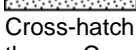




Pattern





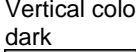


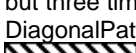
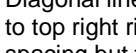

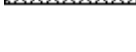
Property of VcMapEntry

For Pattern Maps (vcPatternMap): this property lets you set or retrieve the pattern of a map entry.

	Data Type	Explanation
Property value	VcFillPattern Possible Values: .vc05PercentPattern... vc90PercentPattern 01 - 11 .vcAeroGlassPattern 44	Pattern type Dots in foreground color on background color, the density of the foreground color increasing with the percentage  Vertical color gradient in the color of the fill pattern 

.vcBDiagonalPattern 5	Diagonal lines slanting from bottom left to top right
.vcCrossPattern 6	Cross-hatch pattern
.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width
.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width
.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width
.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width
.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right
.vcDashedHorizontalPattern 2026	Dashed horizontal lines
.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right
.vcDashedVerticalPattern 2027	Dashed vertical lines
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small
.vcDiagonalBrickPattern 2032	Diagonal brick pattern
.vcDivotPattern 2036	Divot pattern
.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines
.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right
.vcHorizontalBrickPattern 2033	Horizontal brick pattern
.vcHorizontalGradientPattern 52	Horizontal color gradient

.vcHorizontalPattern 3	Horizontal lines 
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern 
.vcLargeConfettiPattern 2029	Confetti pattern, large 
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern 
.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern 
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern 
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern 
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern 
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern 
.vcNoPattern 1276	No fill pattern
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large 
.vcPlaidPattern 2035	Plaid pattern 
.vcShinglePattern 2039	Diagonal shingle pattern 
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern 
.vcSmallConfettiPattern 2028	Confetti pattern 
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern 
.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares 
.vcSpherePattern 2041	Checkerboard of spheres 
.vcTrellisPattern 2040	Trellis pattern 

.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright 
.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark 
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright 
.vcVerticalGradientPattern 62	Vertical color gradient 
.vcVerticalPattern 2	Vertical lines 
.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark 
.vcWavePattern 2031	Horizontal waves pattern 
.vcWeavePattern 2034	Interwoven stripes pattern 
.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern 
.vcWideUpwardDiagonalPattern 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of vcBDiagonalPattern 
.vcZigZagPattern 2030	Horizontal zig-zag lines 

Example Code VB.NET

```

Dim mapCltn As VcMapCollection
Dim map As VcMap
Dim mapEntry As VcMapEntry
Dim pattern As VcFillPattern

mapCltn = VcGantt1.MapCollection
mapCltn.SelectMaps(VcMapType.vcPatternMap)
map = mapCltn.MapByName("Map1")
mapEntry = map.FirstMapEntry
pattern = VcFillPattern.vcBDiagonalPattern

```

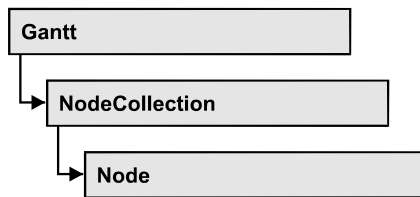
Example Code C#

```

VcMapCollection mapCltn = vcGantt1.MapCollection;
mapCltn.SelectMaps(VcMapType.vcPatternMap);
VcMap map = mapCltn.MapByName("Map1");
VcMapEntry mapEntry = map.FirstMapEntry();
VcFillPattern pattern = VcFillPattern.vcBDiagonalPattern;

```


7.58 VcNode



A node is a basic element of a Gantt diagram. Nodes can be linked to form a structure. What a node looks like is determined by layers, the filters of which are matching the nodes. Nodes can be inserted either interactively or by the VcGantt methods **InsertNodeRecord** or **Open**.

Properties

- AllData
- DataField
- ID
- IncomingLinks
- Marked
- OutgoingLinks
- SnapTargetMode
- SnapTargetMode
- SuperGroup
- UpdateBehaviorName

Methods

- DataRecord
- Delete
- GetPositionInView
- NodeRowInView
- OutlineIndent
- OutlineOutdent
- RelatedDataRecord
- SetPositionInView
- Update

Properties

AllData

Property of VcNode

This record lets you set or retrieve all data of a node at once. When setting the property, a CSV string (using semicolons as separators) or an object that contains all data fields of the node in an array are allowed. When retrieving the property, a string will be returned. (See also **InsertNodeRecord**.)

	Data Type	Explanation
Property value	System.String	All data of the data set

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeModifying(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeModifyingEventArgs) Handles VcGantt1.VcNodeModifying
    Dim allDataOfNode As String
    e.ReturnStatus = VcReturnStatus.vcRetStatFalse

    allDataOfNode = e.Node.AllData
    MsgBox(allDataOfNode)
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeModifying(object sender,
NETRONIC.XGantt.VcNodeModifyingEventArgs e)
{
    e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
    string allDataOfNode = e.Node.AllData.ToString();
    MessageBox.Show(allDataOfNode);
}
```

DataField

Property of VcNode

This property lets you assign/retrieve data to/from the data field of a node. If the data field was modified by the **DataField** property, the diagram needs to be updated by the **Update** method.

The property DataField is an Indexed Property, which in C# is addressed by the methods set_DataField (index, pvn) and get_DataField (index).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of data field

1206 API Reference: VcNode

Property value	System.Object	Content of the data field
----------------	---------------	---------------------------

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles VcGantt1.VcNodeRightClicking
    If MsgBox("Delete node: " + e.Node.DataField(0), MsgBoxStyle.YesNo, "Delete
node") = MsgBoxResult.Yes Then
        e.Node.Delete()
    End If
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeRightClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    DialogResult retVal = MessageBox.Show("Delete node: " +
e.Node.get_DataField(0), "Deleting node", MessageBoxButtons.YesNo);
    if (retVal == DialogResult.Yes)
        e.Node.Delete();
    else
        e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

ID

Read Only Property of VcNode

By this property you can retrieve the ID of a node.

	Data Type	Explanation
Property value	System.String	Node ID

Example Code VB.NET

```
VcNode node = VcGantt1.NodeCollection.FirstNode()

MsgBox (node.ID)
```

Example Code C#

```
VcNode node = vcGantt1.NodeCollection.FirstNode();

MessageBox.Show (node.ID)
```

IncomingLinks

Read Only Property of VcNode

This property lets you access all incoming links of a node.

	Data Type	Explanation
Property value	VcLinkCollection	Link collection

Example Code VB.NET

```

Private Sub VcGantt1_VcNodeRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles VcGantt1.VcNodeRightClicking
    Dim incomingLinks As VcLinkCollection
    Dim link As VcLink
    Dim predecessorNode As VcNode

    incomingLinks = e.Node.IncomingLinks
    For Each link In incomingLinks
        predecessorNode = link.PredecessorNode
        predecessorNode.Marked = True
    Next
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup
End Sub

```

Example Code C#

```

private void vcGantt1_VcNodeRightClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    VcLinkCollection incomingLinks = e.Node.IncomingLinks;
    VcNode predecessorNode;
    foreach (VcLink link in incomingLinks)
    {
        predecessorNode = link.PredecessorNode;
        predecessorNode.Marked = true;
    }
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}

```

Marked**Property of VcNode**

This property lets you set or retrieve whether a node is marked. The marking assigned will be visible only if on the **Nodes** property page the marking type **No Mark** was not selected.

	Data Type	Explanation
Property value	System.Boolean	Node marked/not marked

Example Code VB.NET

```
Dim nodeCltn As VcNodeCollection
Dim node As VcNode
Dim predecessor As VcNode
Dim linkCltn As VcLinkCollection
Dim link As VcLink

nodeCltn = VcGantt1.NodeCollection
nodeCltn.SelectNodes(VcSelectionType.vcAll)

For Each node In nodeCltn
    linkCltn = node.IncomingLinks
    For Each link In linkCltn
        predecessor = link.PredecessorNode
        predecessor.Marked = True
    Next
Next
```

Example Code C#

```
VcNodeCollection nodeCltn = vcGantt1.NodeCollection;
nodeCltn.SelectNodes(VcSelectionType.vcAll);
VcNode predecessorNode;
VcLinkCollection linkCltn;
foreach (VcNode node in nodeCltn)
{
    linkCltn = node.IncomingLinks;
    foreach (VcLink link in linkCltn)
    {
        predecessorNode = link.PredecessorNode;
        predecessorNode.Marked = true;
    }
}
```

OutgoingLinks

Read Only Property of VcNode

This property lets you access all links that leave a node.

	Data Type	Explanation
Property value	VcLinkCollection	Link collection

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles VcGantt1.VcNodeRightClicking

    Dim outgoingLinks As VcLinkCollection
    Dim link As VcLink
    Dim successorNode As VcNode

    outgoingLinks = e.Node.OutgoingLinks
    For Each link In outgoingLinks
        successorNode = link.SuccessorNode
        successorNode.Marked = True
    Next
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup
End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeRightClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    VcLinkCollection outgoingLinks = e.Node.OutgoingLinks;
    VcNode successorNode;
    foreach (VcLink link in outgoingLinks)
    {
        successorNode = link.SuccessorNode;
        successorNode.Marked = true;
    }
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
}
```

SnapTargetMode**Read Only Property of VcNode**

This property lets you set or retrieve whether this node is to be selected as possible snap target manually or automatically.

	Data Type	Explanation
Property value	VcNodeSnapTargetMode	This node's selection mode for moving with snap targets switched on Default value: vcNSTMAutomatically

SnapTargetMode**Read Only Property of VcNode**

This property lets you set or retrieve whether this node is to be selected as possible snap target manually or automatically.

	Data Type	Explanation
Property value	VcNodeSnapTargetMode	This node's selection mode for moving with snap targets switched on Default value: vcNSTMAutomatically

SuperGroup**Read Only Property of VcNode**

This property lets you enquire the group that this node belongs to.

	Data Type	Explanation
Property value	VcGroup	Group that the node belongs to

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles VcGantt1.VcNodeRightClicking

    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup

    Dim group As VcGroup
    group = e.Node.SuperGroup
    Label1.Text = "Group: " + group.Name

End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeRightClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
    VcGroup group = e.Node.SuperGroup;
    label1.Text = "Group: " + group.Name;
}
```

UpdateBehaviorName

Property of VcNode

This property lets you set or retrieve the name of the UpdateBehavior.

	Data Type	Explanation
Property value	System.String	Name of the UpdateBehavior

Methods

DataRecord

Method of VcNode

This property lets you retrieve the node as a data record object. The properties of the data record object give access to the corresponding data table and the data table collection.

	Data Type	Explanation
Return value	VcDataRecord	Data record returned

Delete

Method of VcNode

This method lets you delete a node.

	Data Type	Explanation
Return value	System.Boolean	Node was/was not deleted successfully

Example Code VB.NET

```
Private Sub VcGantt1_VcNodeRightClicking(ByVal sender As Object, ByVal e As
NETRONIC.XGantt.VcNodeClickingEventArgs) Handles VcGantt1.VcNodeRightClicking

    If MsgBox("Delete node: " + e.Node.DataField(0), MsgBoxStyle.YesNo, "Delete
node") = MsgBoxResult.Yes Then
        e.Node.Delete()
        e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup
    End If

End Sub
```

Example Code C#

```
private void vcGantt1_VcNodeRightClicking(object sender,
NETRONIC.XGantt.VcNodeClickingEventArgs e)
{
    DialogResult retVal = MessageBox.Show("Delete node: " +
e.Node.get_DataField(0), "Deleting node", MessageBoxButtons.YesNo);
    if (retVal == DialogResult.Yes)
    {
        e.Node.Delete();
        e.ReturnStatus = VcReturnStatus.vcRetStatNoPopup;
    }
}
```

GetPositionInView

Method of VcNode

This method lets you retrieve the position of a node in the visible area of the diagram.

	Data Type	Explanation
Parameter: viewReferencePoint	VcViewReferencePoint Possible Values: .vcVRPBottomCenter 28 .vcVRPBottomLeft 27 .vcVRPBottomRight 29 .vcVRPCenterCenter 25 .vcVRPCenterLeft 24 .vcVRPCenterRight 26 .vcVRPTopCenter 22 .vcVRPTopLeft 21 .vcVRPTopRight 23	Reference point (of the diagram) bottom center bottom left bottom right center center center left center right top center top left top right

nodeReferencePoint	VcNodeReferencePoint	Node reference point
	Possible Values: .vcNRPBottomCenter 28 .vcNRPBottomLeft 27 .vcNRPBottomRight 29 .vcNRPCenterCenter 25 .vcNRPCenterLeft 24 .vcNRPCenterRight 26 .vcNRPTopCenter 22 .vcNRPTopLeft 21 .vcNRPTopRight 23	bottom center bottom left bottom right center center center left center right top center top left top right
⇨ xOffset	System.Int32	X value of the offset (= distance of the node reference point and the reference point) (unit: pixels)
⇨ yOffset	System.Int32	Y value of the offset (unit: pixels)
Return value	Void	

NodeRowInView

Method of VcNode

This method lets you enquire whether (True) or not (False) the row that this node is in is displayed in the visible section of the diagram.

	Data Type	Explanation
Return value	System.Boolean	Row is/is not in the visible section of the diagram

Example Code VB.NET

```
Dim node As VcNode

node = VcGantt1.GetNodeByID(15)
If Not node.NodeRowInView Then
    VcGantt1.ScrollToNodeLine(node, VcVerticalAlignment.vcVerCenterAligned)
End If
```

Example Code C#

```
VcNode node = vcGantt1.GetNodeByID(2);
if (node.NodeRowInView() == false)
    vcGantt1.ScrollToNodeLine(node, VcVerticalAlignment.vcVerCenterAligned);
```

OutlineIndent

Method of VcNode

This method allows to demote a node in a diagram hierarchy, the node being indented, i.e. moved towards the right within the table while remaining in its row. This method corresponds to the **Outline indent** item in the node context menu.

The return value indicates whether the method could be performed successfully. For example, nodes on the lowest level cannot be demoted.

	Data Type	Explanation
Return value	System.Boolean	Method successful (True)/ not successful (False)

Example Code VB.NET

```
Private Sub CommandOutlineIndent_Click()
    Dim anode As VcNode
    Set anode = getSelectedNode 'Function returns selected node
    MsgBox anode.OutlineIndent
    Set anode = Nothing
End Sub
```

Example Code C#

```
Private Sub CommandOutlineIndent_Click()
    Dim anode As VcNode
    Set anode = getSelectedNode 'Function returns selected node
    MsgBox anode.OutlineIndent
    Set anode = Nothing
End Sub
```

OutlineOutdent

Method of VcNode

This method allows to promote a node in a diagram hierarchy, the node being outdented, i.e. moved to the left within the table and remaining in its row. This method corresponds to the **Outline outdent** item in the context menu for nodes.

The return value indicates whether the method could be performed successfully. For example, nodes on the highest level cannot be promoted.

	Data Type	Explanation
Return value	System.Boolean	Method successful (True)/ not successful (False)

Example Code VB.NET

```
Private Sub CommandOutlineOutdent_Click()
    Dim anode As VcNode
    Set anode = getSelectedNode 'Function returns selected node
    MsgBox anode.OutlineOutdent
    Set anode = Nothing
End Sub
```

Example Code C#

```

Private Sub CommandOutlineOutdent_Click()
    Dim anode As VcNode
    Set anode = getSelectedNode    'Function returns selected node
    MsgBox anode.OutlineOutdent
    Set anode = Nothing
End Sub

```

RelatedDataRecord**Method of VcNode**

This property lets you retrieve a data record from a data table that is related to the node data table. The index passed by the parameter denotes the field in the data record that holds the key of the related data record.

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of data field that holds the key
Return value	VcDataRecord	Related data record returned

SetPositionInView**Method of VcNode**

This method sets that the node will be displayed in a visible position of the diagram after scrolling. The position is specified by an offset vector (x,y) between a reference point in the node and a reference point in the diagram.

	Data Type	Explanation
Parameter: viewReferencePoint	VcViewReferencePoint Possible Values: .vcVRPBottomCenter 28 .vcVRPBottomLeft 27 .vcVRPBottomRight 29 .vcVRPCenterCenter 25 .vcVRPCenterLeft 24 .vcVRPCenterRight 26 .vcVRPTopCenter 22 .vcVRPTopLeft 21 .vcVRPTopRight 23	Reference point (of the diagram) bottom center bottom left bottom right center center center left center right top center top left top right
nodeReferencePoint	VcNodeReferencePoint Possible Values: .vcNRPBottomCenter 28 .vcNRPBottomLeft 27 .vcNRPBottomRight 29 .vcNRPCenterCenter 25 .vcNRPCenterLeft 24	Node reference point bottom center bottom left bottom right center center center left

	.vcNRPCenterRight 26 .vcNRPTopCenter 22 .vcNRPTopLeft 21 .vcNRPTopRight 23	center right top center top left top right
⇨ xOffset	System.Int32	X value of the offset (= distance of the node reference point and the reference point) (unit: pixels)
⇨ yOffset	System.Int32	Y value of the offset (unit: pixels)
Return value	Void	

Example Code VB.NET

```
' scroll the diagram so that the vector between the bottom right corner of the
node and the bottom right corner of the diagram is (-10, -10)
Dim node As VcNode
```

```
node.SetPositionInView(VcViewReferencePoint.vcVRPBottomRight,
VcNodeReferencePoint.vcNRPBottomRight, -10, -10)
```

Example Code C#

```
// scroll the diagram so that the vector between the bottom right corner of the
node and the bottom right corner of the diagram is (-10, -10)
VcNode node;
node.SetPositionInView(VcViewReferencePoint.vcVRPBottomRight,
VcNodeReferencePoint.vcNRPBottomRight, -10, -10);
```

Update

Method of VcNode

If data fields of a node have been modified by the **DataField** property, the diagram needs to be updated by the **Update** method.

	Data Type	Explanation
Return value	System.Boolean	Node was/was not updated successfully

Example Code VB.NET

```
Dim nodeCltn As VcNodeCollection
Dim node As VcNode

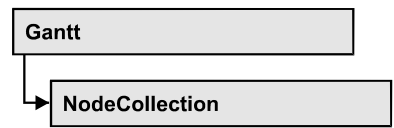
nodeCltn = VcGantt1.NodeCollection
node = nodeCltn.FirstNode

node.DataField(12) = "Group A"
node.Update()
```

Example Code C#

```
VcNodeCollection nodeCltn = vcGantt1.NodeCollection;
VcNode node = nodeCltn.FirstNode();
node.set_DataField(12, "Group A");
node.Update();
```

7.59 VcNodeCollection



An object of the type VcNodeCollection contains all nodes available in the diagram. You can select a part of them by using the method **SelectNodes**. You can access all objects in an iterative loop by **For Each node In NodeCollection** or by the methods **First...** and **Next...**. The number of nodes in the collection object can be retrieved by the property **Count**.

Properties

- Count

Methods

- FirstNode
- GetEnumerator
- NextNode
- SelectNodes

Properties

Count

Read Only Property of VcNodeCollection

This property lets you retrieve the number of nodes in the NodeCollection object.

	Data Type	Explanation
Property value	System.Int32	Number of Nodes in the node collection

Example Code VB.NET

```
Dim nodeCltn As VcNodeCollection

nodeCltn = VcGantt1.NodeCollection
MsgBox("Number of nodes: " + nodeCltn.Count)
```

Example Code C#

```
VcNodeCollection nodeCltn = vcGantt1.NodeCollection;
MessageBox.Show("Number of nodes: " + nodeCltn.Count);
```

Methods

FirstNode

Method of VcNodeCollection

This method can be used to access the initial value, i.e. the first node of a NodeCollection, and then to continue in a forward iteration loop by the method **NextNode** for the nodes following. If there is no node in the NodeCollection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcNode	First Node

Example Code VB.NET

```
Dim nodeCltn As VcNodeCollection
Dim node As VcNode

nodeCltn = VcGantt1.NodeCollection
node = nodeCltn.FirstNode
```

Example Code C#

```
VcNodeCollection nodeCltn = vcGantt1.NodeCollection;
VcNode node = nodeCltn.FirstNode();
```

GetEnumerator

Method of VcNodeCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the node objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

NextNode

Method of VcNodeCollection

This method can be used in a forward iteration loop to retrieve subsequent nodes from a node collection after initializing the loop by the method **FirstNode**. If there is no node left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcNode	Succeeding node

Example Code VB.NET

```

Dim nodeCltn As VcNodeCollection
Dim node As VcNode

nodeCltn = VcGantt1.NodeCollection
node = nodeCltn.FirstNode
While Not node Is Nothing
    node.Marked = False
    node = nodeCltn.NextNode
End While

```

Example Code C#

```

VcNodeCollection nodeCltn = vcGantt1.NodeCollection;
VcNode node = nodeCltn.FirstNode();
while (node != null)
{
    node.Marked = false;
    node = nodeCltn.NextNode;
}

```

SelectNodes**Method of VcNodeCollection**

This method lets you specify the nodes to be collected by the NodeCollection object.

	Data Type	Explanation
Parameter: ⇒ selType	VcSelectionType Possible Values: .vcAll 0 .vcAllLinksCausingCycles 7 .vcAllLinksInCycles 6 .vcAllVisible 1 .vcSelected 2	Nodes to be selected All objects in the diagram will be selected If this selection type is chosen, the link collection will contain all links that cause the existence of cycles. If these links are deleted, cycles will cede to exist in this chart. If this selection type is chosen, the link collection will contain all links that participate in forming cycles. Cycles are chains of nodes and links of which the beginning and end join. All visible objects will be selected All marked objects will be selected
Return value	System.Int32	Number of nodes selected

Example Code VB.NET

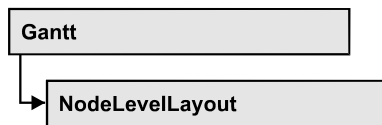
```
Dim nodeCltn As VcNodeCollection
Dim node As VcNode

nodeCltn = VcGantt1.NodeCollection
nodeCltn.SelectNodes(VcSelectionType.vcSelected)
```

Example Code C#

```
VcNodeCollection nodeCltn = vcGantt1.NodeCollection;
nodeCltn.SelectNodes(VcSelectionType.vcSelected);
```


7.60 VcNodeLevelLayout



An object of the type `VcNodeLevelLayout` defines the sorting of nodes as well as the appearance of node rows.

Properties

- `CalendarGridName`
- `CalendarGridsVisible`
- `DateLineName`
- `DateLinesVisible`
- `RowBackgroundColorAsARGB`
- `RowBackgroundColorDataFieldIndex`
- `RowBackgroundColorMapName`
- `RowPattern`
- `RowPatternColorAsARGB`
- `RowPatternColorDataFieldIndex`
- `RowPatternColorMapName`
- `RowPatternDataFieldIndex`
- `RowPatternMapName`
- `SeparationLineColor`
- `SeparationLineInterval`
- `SeparationLinesVisible`
- `SeparationLinesVisibleAtTop`
- `SeparationLineThickness`
- `SeparationLineType`
- `SortDataFieldIndex`
- `SortOrder`

Properties

CalendarGridName

Property of VcNodeLevelLayout

This property lets you set or retrieve the name of the calendar grid. You can also set this property in the **Nodes** section of the **Grouping** dialog.

	Data Type	Explanation
Property value	System.String	name of the calendar grid

CalendarGridsVisible

Property of VcNodeLevelLayout

This property lets you set or retrieve whether calendar grids are to be displayed.

This property also can be set in the **Nodes** section of the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Workfree periods are/are not accentuated

DateLineName

Property of VcNodeLevelLayout

This property lets you set or retrieve the name of the date line for this node level layout. You can also set this property in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.String	Name of the date line

DateLinesVisible

Property of VcNodeLevelLayout

This property lets you set or retrieve whether date lines are to be displayed. This property also can be set in the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Date lines are/are not displayed.

RowBackgroundColorAsARGB

Property of VcNodeLevelLayout

This property lets you set or retrieve the background color of the rows. The default color is white.

	Data Type	Explanation

RowBackgroundColorDataFieldIndex

Property of VcNodeLevelLayout

This property lets you set or retrieve the data field index to be used with a color map specified by the property **RowBackColorMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation

RowBackColorMapName

Property of VcNodeLevelLayout

This property lets you set or retrieve the name of a color map (type vcColorMap). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **RowBackColorDataFieldIndex**, then the background color is controlled by the map. If no data field entry applies, the background color that is specified in the property **RowBackColor** will be used.

	Data Type	Explanation

RowPattern

Read Only Property of VcNodeLevelLayout

This property lets you set or retrieve the background pattern of the node rows of this group level.

	Data Type	Explanation
Property value	FillPatternEnum	Pattern type

RowPatternColorAsARGB

Property of VcNodeLevelLayout

This property lets you set or retrieve the pattern color of the node rows of this group level. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color. When casting an RGB value on an ARGB value, an alpha value of 255 has to be added.

Also see **set/getRowBackColorAsARGB**.

If in the property **RowPatternColorMapName** a map is specified, the map will control the pattern color in dependence of the data.

	Data Type	Explanation
Property value	System.Drawing.Color	ARGB color values {0...255},{0...255},{0...255},{0...255})

RowPatternColorDataFieldIndex

Property of VcNodeLevelLayout

This property lets you set or retrieve the data field index that has to be specified if the property **RowPatternColorMapName** is used. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int32	Data field index

RowPatternColorMapName

Property of VcNodeLevelLayout

This property lets you set or retrieve the name of a color map (type `vcColorMap`). If set to "", no map will be used. Only if a map name and a data field index are specified in the property **RowPatternColorDataFieldIndex**, the pattern color is controlled by the map. If no data field entry applies, the pattern color of the group title row that is specified in the property **RowPatternColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

RowPatternDataFieldIndex

Property of VcNodeLevelLayout

This property lets you set or retrieve the data field index to be used together with the property **RowPatternMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int32	Data field index

RowPatternMapName

Property of VcNodeLevelLayout

This property lets you set or retrieve the name of a pattern map (type `vcPatternMap`). If set to "", no map will be used. Only if a map name and additionally a data field index are specified in the property **RowPatternDataFieldIndex**, the pattern is controlled by the map. If no data field entry applies, the pattern of the layer that is specified in the property **RowPattern** will be used.

	Data Type	Explanation
Property value	System.String	Name of the pattern map

SeparationLineColor

Property of VcNodeLevelLayout

This property lets you set or retrieve the color of the separation lines of the the grouping levels.

This property also can be set in the **Grouping** dialog, section **Nodes**, field **Separation Line**.

	Data Type	Explanation
Property value	System.Drawing.Color	Color value {0...255},{0...255},{0...255}

SeparationLineInterval

Read Only Property of VcNodeLevelLayout

This property lets you set or retrieve after how many activities a separating line is drawn.

	Data Type	Explanation

SeparationLinesVisible

Read Only Property of VcNodeLevelLayout

This property lets you set or retrieve whether separation lines are to be displayed between the activities.

This property also can be set in the **Nodes** section of the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Separation lines are displayed/not displayed

SeparationLinesVisibleAtTop

Read Only Property of VcNodeLevelLayout

This property lets you set or retrieve whether separation lines are to be displayed above activities or below.

This property also can be set in the **Nodes** section of the **Grouping** dialog.

	Data Type	Explanation
Property value	System.Boolean	Separation lines at top are displayed/not displayed

SeparationLineThickness

Read Only Property of VcNodeLevelLayout

This property lets you set or retrieve the line thickness of a separation line between node levels.

If you set this property to values between 1 and 4, an absolute line thickness is defined in pixels. Irrespective of the zoom factor a line will always show the same line thickness in pixels. When printing though, the line thickness is adapted for the sake of legibility and becomes dependent of the zoom factor:

Value	Points	mm
1	1/2 point	0.09 mm
2	1 point	0.18 mm
3	3/2 points	0.26 mm
4	2 points	0.35 mm

A point equals 1/72 inch and represents the unit of the font size.

If you set this property to values between 5 and 1,000, the line thickness is defined in 1/100 mm, so the lines will be displayed in a true thickness in pixels that depends on the zoom factor.

This property also can be set in the **Grouping** dialog, section **Groupwise**, field **Separation Line**.

	Data Type	Explanation
Property value	System.Int32	Line thickness LineType {1...4}: line thickness in pixels LineType {5...1000}: line thickness in 1/100 mm

SeparationLineType

Read Only Property of VcNodeLevelLayout

This property lets you specify/enquire the line type of a date line.

This property also can be set in the **Grouping** dialog, section **Nodes**, field **Separation Line**.

	Data Type	Explanation
Property value	LineTypeEnum	Type of separation lines of hierarchy levels

SortDataFieldIndex

Property of VcNodeLevelLayout

This property lets you set/retrieve the data field index used for sorting the nodes of this VcGroupLevelLayout object

	Data Type	Explanation
Parameter: ⇒ sortlevel	System.Int32	Sorting level
Property value	System.Int32	sorting field Default value: vcAscending

SortOrder

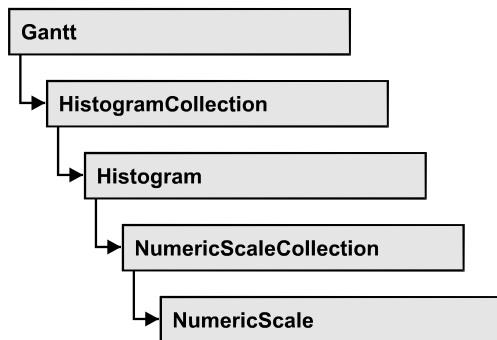
Property of VcNodeLevelLayout

This property lets you specify the sorting order of activities (ascending or descending). The property **SortDataFieldIndex** lets you specify the field the activities are sorted by. This property also can be set in the **Grouping** dialog.

1228 API Reference: VcNodeLevelLayout

	Data Type	Explanation
Parameter: ⇒ sortLevel	System.Int32	Sorting level
Property value	SortOrderEnum	Ascending or descending order Default value: vcAscending

7.61 VcNumericScale



An object of the type VcNumericScale is the scale of the vertical axis of a histogram.

Properties

- DoubleOutputFormat
- Font
- FontColor
- Histogram
- LineColor
- MajorTicks
- MajorTicksEx
- MinorTicks
- MinorTicksEx
- Name
- PatternBackgroundColorAsARGB
- PatternColorAsARGB
- PatternEx
- ThreeDEffect
- TickColor
- Title
- Unit
- UnitEx
- UnitLabel
- UnitWidth
- UpdateBehaviorName

Properties

DoubleOutputFormat

Property of VcNumericScale

This property lets you set or retrieve the output format of numbers as a double value in a numeric scale. The format is presented by the below characters:

- Text
- I
- D

plus the separators **comma** and **period**. **Text** represents a character string; **I** represents the figures in front of the decimal separator and **D** represents the figures after the decimal separator. The overall sequence is **Text I D Text**, where a comma and a period can be inserted in the places desired. As an example be the number -284901,3458. By the format **I,DDDD ppm** it will be output as **-284901,3458 ppm**. By the format **\$I,III.DD** it will be output as **\$-284,901.35**.

	Data Type	Explanation
Property value	System.String	Character string which describes the double format, for example "\$I,III.DD".

Example Code VB.NET

```
VcGantt1.DoubleOutputFormat = "I,DDDD ppm"
```

Example Code C#

```
vcGantt1.DoubleOutputFormat = "$I,III.DD";
```

Font

Property of VcNumericScale

This property lets you set or retrieve the font attributes of the numeric scale.

	Data Type	Explanation
Property value	System.DrawingFont	Font attributes of the numeric scale

Example Code VB.NET

```

Dim histogram As VcHistogram
Dim numericScale As VcNumericScale
Dim newFont As Font

histogram = VcGantt1.HistogramCollection.FirstHistogram()
numericScale = histogram.NumericScaleCollection.FirstNumericScale()
newFont = New Font("Times New Roman", 14, FontStyle.Italic)
numericScale.Font = newFont

```

Example Code C#

```

VcHistogram histogram = vcGantt1.HistogramCollection.FirstHistogram();
VcNumericScale numericScale =
    histogram.NumericScaleCollection.FirstNumericScale()
Font newFont = new Font("Times New Roman", 14, FontStyle.Italic);
numericScale.Font = newFont;

```

FontColor

Property of VcNumericScale

This property lets you set or retrieve the font color of the numeric scale.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255}) Default value: RGB (0,0,0)

Example Code VB.NET

```

Dim histogram As VcHistogram
Dim numericScale As VcNumericScale

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
numericScale = histogram.NumericScaleCollection.Active
numericScale.FontColor = Color.Blue

```

Example Code C#

```

VcHistogram histogram =
    vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcNumericScale numericScale = histogram.NumericScaleCollection.Active;
numericScale.FontColor = Color.LightSteelBlue;

```

Histogram

Read Only Property of VcNumericScale

This property lets you retrieve the histogram to which the numeric scale belongs.

	Data Type	Explanation
Property value	VcHistogram	Histogram object

Example Code VB.NET

```
Private Sub VcGantt1_VcNumericScaleLeftDoubleClicking(ByVal sender As
System.Object, ByVal e As NETRONIC.XGantt.VcNumericScaleClickingEventArgs)
Handles VcGantt1.VcNumericScaleLeftDoubleClicking
```

```
    MessageBox.Show("Clicked on numeric scale of the histogram " +
e.NumericScale.Histogram.Name)
```

```
End Sub
```

Example Code C#

```
private void vcGantt1_VcNumericScaleLeftDoubleClicking(object sender,
VcNumericScaleClickingEventArgs e)
{
    MessageBox.Show("Clicked on numeric scale of the histogram " +
e.NumericScale.Histogram.Name);
}
```

LineColor

Property of VcNumericScale

This property lets you set or retrieve the tick color for all numeric ribbons of histograms.

If you set the color, it will be changed for the border lines of **all** numeric ribbons, retrieving will deliver the tick color of the **first** numeric scale ribbon

.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255}

MajorTicks

Property of VcNumericScale

This property lets you set or retrieve after how many units a major tick is drawn that has an annotation. Also see **set/getMinorTick**. You can also set the number of the units in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.Int16	Number of units between two major ticks

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim numericScale As VcNumericScale

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
numericScale = histogram.NumericScaleCollection.Active
numericScale.MajorTicks = 4
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcNumericScale numericScale = histogram.NumericScaleCollection.Active;
numericScale.MajorTicks = 4;
```

MajorTicksEx

Property of VcNumericScale

This property lets you set or retrieve after how many units a major tick is drawn that has an annotation. Compared to the property **MajorTicks**, this property can be used to set floating point values. You can also set the number of units in the **Edit Histogram** dialog.

Also see **set/getMinorTicks**.

	Data Type	Explanation
Property value	System.Double	Number of units between two major ticks

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim numericScale As VcNumericScale

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
numericScale = histogram.NumericScaleCollection.Active
numericScale.MajorTicks = 4
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcNumericScale numericScale = histogram.NumericScaleCollection.Active;
numericScale.MajorTicks = 4;
```

MinorTicks

Property of VcNumericScale

This property lets you set or retrieve after how many time units a minor tick without annotation is drawn. Also see **set/getMinorTick**. You can also set the number of the units in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.Int16	Number of units between two minor ticks

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim numericScale As VcNumericScale

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
numericScale = histogram.NumericScaleCollection.Active
numericScale.MinorTicks = 2
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcNumericScale numericScale = histogram.NumericScaleCollection.Active;
numericScale.MinorTicks = 2;
```

MinorTicksEx

Property of VcNumericScale

This property lets you set or retrieve after how many time units a minor tick without annotation is drawn. Compared to the property **MinorTicks**, this property can be used to set floating point values. You can also set the number of the units in the **Edit Histogram** dialog. Also see **set/getMajorTicks**.

	Data Type	Explanation
Property value	System.Double	Number of units between two minor ticks

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim numericScale As VcNumericScale

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
numericScale = histogram.NumericScaleCollection.Active
numericScale.MinorTicks = 2
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcNumericScale numericScale = histogram.NumericScaleCollection.Active;
numericScale.MinorTicks = 2;
```

Name

Read Only Property of VcNumericScale

This property lets you retrieve the name of a numeric scale of an histogram. The name can be set in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.String	Name of the numeric scale

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim numericScale As VcNumericScale

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
numericScale = histogram.NumericScaleCollection.Active
MsgBox("Active numeric Scale: " + numericScale.Name)
```

Example Code C#

```
VcHistogram histogram =
VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcNumericScale numericScale = histogram.NumericScaleCollection.Active;
MessageBox.Show("Active numeric scale: " + numericScale.Name);
```

PatternBackgroundColorAsARGB

Property of VcNumericScale

This property lets you set or retrieve the background color of the numeric scale. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color. When casting an RGB value on an ARGB value, an alpha value of 255 has to be added.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255}) Default value: -1

PatternColorAsARGB

Property of VcNumericScale

This property lets you set or retrieve the pattern color of the numeric scale. Color values have a transparency or alpha value, followed by a value for a







red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color. When casting an RGB value on an ARGB value, an alpha value of 255 has to be added.


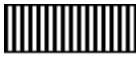





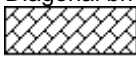
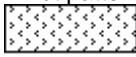
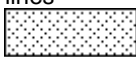
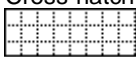


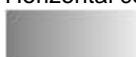




	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255}) Default value: -1






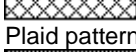


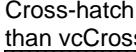
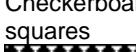

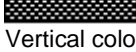
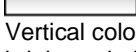
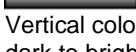




PatternEx







Property of VcNumericScale

This property lets you set or retrieve the background pattern of the numeric scale.

	Data Type	Explanation
Property value	VcFillPattern	Pattern type
	Possible Values: .vc05PercentPattern... .vc90PercentPattern 01 - 11	Dots in foreground color on background color, the density of the foreground color increasing with the percentage 
	.vcAeroGlassPattern 44	Vertical color gradient in the color of the fill pattern 
	.vcBDiagonalPattern 5	Diagonal lines slanting from bottom left to top right 
	.vcCrossPattern 6	Cross-hatch pattern 
	.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width 
	.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 

.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width 
.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 
.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
.vcDashedHorizontalPattern 2026	Dashed horizontal lines 
.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 
.vcDashedVerticalPattern 2027	Dashed vertical lines 
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small 
.vcDiagonalBrickPattern 2032	Diagonal brick pattern 
.vcDivotPattern 2036	Divot pattern 
.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines 
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines 
.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right 
.vcHorizontalBrickPattern 2033	Horizontal brick pattern 
.vcHorizontalGradientPattern 52	Horizontal color gradient 
.vcHorizontalPattern 3	Horizontal lines 
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern 
.vcLargeConfettiPattern 2029	Confetti pattern, large 
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern 

.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern 
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern 
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern 
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern 
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern 
.vcNoPattern 1276	No fill pattern
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large 
.vcPlaidPattern 2035	Plaid pattern 
.vcShinglePattern 2039	Diagonal shingle pattern 
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern 
.vcSmallConfettiPattern 2028	Confetti pattern 
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern 
.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares 
.vcSpherePattern 2041	Checkerboard of spheres 
.vcTrellisPattern 2040	Trellis pattern 
.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright 
.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark 
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright 
.vcVerticalGradientPattern 62	Vertical color gradient 
.vcVerticalPattern 2	Vertical lines

.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark 
.vcWavePattern 2031	Horizontal waves pattern 
.vcWeavePattern 2034	Interwoven stripes pattern 
.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern 
.vcWideUpwardDiagonalPattern 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of vcBDiagonalPattern 
.vcZigZagPattern 2030	Horizontal zig-zag lines 

ThreeDEffect

Property of VcNumericScale

This property lets you set or retrieve whether the three-dimensional look of the numeric scale is switched on.

	Data Type	Explanation
Property value	System.Boolean	3D effect switched on (True)/switched off (False) Default value: False

Example Code VB.NET

```
Dim histogram As VcHistogram
Dim numericScale As VcNumericScale

histogram = VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1")
numericScale = histogram.NumericScaleCollection.Active
numericScale.ThreeDEffect = True
```

Example Code C#

```
VcHistogram histogram =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1");
VcNumericScale numericScale = histogram.NumericScaleCollection.Active;
numericScale.ThreeDEffect = true;
```

TickColor

Property of VcNumericScale

This property lets you set or retrieve the color of all border lines of the numeric scales of histograms.

If you set the color, it will be changed for the border lines of **all** numeric scales, retrieving will deliver the border line color of the **first** numeric scale .

	Data Type	Explanation
Property value	System.Drawing.Color RGB ({0...255},{0...255},{0...255})	RGB color values ({0...255},{0...255},{0...255}) Default value: 0,0,0

Title

Property of VcNumericScale

This property lets you set or retrieve a title of the numeric scale. The ribbon that displays the title needs to be of the ribbon type **textual**. Scales and ribbons can be generated by the **Edit histogram** dialog box which can be invoked from the **Layout** property page.

	Data Type	Explanation
Parameter: ⇒ position	VcNumericAnnotationPosition Possible Values: .vc10PercentFromTop 4 .vc30PercentFromTop 3 .vc50PercentFromTop 2 .vc70PercentFromTop 1 .vc90PercentFromTop 0	Position of the title in the numeric scale 10 % of total scale length away from top 30 % of total scale length away from top 50 % of total scale length away from top 70 % of total scale length away from top 90 % of total scale length away from top
Property value	System.String	Title of the numeric scale

Example Code VB.NET

```
' Title positioned at 50% downward from top
numericScale.Title(VcNumericAnnotationPosition.vc50PercentFromTop) = "1350
Loops"
```

Example Code C#

```
// Title positioned at 50% downward from top
numericScale.set_Title(VcNumericAnnotationPosition.vc50PercentFromTop, "1350
Loops");
```

Unit

Property of VcNumericScale

This property lets you set or retrieve the units of the numeric scale. Also see **set/getUnitWidth**. This property also can be set in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.Int32	unit

UnitEx

Property of VcNumericScale

This property lets you set or retrieve the basic unit of the numeric scale as a double value.

	Data Type	Explanation
Property value	System.Double	unit

Example Code VB.NET

```
Dim numCol As VcNumericScaleCollection
Dim numScale As VcNumericScale

Set numCol = VcGantt1.HistogramCollection.FirstHistogram.NumericScaleCollection
Set numScale = numCol.FirstNumericScale
numScale.UnitEx = numScale.UnitEx / 2
```

Example Code C#

```
VcNumericScaleCollection numColl =
vcGantt1.HistogramCollection.FirstHistogram().NumericScaleCollection;
VcNumericScale numScale = numColl.FirstNumericScale();
numScale.UnitEx = numScale.UnitEx / 2;
```

UnitLabel

Property of VcNumericScale

This property lets you set or retrieve the designation of the units of the numeric scale. This designation is displayed in the middle of the upper border of the numeric scale.

	Data Type	Explanation
Property value	System.String	Designation of the unit

Example Code VB.NET

```
Dim numericScaleCltn As VcNumericScaleCollection
Dim activeNumericScale As VcNumericScale

numericScaleCltn =
VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on
activeNumericScale = numericScaleCltn.Active
activeNumericScale.UnitLabel = "Hours"
```

Example Code C#

```
VcNumericScaleCollection numericScaleCltn =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on;
VcNumericScale activeNumericScale = numericScaleCltn.Active;
activeNumericScale.UnitLabel = "Hours";
```

UnitWidth

Property of VcNumericScale

This property lets you set or retrieve the width of the units of the numeric scale (by 1/100 mm). Also see **set/getUnit**. This property also can be set in the **Edit Histogram** dialog.

	Data Type	Explanation
Property value	System.Int32	unit width (1/100 mm)

Example Code VB.NET

```
Dim numericScaleCltn As VcNumericScaleCollection
Dim activeNumericScale As VcNumericScale

numericScaleCltn =
VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on
activeNumericScale = numericScaleCltn.Active
activeNumericScale.UnitWidth = 200
```

Example Code C#

```
VcNumericScaleCollection numericScaleCltn =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on;
VcNumericScale activeNumericScale = numericScaleCltn.Active;
activeNumericScale.UnitWidth = 200;
```

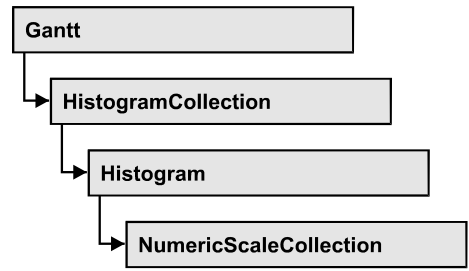
UpdateBehaviorName

Property of VcNumericScale

This property lets you set or retrieve the name of the UpdateBehavior.

	Data Type	Explanation
Property value	System.String	Name of the UpdateBehavior

7.62 VcNumericScaleCollection



An object of the type VcNumericScaleCollection automatically contains all available numeric scales. You can access all objects in an iterative loop by **For Each numericScale In NumericScaleCollection** or by the methods **First...** and **Next...**. You can access a single scale using the methods **NumericScaleByName** and **NumericScaleByIndex**. The number of scales in the collection object can be retrieved by the property **Count**. By the property **Active** you can set or retrieve the scale that is presently active.

Properties

- Active
- Count

Methods

- FirstNumericScale
- GetEnumerator
- NextNumericScale
- NumericScaleByIndex
- NumericScaleByName

Properties

Active

Property of VcNumericScaleCollection

This method lets you set or retrieve the active numeric scale of the histogram.

	Data Type	Explanation
Property value	VcNumericScale	Currently used numeric scale

Example Code VB.NET

```
Dim numericScaleCltn As VcNumericScaleCollection
Dim activeNumericScale As VcNumericScale

numericScaleCltn =
VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on
activeNumericScale = numericScaleCltn.Active
```

Example Code C#

```
VcNumericScaleCollection numericScaleCltn =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on;
VcNumericScale activeNumericScale = numericScaleCltn.Active;
```

Count

Property of VcNumericScaleCollection

This property lets you retrieve the number of numeric scales in the NumericScaleCollection object.

	Data Type	Explanation
Property value	System.Int32	Number of numeric scales

Example Code VB.NET

```
Dim numericScaleCltn As VcNumericScaleCollection
Dim numberOfNumericScales As Integer

numericScaleCltn =
VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on
numberOfNumericScales = numericScaleCltn.Count
```

Example Code C#

```
VcNumericScaleCollection numericScaleCltn =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on;
int numberOfNumericScale = numericScaleCltn.Count;
```

Methods

FirstNumericScale

Method of VcNumericScaleCollection

This method can be used to access the initial value, i.e. the first numeric scale of a numeric scale collection, and then to continue in a forward iteration loop by the method **NextNumericScale** for the scales following. If there is no

scale in the numeric scale collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcNumericScale	First numeric scale

Example Code VB.NET

```
Dim numericScaleCltn As VcNumericScaleCollection
Dim numericScale As VcNumericScale

numericScaleCltn =
VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on
numericScale = numericScaleCltn.FirstNumericScale
```

Example Code C#

```
VcNumericScaleCollection numericScaleCltn =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on;
numericScale = numericScaleCltn.FirstNumericScale();
```

GetEnumerator

Method of VcNumericScaleCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the numeric scales included.

	Data Type	Explanation
Return value	VcObject	Reference object

NextNumericScale

Method of VcNumericScaleCollection

This method can be used in a forward iteration loop to retrieve subsequent numeric scales from a numeric scale collection after initializing the loop by the method **FirstNumericScale**. If there is no numeric scale left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcNumericScale	Succeeding numeric scale

Example Code VB.NET

```

Dim numericScaleCltn As VcNumericScaleCollection
Dim numericScale As VcNumericScale

numericScaleCltn =
VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on
numericScale = numericScaleCltn.FirstNumericScale

While Not numericScale Is Nothing
    ListBox1.Items.Add(numericScale.Name)
    numericScale = numericScaleCltn.NextNumericScale
End While

```

Example Code C#

```

VcNumericScaleCollection numericScaleCltn =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on;
numericScale = numericScaleCltn.FirstNumericScale();
while (numericScale != null)
{
    listBox1.Items.Add(numericScale.Name);
    numericScale = numericScaleCltn.NextNumericScale();
}

```

NumericScaleByIndex**Method of VcNumericScaleCollection**

This method lets you access a numeric scale by its index. If a numeric scale does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the numeric scale
Return value	VcNumericScale	Numeric scale object returned

NumericScaleByName**Method of VcNumericScaleCollection**

By this method you can retrieve a numeric scale by its name. If a numeric scale of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ numericScaleName	System.String	Name of the numeric scale

1248 API Reference: VcNumericScaleCollection

Return value	VcNumericScale	Numeric scale
--------------	----------------	---------------

Example Code VB.NET

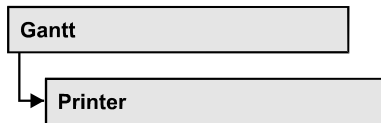
```
Dim numericScaleCltn As VcNumericScaleCollection

numericScaleCltn =
VcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on
numericScaleCltn.Active = numericScaleCltn.NumericScaleByName("STEP1")
```

Example Code C#

```
VcNumericScaleCollection numericScaleCltn =
vcGantt1.HistogramCollection.HistogramByName("HISTOGRAM_1").NumericScaleCollecti
on;
numericScaleCltn.Active = numericScaleCltn.NumericScaleByName("STEP1");
```

7.63 VcPrinter



The VcPrinter object offers a variety of properties to set up the printing process. You can enter the width of top, bottom, left and right margins, set a page frame, page numbers, a page description, cutting marks and the print date. Beside, you can specify the number of pages that the diagram is to be printed on. Zoom factor, alignment, orientation, paper size and color mode are more properties that you can vary for a perfect print.

Properties

- AbsoluteBottomMarginInInches
- AbsoluteLeftMarginInCM
- AbsoluteLeftMarginInInches
- AbsoluteRightMarginInCM
- AbsoluteRightMarginInInches
- AbsoluteTopMarginInCM
- AbsoluteTopMarginInInches
- Alignment
- AllBorderBoxesShownOnCombinedControls
- CombiningControlsEnabled
- CurrentHorizontalPagesCount
- CurrentVerticalPagesCount
- CurrentZoomFactor
- CuttingMarks
- DateFormat
- DefaultPrinterName
- DiagramEnabled
- DiagramEnabled
- DocumentName
- FitToPage
- FoldingMarksType
- MarginsShownInInches
- MaxHorizontalPagesCount
- MaxVerticalPagesCount
- Orientation
- PageDescription

- PageDescriptionString
- PageFrame
- PageNumberMode
- PageNumbers
- PagePaddingEnabled
- PaperSize
- PrintDate
- PrinterName
- PrintPreviewWithFirstPage
- ReOptimizeNodesInGroupsEnabled
- ScalingMode
- TableColumnRanges
- TableTimeScaleOnAllPages
- TableWidthAdoptionFromViewOnScreen
- TimeColumnEndDate
- TimeColumnStartDate
- TimeScaleAdjustment
- VcCalendarGrid
- ZoomFactorAsDouble

Properties

AbsoluteBottomMarginInInches

Property of VcPrinter

This property lets you set or retrieve the absolute height of the bottom margin of the pages to be printed in inches. The true width may be larger if the printer used has to print margins by obligation.

Tip: The internal conversion factor is 2.5 cm/inch instead of the actual correct 2.54 cm/inch so that the values shown in the **Page Setup** dialog will be smoother (1.5 cm so add up to 0.6 inches, 1 cm add up to 0.4 inches).

	Data Type	Explanation
Property value	System.Double	Absolute height of the bottom margin of the page in inches Default value: 0

Example Code VB.NET

```
VcGantt1.Printer.AbsoluteBottomMarginInInches = 0.5 ' 0.5 inches
```

Example Code C#

```
vcGantt1.Printer.AbsoluteBottomMarginInInches = 0.5; // 0.5 inches
```

AbsoluteLeftMarginInCM**Property of VcPrinter**

This property lets you set or retrieve the absolute width of the left margin of the pages to be printed in cm. The true width may be larger if the printer used has to print margins by obligation.

	Data Type	Explanation
Property value	System.Double	Width of the left margin of the page in cm Default value: 0

Example Code VB.NET

```
VcGantt1.Printer.AbsoluteTopMarginInCM = 1.5 ' 2 cm
```

Example Code C#

```
vcGantt1.Printer.AbsoluteTopMarginInCM = 1.5; // 1.5 cm
```

AbsoluteLeftMarginInInches**Property of VcPrinter**

This property lets you set or retrieve the absolute width of the left margin of the pages to be printed in inches. The true width may be larger if the printer used has to print margins by obligation.

Tip: The internal conversion factor is 2.5 cm/inch instead of the actual correct 2.54 cm/inch so that the values shown in the **Page Setup** dialog will be smoother (1.5 cm so add up to 0.6 inches, 1 cm add up to 0.4 inches).

	Data Type	Explanation
Property value	System.Double	Absolute width of the left margin of the page in inches Default value: 0

Example Code VB.NET

```
VcGantt1.Printer.AbsoluteBottomMarginInInches = 0.5 ' 0.5 inches
```


Example Code C#

```
vcGantt1.Printer.AbsoluteBottomMarginInInches = 0.5;    // 0.5 inches
```

AbsoluteRightMarginInCM**Property of VcPrinter**

This property lets you set or retrieve the absolute width of the right margin of the pages to be printed in cm. The true width may be larger if the printer used has to print margins by obligation.

	Data Type	Explanation
Property value	System.Double	Width of the right margin of the page in cm Default value: 0

Example Code VB.NET

```
VcGantt1.Printer.AbsoluteTopMarginInCM = 1.5    ' 2 cm
```

Example Code C#

```
vcGantt1.Printer.AbsoluteTopMarginInCM = 1.5;    // 1.5 cm
```

AbsoluteRightMarginInInches**Property of VcPrinter**

This property lets you set or retrieve the absolute width of the right margin of the pages to be printed in inches. The true width may be larger if the printer used has to print margins by obligation.

Tip: The internal conversion factor is 2.5 cm/inch instead of the actual correct 2.54 cm/inch so that the values shown in the **Page Setup** dialog will be smoother (1.5 cm so add up to 0.6 inches, 1 cm add up to 0.4 inches).

	Data Type	Explanation
Property value	System.Double	Absolute width of the right margin of the page in inches Default value: 0

Example Code VB.NET

```
VcGantt1.Printer.AbsoluteBottomMarginInInches = 0.5    ' 0.5 inches
```

Example Code C#

```
vcGantt1.Printer.AbsoluteBottomMarginInInches = 0.5;    // 0.5 inches
```

AbsoluteTopMarginInCM

Property of VcPrinter

This property lets you set or retrieve the absolute height of the top margin of the pages to be printed in cm. The true width may be larger if the printer used has to print margins by obligation.

	Data Type	Explanation
Property value	System.Double	Height of the top margin of the page in cm Default value: 0

Example Code VB.NET

```
VcGantt1.Printer.AbsoluteTopMarginInCM = 1.5 ' 2 cm
```

Example Code C#

```
vcGantt1.Printer.AbsoluteTopMarginInCM = 1.5; // 1.5 cm
```

AbsoluteTopMarginInInches

Property of VcPrinter

This property lets you set or retrieve the absolute height of the top margin of the pages to be printed in inches. The true width may be larger if the printer used has to print margins by obligation.

Tip: The internal conversion factor is 2.5 cm/inch instead of the actual correct 2.54 cm/inch so that the values shown in the **Page Setup** dialog will be smoother (1.5 cm so add up to 0.6 inches, 1 cm add up to 0.4 inches).

	Data Type	Explanation
Property value	System.Double	Absolute height of the top margin of the page in inches Default value: 0

Example Code VB.NET

```
VcGantt1.Printer.AbsoluteBottomMarginInInches = 0.5 ' 0.5 inches
```

Example Code C#

```
vcGantt1.Printer.AbsoluteBottomMarginInInches = 0.5; // 0.5 inches
```

Alignment

Property of VcPrinter

This property lets you set or retrieve the alignment of the diagram on a page. The property will be effective either if the diagram is put out onto a single page or if the **TableTimeScaleOnAllPages** property was set. In any other case the output will be centered.

	Data Type	Explanation
Property value	VcPrinterAlignment	Alignment of the output with its sheet Default value: vcPCenterCenter
	Possible Values:	
	.vcPBottomCenter 28	Vertical alignment: bottom; horizontal alignment: center
	.vcPBottomLeft 27	Vertical alignment: bottom; horizontal alignment: left
	.vcPBottomRight 29	Vertical alignment: bottom; horizontal alignment: right
	.vcPCenterCenter 25	Vertical alignment: center; horizontal alignment: center
	.vcPCenterLeft 24	Vertical alignment: center; horizontal alignment: left
	.vcPCenterRight 26	Vertical alignment: center; horizontal alignment: right
	.vcPTopCenter 22	Vertical alignment: top; horizontal alignment: center
	.vcPTopLeft 21	Vertical alignment: top; horizontal alignment: left
	.vcPTopRight 23	Vertical alignment: top; horizontal alignment: right

Example Code VB.NET

```
VcGantt1.Printer.Alignment = VcPrinterAlignment.vcPTopLeft
```

Example Code C#

```
vcGantt1.Printer.Alignment = VcPrinterAlignment.vcPTopLeft;
```

AllBorderBoxesShownOnCombinedControls

Property of VcPrinter

If this property is set to "True" all border boxes are printed even if combined printing is activated. If it is set to "False", the border boxes are ignored. See the objects **VcBorderArea** and **VcBorderBox**.

	Data Type	Explanation
Property value	System.Boolean	Border boxes are (True)/are not (False) printed if combined printing is enabled

Example Code VB.NET

```
VcGantt1.AllBorderBoxesShownOnCombinedControls = True
```

Example Code C#

```
vcGantt1.AllBorderBoxesShownOnCombinedControls = True;
```

CombiningControlsEnabled**Read Only Property of VcPrinter**

If this property is set to **True**, all XGantt controls of the parent window are arranged one below the other according to their relative vertical position for exporting, printing and in the print preview. Thus it is possible to display more than one diagram at once.

Tip: When this feature is used, the properties **RepeatTableTimeScale** and **TimeScaleAdjustment** will be ignored and their value assumed as "False". Likewise, the property **VcPrinter.FoldingMarksType** will be ignored and its value assumed as "vcFMTNone" .

	Data Type	Explanation
Property value	System.Boolean	XGantt controls of the parent window are (True) / are not (False) arranged one below the other Default value: False

Example Code VB.NET

```
VcGantt1.Printer.CombiningControlsEnabled = True
```

Example Code C#

```
vcGantt1.Printer.CombiningControlsEnabled = true;
```

CurrentHorizontalPagesCount**Read Only Property of VcPrinter**

This property lets you retrieve the actual number of pages in horizontal direction onto which the chart is to be printed. Also see **CurrentVerticalPagesCount** and **MaxHorizontalPagesCount**.

	Data Type	Explanation
Property value	System.Int32	Current number of pages counted in horizontal direction

CurrentVerticalPagesCount

Read Only Property of VcPrinter

This property lets you retrieve the actual number of pages in vertical direction onto which the chart is to be printed. Also see **CurrentHorizontalPagesCount** and **MaxVerticalPagesCount**.

	Data Type	Explanation
Property value	System.Int32	Current number of pages counted in vertical direction

CurrentZoomFactor

Read Only Property of VcPrinter

This property lets you retrieve the actual zoom factor for the scaling mode **vcFitToPageCount** (zoom factor = 100: original size, zoom factor > 100: enlargement, zoom factor < 100: reduction).

	Data Type	Explanation
Property value	System.Double	Current zoom factor

CuttingMarks

Property of VcPrinter

This property lets you set or retrieve, whether (True) or not (False) cutting marks are to be printed onto a page.

	Data Type	Explanation
Property value	System.Boolean	Cutting marks are (True) / are not (False) printed Default value: False

Example Code VB.NET

```
VcGantt1.Printer.CuttingMarks = True
```

Example Code C#

```
vcGantt1.Printer.CuttingMarks = true;
```

DateFormat

Property of VcPrinter

This property lets you set the date format that is to be used in the DatePicker dialog elements of the **Page Layout** dialog. The empty string represents the default date format TS. To compose the date you can use the below tokens:

D:	first letter of the day of the week (not adjustable)
TD:	Day of the Week (adjustable by using the event VcTextEntrySupplying)
DD:	two-digit figure for the day of the month: 01-31
DDD:	first three letters of the day of the week (not adjustable)
M:	first letter of the name of the month (not adjustable)
TM:	name of the month (adjustable by using the event VcTextEntrySupplying)
MM:	two-digit figure for the month: 01-12
MMM:	first three letters of the name of the month (not adjustable)
YY:	two-digit figure for the year
YYYY:	four-digit figure for the year
WW:	two-digit figure for the number of the calendar week: 01-53
TW:	text for "calendar week" (adjustable by using the event VcTextEntrySupplying)
Q:	one-digit figure for the quarter: 1-4
TQ:	name of quarter (adjustable by using the event VcTextEntrySupplying)
hh	two-digit figure for the hour in 24 hours format: 00-23
HH:	two-digit figure for the hour in 12 hours format: 01-12
Th:	Text of "o' clock" (adjustable by using the event VcTextEntrySupplying)
TH:	"am" or "pm" (adjustable by using the event VcTextEntrySupplying)
mm	two-digit figure for the minute: 00-59
ss:	two-digit figure for the second: 00-59
TS:	short date format, as defined in the regional settings of the windows control panel

TL: long date format, as defined in the regional settings of the windows control panel

TT: time format, as defined in the regional settings of the windows control panel

Note: Characters which are not to be interpreted as part of the date should be preceded by a backslash '\'. '\\' for instance results in '\'. The special characters: ':, /, -' and **blank** don't need '\' as prefix.

	Data Type	Explanation
Property value	System.String	Date format in Page Layout dialog Default value: " "

DefaultPrinterName

Read Only Property of VcPrinter

This property lets you return the current name of the system's current default printer.

	Data Type	Explanation
Property value	System.String	Name of current default printer

DiagramEnabled

Property of VcPrinter

This property lets you specify whether the diagram (timescale and layers) shall be also printed or not.

	Data Type	Explanation
Property value	System.Boolean	Diagram is (True) / is not (False) printed Default value: True

Example Code VB.NET

```
VcGantt1.Printer.DiagramEnabled = True
```

Example Code C#

```
vcGantt1.Printer.DiagramEnabled = true;
```

DiagramEnabled

Property of VcPrinter

This property lets you specify whether the diagram (time scale and layers) shall be printed or not.

	Data Type	Explanation
Property value	System.Boolean	Diagram is (True) / is not (False) printed Default value: True

DocumentName

Property of VcPrinter

This property lets you set or enquire the name of the document. When printing, the document name is displayed in the list of the documents to print and has special functions with certain printer drivers as e.g. drivers which create PDF files.

	Data Type	Explanation
Property value	System.String	Name of document Default value: " "

FitToPage

Property of VcPrinter

This property lets you set or retrieve, whether (True) the diagram is to printed to a set of pages defined by the properties **MaxHorizontalPagesCount** and **MaxVerticalPagesCount**, or whether (False) it is to be printed by the enlargement set by the **ZoomFactor** property.

	Data Type	Explanation
Property value	System.Boolean	Diagram is printed on a defined set of pages/is printed in a defined enlargement.

Example Code VB.NET

```
VcGantt1.Printer.FitToPage = True
```

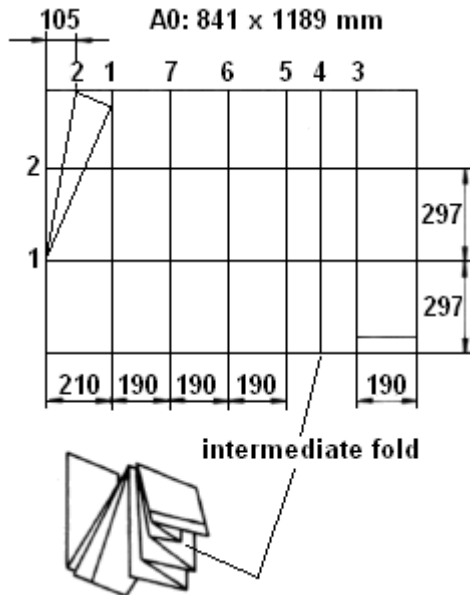
Example Code C#

```
vcGantt1.Printer.FitToPage = true;
```

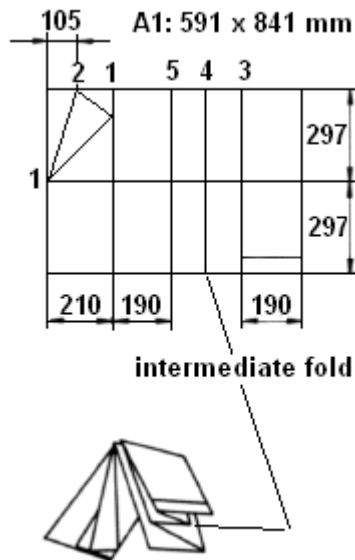

FoldingMarksType

Property of VcPrinter

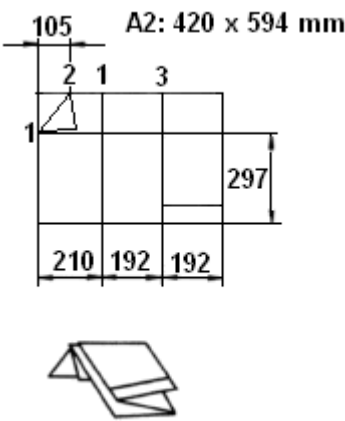
This property lets you set or retrieve folding marks according to DIN 824. The folding marks allow to fold paper sheets of the German DIN-A standard:



Folding of the DIN-A-0 format

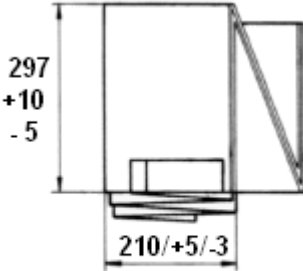


Folding of the DIN-A-1 format



Folding of the DIN-A-2 format

	Data Type	Explanation
Property value	VcFoldingMarksType	Folding marks
		Default value: vcFMTNone
	Possible Values:	
	.vcFMTDIN824FormA 65	Folding marks according to DIN824-A: the drawing can be punched and filed directly to a folder.
		DIN 824-A way of folding
	.vcFMTDIN824FormB 66	Folding marks according to DIN824-B: the chart can be punched and filed to a folder by a flexi filing fastener.
		DIN 824-B way of folding

.vcFMTDIN824FormC 67	Folding marks according to DIN824-C: the folded chart is not to be punched but to be put into a sheet protector.  DIN 824-C way of folding
.vcFMTNone 0	No folding marks

MarginsShownInInches

Property of VcPrinter

This property lets you set or retrieve whether the measuring unit of the margins in the "Page Layout dialog shall be switched to inches. (At present only possible at runtime).

Tip: The internal conversion factor is 2.5 cm/inch instead of the actual correct 2.54 cm/inch so that the values shown in the **Page Setup** dialog will be smoother (1.5 cm so add up to 0.6 inches, 1 cm add up to 0.4 inches).

	Data Type	Explanation
Property value	System.Boolean	Measuring unit of the margins in the Page Layout dialog in inches (True)/ in cm (False) Default value: False

MaxHorizontalPagesCount

Property of VcPrinter

This property lets you set or retrieve the horizontal number of pages für printing and for the print preview. This property only works if the property **ScalingMode** was set to either **vcFitToPageCount** or to **vcZoomWithHorizontalFit**. Also see **MaxVerticalPagesCount**.

	Data Type	Explanation
Property value	System.Int32	Maximum number of pages counted in horizontal direction Default value: 1

Example Code VB.NET

```
VcGantt1.Printer.MaxHorizontalPagesCount = 4
```

Example Code C#

```
vcGantt1.Printer.MaxHorizontalPagesCount = 4;
```

MaxVerticalPagesCount

Property of VcPrinter

This property lets you set or retrieve the vertical number of pages für printing and for the print preview. This property only works if the property **ScalingMode** was set to **vcFitToPageCount**. Also see **MaxHorizontalPagesCount**.

	Data Type	Explanation
Property value	System.Int32	Maximum number of pages counted in vertical direction Default value: 1

Example Code VB.NET

```
VcGantt1.Printer.MaxVerticalPagesCount = 4
```

Example Code C#

```
vcGantt1.Printer.MaxVerticalPagesCount = 4;
```

Orientation

Property of VcPrinter

This property lets you set or retrieve the orientation of the output.

	Data Type	Explanation
Property value	VcOrientation	Orientation Default value: VcPortrait
	Possible Values: .vcLandscape 42 .vcPortrait 41	Printing orientation landscape Printing orientation portrait

Example Code VB.NET

```
VcGantt1.Printer.Orientation = VcOrientation.vcLandscape
```

Example Code C#

```
vcGantt1.Printer.Orientation = VcOrientation.vcLandscape;
```

PageDescription

Property of VcPrinter

This property lets you set or retrieve whether (True) or not (False) the page description string is to appear in the bottom left corner of a page. The contents of the page description string you can set by the **PageDescriptionString** property.

	Data Type	Explanation
Property value	System.Boolean	Page description is (True) / is not printed (False) Default value: False

Example Code VB.NET

```
VcGantt1.Printer.PageDescription = True
```

Example Code C#

```
vcGantt1.Printer.PageDescription = true;
```

PageDescriptionString

Property of VcPrinter

This property lets you set or retrieve a page description in the bottom left corner of each page. Whether or not the page description string is printed you can control by the **PageDescription** property. For numbering the pages you may enter the below codes which will be replaced by the corresponding contents on the printout:

{PAGE} = consecutive numbering of pages

{NUMPAGES} = total number of pages

{ROW} = line position of the section in the complete chart

{COLUMN} = column position of the section in the complete chart

	Data Type	Explanation
Property value	System.String	Page description Default value: Empty string ""

Example Code VB.NET

```
VcGantt1.Printer.PageDescriptionString = "Gantt-Graphics"
```

Example Code C#

```
vcGantt1.Printer.PageDescriptionString = "Gantt-Graphics";
```

PageFrame

Property of VcPrinter

This property lets you set or retrieve, whether (True) or not (False) a frame is to be drawn around the output. If the **TableTimeScaleOnAllPages** property was set, the frame will be drawn around the part on each page, otherwise it will be drawn around the diagram as a whole.

	Data Type	Explanation
Property value	System.Boolean	Page frame is (True) / is not (False) displayed Default value: True

Example Code VB.NET

```
VcGantt1.Printer.PageFrame = True
```

Example Code C#

```
vcGantt1.Printer.PageFrame = true;
```

PageNumberMode

Property of VcPrinter

This property lets you set or retrieve in which way the page numbers are to be displayed: "Page N of M pages" or "x.y" (row no./column no.).

	Data Type	Explanation
Property value	VcPageNumberMode	Mode of page numbering Default value: vcPRowColumn
	Possible Values: .vcPageNOfM 1597 .vcPRowColumn 1596	"Page N of M pages" "x.y" (row no./column no.).

Example Code VB.NET

```
Dim printer As VcPrinter
printer.Orientation = VcOrientation.vcLandscape
printer.PageNumberMode = VcPageNumberMode.vcPageNOFM
printer.PageNumbers = True
printer.FitToPage = False
VcGantt1.ShowPrintPreviewDialog()
```

Example Code C#

```
VcPrinter printer = vcGantt1.Printer;
printer.Orientation = VcOrientation.vcLandscape;
printer.PageNumberMode = VcPageNumberMode.vcPageNOFM;
printer.PageNumbers = true;
printer.FitToPage = false;
vcGantt1.ShowPrintPreviewDialog();
```

PageNumbers

Property of VcPrinter

This property lets you set or retrieve, whether (True) or not (False) a page number is printed. The mode of page numbering is set with the help of the property **PageNumberMode**.

	Data Type	Explanation
Property value	System.Boolean	Page numbers are (True) / are not (False) printed Default value: False

Example Code VB.NET

```
VcGantt1.Printer.PageNumbers = True
```

Example Code C#

```
vcGantt1.Printer.PageNumbers = true;
```

PagePaddingEnabled

Property of VcPrinter

This property lets you specify or retrieve whether enough space is to be left between the diagram and the boxes of the title and legend area so that the boxes are always printed in full width and are attached to the margin. If the property is set to **False** there will be no space left between the diagram and the boxes and their width may vary on the different pages depending on the diagram.

	Data Type	Explanation
Property value	System.Boolean	Space between diagram and boxes for legend/title is (True) / is not (False) left Default value: True

Example Code VB.NET

```
VcGantt1.Printer.PagePaddingEnabled = True
```

Example Code C#

```
vcGantt1.Printer.PagePaddingEnabled = true;
```

PaperSize

Property of VcPrinter

This property lets you set or retrieve the paper size to be used.

	Data Type	Explanation
Property value	VcPaperSize	Paper size
	Possible Values:	
	.vcDIN_A2 66	DIN A2
	.vcDIN_A3 8	DIN A3
	.vcDIN_A4 9	DIN A4
	.vcISO_C 24	ISO C
	.vcISO_D 25	ISO D
	.vcISO_E 26	ISO E
	.vcUS_LEGAL 5	US LEGAL
	.vcUS_LETTER 1	US LETTER

Example Code VB.NET

```
VcGantt1.Printer.PaperSize = VcPaperSize.vcDIN_A3
```

Example Code C#

```
vcGantt1.Printer.PaperSize = VcPaperSize.vcDIN_A3;
```

PrintDate

Property of VcPrinter

This property lets you set or retrieve, whether (True) or not (False) the print date is to appear in the bottom left corner of a page.

	Data Type	Explanation
Property value	System.Boolean	Print date is/is not set

Example Code VB.NET

```
VcGantt1.Printer.PrintDate = True
```

Example Code C#

```
vcGantt1.Printer.PrintDate = true;
```

PrinterName**Read Only Property of VcPrinter**

This property lets you set or retrieve the name of the currently selected printer. You can use this property for saving and restoring the state of the printer object.

If you transfer an empty string when setting the property, the system printer will be used.

<Tip:> Please note that the name of network printers has to be written in UNC notation, e.g. "\\server01\printer5".

	Data Type	Explanation
Property value	System.String	Printer name

PrintPreviewWithFirstPage**Property of VcPrinter**

This property lets you set or retrieve the mode of starting the page preview: either all pages of the diagram will be displayed (False) or only the first page will be displayed (True).

	Data Type	Explanation
Property value	System.Boolean	At the start of the page preview: only first page of the diagram (True) / all pages of the diagram (False)

Example Code VB.NET

```
Dim printer As VcPrinter
printer.Orientation = VcOrientation.vcLandscape
printer.PrintPreviewWithFirstPage = True
printer.FitToPage = False

VcGantt1.ShowPrintPreviewDialog()
```

Example Code C#

```
VcPrinter printer = vcGantt1.Printer;
printer.Orientation = VcOrientation.vcLandscape;
printer.PrintPreviewWithFirstPage = true;
printer.FitToPage = false;

vcGantt1.ShowPrintPreviewDialog();
```

ReOptimizeNodesInGroupsEnabled**Property of VcPrinter**

If the property **TimeScaleAdjustment** was set to true, this property allows to automatically update for the output or for the print preview the optimized arrangement of groups that are in the optimized state of display. This is only necessary if there are layers with text on the outside. The automatic optimization is very time-consuming and may lead to high response times in the print preview.

	Data Type	Explanation
Property value	System.Boolean	With the TimeScaleAdjustment property switched on: optimized groups are (True)/are not (False) reoptimized for output or print preview Default value: False

Example Code VB.NET

```
VcGantt1.Printer.ReOptimizeNodesInGroupsEnabled = True
```

Example Code C#

```
vcGantt1.Printer.ReOptimizeNodesInGroupsEnabled = true;
```

ScalingMode**Read Only Property of VcPrinter**

This property lets you set or retrieve the scaling mode for output. If the scaling mode is set to **vcZoomFactor**, the value of the property **ZoomFactor** defines the size of the output. If set to **vcFitToPageCount**, the values of **MaxHorizontalPagesCount** and **MaxVerticalPagesCount** are essential. If set to **vcZoomWithHorizontalFit**, the values of **ZoomFactor** and **MaxHorizontalPagesCount** define a zoom factor providing a fixed number of pages in width. The number of pages is maintained by downsizing or expanding the time scale. When using **vcZoomFactor** or **vcFitToPageCount**, you can achieve at covering the pages evenly by the property **AdjustTimeScale**.

	Data Type	Explanation
Property value	VcScalingMode Possible Values: .vcFitToPageCount 1 .vcZoomFactor 0 .vcZoomWithHorizontalFit 2	Scaling mode "Fit to Page" Scaling mode: "Zoomfactor". Scaling mode "Combined Fit"

TableColumnRanges

Property of VcPrinter

This property lets you set the number of table columns to be printed. Similar to Microsoft Word you can specify single columns or ranges of columns, that are to be separated by comas or semicolons. Example: "1;5-7;3" specifies the columns 1 and 3 and the range from 5 to 7. "0", a simple comma or semicolon will result in no column printed. By setting the default value -1 you can have all columns printed.

	Data Type	Explanation
Property value	System.String	Number of table columns which are printed Default value: empty string

Example Code VB.NET

```
VcGantt1.Printer.TableColumnRanges = "1;5-7;3"
```

Example Code C#

```
vcGantt1.TableColumnRanges = "1;5-7;3";
```

TableTimeScaleOnAllPages

Property of VcPrinter

This property lets you set or retrieve, whether (True) or not (False) the title, legend, table and time scale are to appear on each page.

	Data Type	Explanation
Property value	System.Boolean	Title, legend, table and time scale are repeated on each page (True)/ Title, legend, table and time scale are output only once and cut if necessary (False)

Example Code VB.NET

```
VcGantt1.Printer.TableTimeScaleOnAllPages = True
```

Example Code C#

```
vcGantt1.Printer.TableTimeScaleOnAllPages = true;
```

TableWidthAdoptionFromViewOnScreen**Property of VcPrinter**

This property lets you specify or retrieve whether the table width that is currently shown on the screen is to be adopted for the print preview and for the output.

This property can be also set in the **Page Layout** dialog.

	Data Type	Explanation
Property value	System.Boolean	The table width that is currently shown on the screen is (True) / is not (False) to be adopted for the print preview and for the output Default value: False

Example Code VB.NET

```
VcGantt1.Printer.TableWidthAdoptionFromViewOnScreen = True
```

TimeColumnEndDate**Property of VcPrinter**

This property lets you set or retrieve the end date of the time range to be used for the output. The time range can only be restricted in comparison to the time range displayed on the screen. Hence only an earlier start date than that having been set by the VcGantt property **TimeScaleStart** leads to a modified output.

This property can be also set in the **Page Layout** dialog.

	Data Type	Explanation
Property value	System.DateTime	End date of the time range shown in the output Default value: System.DateTime.MaxValue

TimeColumnStartDate

Property of VcPrinter

This property lets you set or retrieve the start date of the time range to be used for the output. The time range can only be restricted in comparison to the time range displayed on the screen. Hence only a later start date than that having been set by the VcGantt property **TimeScaleStart** leads to a modified output.

This property can be also set in the **Page Layout** dialog.

	Data Type	Explanation
Property value	System.DateTime	Start date of the time range range shown in the output Default value: System.DateTime.MinValue

TimeScaleAdjustment

Property of VcPrinter

This property improves utilization of the printing pages:

- If the scaling type **fit to page** is selected: The zoom factor is calculated in a way that utilizes the selected number of pages in the dimension of height. The time scale will be downsized or enlarged to adapt to the selected number of pages in the dimension of width.
- If **scaling by zoom factor** is selected: The time scale will be downsized or enlarged so that the selected number of pages is used to full capacity in the dimension of width.

	Data Type	Explanation
Parameter: ⇔ Rückgabewert	System.Boolean	Adjustment of time scale
Property value	System.Int32	Adjustment of time scale Default value: False

Example Code VB.NET

```
VcGantt1.Printer.TimeScaleAdjustment = True
```

Example Code C#

```
vcGantt1.TimeScaleAdjustment = true;
```

VcCalendarGrid

Property of VcPrinter

This property lets you set or retrieve the absolute height of the bottom margin of the pages to be printed in cm. The true width may be larger if the printer used has to print margins by obligation.

	Data Type	Explanation

Example Code VB.NET

```
VcGantt1.Printer.AbsoluteTopMarginInCM = 1.5 ' 2 cm
```

Example Code C#

```
vcGantt1.Printer.AbsoluteTopMarginInCM = 1.5; // 1.5 cm
```

ZoomFactorAsDouble

Property of VcPrinter

This property lets you set or retrieve the zoom factor for the scaling modes **VcZoomFactor** and **vcZoomWithHorizontalFit** to enlarge or downsize the output (zoom factor = 100: original size, zoom factor > 100: enlargement, zoom factor < 100: reduction).

	Data Type	Explanation
Property value	System.Double	Zoom factor of the diagram

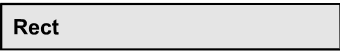
Example Code VB.NET

```
VcGantt1.Printer.ZoomFactor = 150
```

Example Code C#

```
vcGantt1.Printer.ZoomFactor = 150;
```

7.64 VcRect



An object of the type **VcRect** designates a rectangle object and is only available in VcInPlaceEditorShowing.

Properties

- Bottom
- Height
- Left
- Right
- Top
- Width

Properties

Bottom

Property of VcRect

This property returns/sets the bottom coordinate of the VcRect object.

	Data Type	Explanation
Property value	System.Int32	Position of the bottom border of the rectangle

Height

Read Only Property of VcRect

This property returns the height of the VcRect object.

	Data Type	Explanation
Property value	System.Int32	Height of the rectangle

Left

Property of VcRect

This property returns/sets the left coordinate of the VcRect object.

	Data Type	Explanation
Property value	System.Int32	Position of the left border of the rectangle

Example Code VB.NET

```

Private Sub VcGantt1_VcInPlaceEditorShowing(ByVal sender As Object, ByVal e
As NETRONIC.XGantt.VcInPlaceEditorShowingEventArgs) Handles
VcGantt1.VcInPlaceEditorShowing
    Dim node As VcNode
    node = e.EditObject
    If e.EditObjectType = VcObjectType.vcObjTypeNodeInTable Then
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse
        Select Case e.FieldIndex
            Case 1 'Name
                TextBox1.Left = e.FldRectVisible.Left + VcGantt1.Left
                TextBox1.Top = e.FldRectVisible.Top + VcGantt1.Top
                TextBox1.Width = e.FldRectVisible.Width
                TextBox1.Height = e.FldRectVisible.Height
                TextBox1.Text = node.DataField(0)
                TextBox1.Visible = True
                TextBox1.Focus()
            Case 2, 3 'Start or End
                DateTimePicker1.Left = e.FldRectVisible.Left + VcGantt1.Left
                DateTimePicker1.Top = e.FldRectVisible.Top + VcGantt1.Top
                DateTimePicker1.Value = node.DataField(0)
                DateTimePicker1.Visible = True
                DateTimePicker1.Focus()
            Case 13 'Employee
                ComboBox1.Left = e.FldRectVisible.Left + VcGantt1.Left
                ComboBox1.Top = e.FldRectVisible.Top + VcGantt1.Top
                ComboBox1.Width = e.FldRectVisible.Width
                ComboBox1.Height = e.FldRectVisible.Height
                ComboBox1.Text = node.DataField(0)
                ComboBox1.Visible = True
                ComboBox1.Focus()
        End Select
    End If
End Sub

```


Example Code C#

```

private void vcGantt1_VcInPlaceEditorShowing(object sender,
NETRONIC.XGantt.VcInPlaceEditorShowingEventArgs e)
{
    VcNode node = (VcNode)e.EditObject;
    if (e.EditObjectType == VcObjectType.vcObjTypeNodeInTable)
    {
        e.ReturnStatus = VcReturnStatus.vcRetStatFalse;
        switch (e.FieldIndex)
        {
            case 1: //Name
                textBox1.Left = e.FldRectVisible.Left + vcGantt1.Left;
                textBox1.Top = e.FldRectVisible.Top + vcGantt1.Top;
                textBox1.Width = e.FldRectVisible.Width;
                textBox1.Height = e.FldRectVisible.Height;
                textBox1.Text = Convert.ToString(node.get_DataField(0));
                textBox1.Visible = true;
                textBox1.Focus();
                break;
            case 2: //Start or end
                dateTimePicker1.Left = e.FldRectVisible.Left + vcGantt1.Left;
                dateTimePicker1.Top = e.FldRectVisible.Top + vcGantt1.Top;
                dateTimePicker1.Value = Convert.ToDateTime(node.get_DataField(0));
                dateTimePicker1.Visible = true;
                dateTimePicker1.Focus();
                break;
            case 13: //Employee
                comboBox1.Left = e.FldRectVisible.Left + vcGantt1.Left;
                comboBox1.Top = e.FldRectVisible.Top + vcGantt1.Top;
                comboBox1.Width = e.FldRectVisible.Width;
                comboBox1.Height = e.FldRectVisible.Height;
                comboBox1.Text = Convert.ToString(node.get_DataField(0));
                comboBox1.Visible = true;
                comboBox1.Focus();
                break;
        }
    }
}

```

Right**Property of VcRect**

This property returns/sets the right coordinate of the VcRect object.

	Data Type	Explanation
Property value	System.Int32	Position of the right border of the rectangle

Top**Property of VcRect**

This property returns/sets the top coordinate of the VcRect object.

	Data Type	Explanation
Property value	System.Int32	Position of the top border of the rectangle

Example Code VB.NET

```
DateTimePicker1.Top = e.FldRectVisible.Top + VcGantt1.Top
```

Example Code C#

```
dateTimePicker1.Top = e.FldRectVisible.Top + vcGantt1.Top;
```

Width

Read Only Property of VcRect

This property returns the width of the VcRect object.

	Data Type	Explanation
Property value	System.Int32	Width of the rectangle

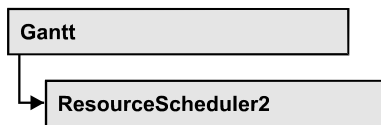
Example Code VB.NET

```
Text1.Width = fldRectVisible.Width
```

Example Code C#

```
textBox1.Width = e.FldRectVisible.Width;
```

7.65 VcResourceScheduler2



The ResourceScheduler2 is a substantial enhancement of ResourceScheduler1 (version 3.1). The different object types required for resource scheduling are now anticipated in data tables of their own, which was facilitated by version 4.0 of VARCHART XGantt. In contrast, ResourceScheduler1 merely allowed the different objects like tasks, operations, assignments and resources to be implicitly defined in the maindata table.

The below object types exist in ResourceScheduler2 and need to be defined in data tables of their own; resources may even be defined in up to 25 different tables:

- **Tasks:** These objects are composed by operations (see below) and hold basic properties such as the release date, the due date, priority and quantity.
- **Operations:** These objects can be assigned to resources (see below) by assignments (see below) and will contain the start and end dates of the processing time as a result of scheduling. Operations have a defined position within a sequence of their task and can be marked as "started". Beside, several different sequences of operations can be defined that represent mutually exclusive "routes" of processing. All operations of a route selected by the scheduling procedure will be scheduled.
- **Resources:** As their main features, these objects are part of a capacity curve and after scheduling, they also are part of a workload curve. Beside, they time the operations that they have received (timing resource). Therefore, in order to be scheduled, an operation needs to be assigned to a resource. Beside a timing resource, also work and material resources can be assigned to an operation. Another essential feature of a timing resource is its ability to be grouped on multiple levels. A timing resource may belong to different groups at one time.
- **Assignments:** These objects are the links between operations and resources, that allow to specify a factor for the quantity to be multiplied or divided. When groups of timing resources are scheduled, the assignments are marked correspondingly and additional assignments are generated for each single resource, so that they can be scheduled and displayed in VARCHART XGantt.

- **Links:** These objects describe the sequence of tasks, i.e., preceding tasks have to be finished before the succeeding ones can start.

Properties

- AssignmentDataTableName
- AssignmentIsResultFieldIndex
- AssignmentIsVisibleFieldIndex
- AssignmentLoadOrConsumptionPerItemFieldIndex
- AssignmentMaximumLoadFieldIndex
- AssignmentMinimumLoadFieldIndex
- AssignmentMinimumMaximumLoadType
- AssignmentOperationIDFieldIndex
- AssignmentResourceIDFieldIndex
- AssignmentResourceSelectionStrategyFieldIndex
- BaseCalendarUsageForSupplementTimes
- BaseTimeUnit
- BaseTimeUnitsPerStep
- DataRecordEventsEnabled
- DefaultOperationMaximumInterruptionTime
- DefaultResourceCalendarName
- FullUsageOfPlanningUnitsEnabled
- LinkDataTableName
- LinkDurationFieldIndex
- LinkPredecessorOperationIDFieldIndex
- LinkPredecessorTaskIDFieldIndex
- LinkSuccessorOperationIDFieldIndex
- LinkSuccessorTaskIDFieldIndex
- OperationDataTableName
- OperationLoadPerItemFieldIndex
- OperationMaximumInterruptionTimeFieldIndex
- OperationMinimumSupplementTimeFieldIndex
- OperationOverlapQuantityFieldIndex
- OperationPostLoadFieldIndex
- OperationPostOffsetFieldIndex
- OperationPreparationLoadFieldIndex
- OperationPreparationOffsetFieldIndex
- OperationResultEndDateFieldIndex
- OperationResultPostEndDateFieldIndex
- OperationResultPreparationStartDateFieldIndex

- OperationResultProcessingTimeFieldIndex
- OperationResultSelectedTimingResourceIDFieldIndex
- OperationResultStartDateFieldIndex
- OperationResultStatusFieldIndex
- OperationRouteFieldIndex
- OperationSequenceNumberFieldIndex
- OperationStartLockDateFieldIndex
- OperationTaskIDFieldIndex
- OperationWorkInProgressFieldIndex
- PlanningEndDate
- PlanningStartDate
- PlanningStrategy
- ResourceCalendarNameFieldIndex
- ResourceCapacityType
- ResourceCapacityTypeFieldIndex
- ResourceConstraintTypeFieldIndex
- ResourceDataTableName
- ResourceEfficiencyFieldIndex
- ResourceGroupDataTableName
- ResourceGroupIDFieldIndex
- ResourceNameFieldIndex
- ResourceResultLoadCurveNamePrefix
- ResourceResultStockCurveNamePrefix
- ResourceSelectionStrategy
- ResourceType
- ResultProcessingStepCount
- TaskDataTableName
- TaskDueDateFieldIndex
- TaskPlanningStrategyFieldIndex
- TaskPriorityFieldIndex
- TaskQuantityFieldIndex
- TaskReleaseDateFieldIndex
- TaskResultEndDateFieldIndex
- TaskResultPostEndDateFieldIndex
- TaskResultPreparationStartDateFieldIndex
- TaskResultProcessingStepFieldIndex
- TaskResultProcessingTimeFieldIndex
- TaskResultRouteFieldIndex
- TaskResultStartDateFieldIndex
- ToleranceTimeOnASAPDueDates

- ToleranceTimeOnJITReleaseDates
- ToleranceTimeOnStartLockDates
- WorkInProcessType
- WritingDebugFilesEnabled

Methods

- DetermineIDOfFirstOperationByTaskID
- DetermineIDOfLastOperationByTaskID
- Process

Properties

AssignmentDataTableName

Property of VcResourceScheduler2

This property lets you set or retrieve the name of the assignment data table that holds assignments of operations to resources. Setting this name is mandatory.

Administrate Data Tables

Data Tables

Name	Status	Multiple primary keys allowed	Description
Maindata	<input checked="" type="checkbox"/>		
Relations	<input type="checkbox"/>		
Task	<input type="checkbox"/>		
Operation	<input type="checkbox"/>		
Assignment	<input checked="" type="checkbox"/>		
Link	<input type="checkbox"/>		
TimingResource	<input type="checkbox"/>		
WorkResource	<input type="checkbox"/>		
ResourceGroup	<input type="checkbox"/>		

Data Table Fields

Index	Name	Primary key	Type	Date format	Editable	Hidden	Relationship
0	ID	<input checked="" type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
1	OperationID	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
2	ResourceID	<input checked="" type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
3	IsResult	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
4	IsVisible	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
5	MinimumLoad	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
6	MaximumLoad	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
7	LoadOrConsumptionPerItem	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

OK Close Apply Help

	Data Type	Explanation
Property value	System.String	Name of the assignment data table Default value: Empty string

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.AssignmentDataTableName = "Assignment"
```

Example Code C#

```
vcGantt1.ResourceScheduler2.AssignmentDataTableName("Assignment");
```

AssignmentIsResultFieldIndex

Property of VcResourceScheduler2

This property lets you set or retrieve the index of a data field in the assignment data table where VARCHART XGantt notes whether the corresponding data set was generated by itself. In the picture referring to **AssignmentDataTableName**, the field index for example is 3. Setting this property is optional. The scheduling procedure generates assignments only, if during the start among the existing assignments there are ones that refer to resource groups. Then the scheduling procedure generates an assignment to a resource that it selects from the group, and sets its corresponding field to 1. Assignments provided by the application either should not hold a value at all or should set it to 0.

Using this field allows for multiple invoking while the results are kept stable, which saves the application from having to manually re-set the assignments to their original state. The scheduling procedure continues to use assignments once generated in order to avoid dispensable actions of deleting and generating.

	Data Type	Explanation
Property value	System.Int32	Index of the data field in the assignment data table that is designated to hold the values on the identification of data records that were generated by resource scheduling. {-1...NumberOfFieldsInAssignmentDataTable -1}. By setting the index to -1, no data field of the assignment data table will be assigned to this property. Default value: -1

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.AssignmentIsResultFieldIndex = 3
```

Example Code C#

```
vcGantt1.ResourceScheduler2.AssignmentIsResultFieldIndex = 3;
```

AssignmentIsVisibleFieldIndex

Property of VcResourceScheduler2

This property lets you set or retrieve the index of a data field in the assignment data table where the resource scheduling module notes whether the assignment should be made visible. In the picture referring to **AssignmentDataTableName**, the field index for example is 4. The field is

useful for instance for displaying assignments to groups of resources in the Gantt graph before running the resource scheduling module, and for displaying the resulting single resources afterwards.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the assignment data table that is designated to hold the values on the visibility.</p> <p>{-1...NumberOfFieldsInAssignmentDataTable -1}. By setting the index to -1, no data field of the operations table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.AssignmentIsVisibleFieldIndex = 4
```

Example Code C#

```
vcGantt1.ResourceScheduler2.AssignmentIsVisibleFieldIndex = 4;
```

AssignmentLoadOrConsumptionPerItemFieldIndex

Property of VcResourceScheduler2

This property lets you set or retrieve the index of a data field in the assignment table which holds a value per item see property **TaskQuantityFieldIndex**). You can assign values per item to work resources and a material resources only. An index of -1 will be interpreted as 1. If the data field in the data set does not contain a valid value, 0 will be assumed. If the data field is of the type **String**, you can also enter a float value.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data set in the assignment data table that is designated to hold the value.</p> <p>{-1...NumberOfFieldsInOperationsDataTable -1}. By setting the index to -1, no data field of the operations table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.AssignmentLoadOrConsumptionPerItemFieldIndex = 7
```

Example Code C#

```
vcGantt1.ResourceScheduler2.AssignmentLoadOrConsumptionPerItemFieldIndex = 7;
```

AssignmentMaximumLoadFieldIndex

Property of VcResourceScheduler2

The index passed as the property value specifies a data field in the assignment data table that holds the maximum workload limit of a resource. In the picture referring to **AssignmentDataTableName**, the field index for example is 6.

This kind of limit can only be assigned to assignments of timing resources. The data field contains percentage values from {0...100}, where both, the value 0 and an empty field equal 100.

Values between 1 and 99 in the data field will disable the properties **FullUsageOfPlanningUnitsEnabled** and **OperationMaximumInterruption-TimeFieldIndex**.

Also see **AssignmentMinimumLoadFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the assignment data table that is designated to hold the maximum workload limit of a resource.</p> <p>{-1...NumberOfFieldsInAssignmentDataTable -1}. By setting the index to -1, no data field of the assignment data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.AssignmentMaximumLoadFieldIndex = 6
```

Example Code C#

```
vcGantt1.ResourceScheduler2.AssignmentMaximumLoadFieldIndex = 6;
```

AssignmentMinimumLoadFieldIndex

Property of VcResourceScheduler2

The index passed as the property value specifies a data field in the assignment data table that holds the minimum workload limit of a resource. In the picture referring to **AssignmentDataTableName**, the field index for example is 5.

The limit can only be assigned to timing resources. The data field contains percentage values from {0...100}. Also see **AssignmentMaximumLoadFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the assignment data table that is designated to hold the minimum workload limit of a resource.</p> <p>{-1...NumberOfFieldsInAssignmentDataTable -1}. By setting the index to -1, no data field of the assignment data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.AssignmentMinimumLoadFieldIndex = 5
```

Example Code C#

```
vcGantt1.ResourceScheduler2.AssignmentMinimumLoadFieldIndex = 5;
```

AssignmentMinimumMaximumLoadType

Property of VcResourceScheduler2

This property lets you set or retrieve whether the values that are assigned to the data fields by the indices set by the properties **AssignmentMinimumLoadFieldIndex** and **AssignmentMaximumLoadFieldIndex** are relative to the resource capacity or absolute.

Absolute values are useful e.g. if the assigned resource is a team with a varying number of persons and the assignment shall not occupy the whole team.

	Data Type	Explanation
Property value	VcResourceSchedulingMinimumMaximumLoadType	<p>Field values absolute/relative to resource capacity</p> <p>Default value: vcResSchedPercentageValues</p> <p>Possible Values:</p> <p>.vcResSchedAbsoluteValues 2</p> <p>.vcResSchedPercentageValues 0</p> <p>Data field values absolute to resource capacity</p> <p>Data field values relative to resource capacity</p>

AssignmentOperationIDFieldIndex

Property of VcResourceScheduler2

This property lets you set or retrieve the index to a data field in the assignment data table which holds the ID of an operation. In the picture referring to **AssignmentDataTableName**, the field index for example is 1. This property needs to be set to a figure unequal to -1 before calling the method **Process**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the assignment data table that is designated to hold the operation ID.</p> <p>{-1...NumberOfFieldsInAssignmentDataTable -1}. By setting the index to -1, no data field of the assignment data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.AssignmentOperationIDFieldIndex = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.AssignmentOperationIDFieldIndex = 1;
```

AssignmentResourceIDFieldIndex

Property of VcResourceScheduler2

This indexed property lets you set or retrieve the index of a data field in the assignment table that holds IDs of resources. In the picture referring to **AssignmentDataTableName**, the field index for example is 2.

The index passed as a parameter denotes one out of 25 resource tables. The ones used are set by the indexed property **ResourceDataTableName**.

	Data Type	Explanation
Parameter: ⇒ resourceTableIndex	System.Int16	<p>Index of a resource table according to the assignments made by ResourceDataTableName</p> <p>{0...24}</p>

Property value	System.Int32	Index of the data field in the assignment data table that is designated to hold resource IDs. {-1...NumberOfFieldsInAssignmentDataTable -1}. By setting the index to -1, no data field of the assignment data table will be assigned to this property. Default value: -1
-----------------------	--------------	---

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.AssignmentResourceIDFieldIndex(0) = 2
```

Example Code C#

```
vcGantt1.ResourceScheduler2.set_AssignmentResourceIDFieldIndex(0,2);
```

AssignmentResourceSelectionStrategyFieldIndex

Property of VcResourceScheduler2

The index passed as the property value specifies a data field in the assignment data table that defines a resource selection strategy for the respective assignment to a resource group. If this field is empty at a resource or the property is set to -1, the value of the general property **ResourceSelectionStrategy** is valid (see there).

The data field can contain the below list of values:

0: equals vcResSchedRSSequential

1: equals vcResSchedRSLeastLoaded

2: equals vcResSchedRSMostLoaded

3: equals vcResSchedRSHighestEfficiency

7: equals vcResSchedRSFirstAvailable

The values 1 and 2 (LeastLoaded and MostLoaded) entail consecutive adding of resource occupation that forms the base for selecting the resource loaded least or most. So if planning periods of tasks differ widely or if both planning strategies are applied, the results may not prove satisfactory.

When using the value 7 (FirstAvailable) the selection merely depends on the first timing resource. Other assignments of the operation are not taken into

consideration. So when using material and work resources, the results may not turn out satisfactory.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the assignment data table that is designated to hold the data of the planning strategy.</p> <p>{-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property.</p> <p>Default value: -1</p>

BaseCalendarUsageForSupplementTimes

Property of VcResourceScheduler2

If this property is set to **false**, no calendar will be used to define minimum supplement times (indirectly defined by the property **VcResourceScheduler2.OperationMinimumSupplementTimeFieldIndex**), so the time period specified will directly apply (example: could be used for drying produced parts). If this property is set to **true**, the base calendar of the Gantt object will be used with the supplement time being worked off as a working time defined in the base calendar (example: could be used for the transport of produced parts).

Please also see **VcResourceScheduler2.OperationMinimumSupplementTimeFieldIndex**.

	Data Type	Explanation
Property value	System.Boolean	<p>true: The base calendar of the Gantt object will be used.</p> <p>false: The specified time period will be used directly.</p> <p>Default value: false</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.BaseCalendarUsageForSupplementTimes = True
```

Example Code C#

```
vcGantt1.ResourceScheduler2.BaseCalendarUsageForSupplementTimes = true;
```

BaseTimeUnit

Property of VcResourceScheduler2

This property lets you set or retrieve the basic time unit for resource scheduling, which may differ from the basic time unit set by **VcGantt.TimeUnit**. The values of the capacity, work load and stock curves refer to the base unit defined here.

	Data Type	Explanation
Property value	VcTimeUnit	Time unit Default value: Value, which was set during design time by vcGantt.TimeUnit. If no setting was made, the value is vcDay .
	Possible Values: .vcDay 5 .vcHour 6 .vcMinute 7 .vcSecond 8	Time unit day Time unit hour Time unit minute Time unit second

Example Code VB.NET

```
VcResourceScheduler2.BaseTimeUnit = VcTimeUnit.vcMinute
VcResourceScheduler2.BaseTimeUnitsPerStep = 15
```

Example Code C#

```
vcGantt1.ResourceScheduler2.BaseTimeUnit = VcTimeUnit.vcMinute;
vcGantt1.ResourceScheduler2.BaseTimeUnitsPerStep = 15;
```

BaseTimeUnitsPerStep

Read Only Property of VcResourceScheduler2

This property lets you set or retrieve the size of steps of the scheduling. The larger this value, the faster, but also the coarser the result will be. The value entered here represents a multiple of the base unit set by **VcResourceScheduler2.BaseTimeUnit**.

	Data Type	Explanation
Property value	System.Int16	Number of time units per step Default value: 1

Example Code VB.NET

```
VcResourceScheduler2.BaseTimeUnit = VcTimeUnit.vcMinute
VcResourceScheduler2.BaseTimeUnitsPerStep = 30
```

Example Code C#

```
vcGantt1.ResourceScheduler2.BaseTimeUnit = VcTimeUnit.vcMinute;
vcGantt1.ResourceScheduler2.BaseTimeUnitsPerStep = 30;
```

DataRecordEventsEnabled**Property of VcResourceScheduler2**

If this property is set to **true**, events will be triggered that indicate data modifications during the process method: **DataRecordModifying**, **DataRecordModified**, **DataRecordCreating**, **DataRecordCreated**, **DataRecordDeleting** and **DataRecordDeleted**.

	Data Type	Explanation
Property value	System.Boolean	true : events are triggered. false : events are not triggered. Default value : false

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.DataRecordEventsEnabled = True
```

Example Code C#

```
vcGantt1.ResourceScheduler2.DataRecordEventsEnabled = true;
```

DefaultOperationMaximumInterruptionTime**Property of VcResourceScheduler2**

By this property you can set or retrieve a default value of the maximum time span, for which the operation is allowed be interrupted. The value is a number that represents base time units (see property **BaseTimeUnit**). The value applies if the property **OperationMaximumInterruptionTimeFieldIndex** was set to -1 or if the value read from the operations table equals 0 or if the field is empty. If the value is set to 0, no interruption is allowed.

This property will be disabled by setting the maximum load to less than 100% (see property **AssignmentMaximumLoadFieldIndex**).

	Data Type	Explanation
Property value	System.Int32	Number of base time units Default value : 0

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.DefaultOperationMaximumInterruptionTime = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.DefaultOperationMaximumInterruptionTime = 1;
```

DefaultResourceCalendarName**Property of VcResourceScheduler2**

This property lets you set a calendar name which is used if no calendar of the same name as the resource is found by the properties **VcResourceScheduler2.ResourceCalendarNameFieldIndex** and **VcResourceScheduler2.ResourceNameFieldIndex**. If you do not set the property, the resource will use the default calendar of the XGantt object. (see **VcCalendarCollection.Active**).

	Data Type	Explanation
Property value	System.String	Name of the calendar Default value: Empty String

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.DefaultResourceCalendarName = ""
```

Example Code C#

```
vcGantt1.ResourceScheduler2.DefaultResourceCalendarName = "";
```

FullUsageOfPlanningUnitsEnabled**Property of VcResourceScheduler2**

If this property is set to **True**, during the first and/or the last time unit of the occupation time of a resource allocated to a task, a second task may finish or start. This way, remaining capacities can be used up. If this property is set to **False**, remaining capacities will not be used.

This property merely influences the first operation of a task. It does not have any impact on the operations following.

This property will be disabled by setting the maximum load to less than 100% (see property **AssignmentMaximumLoadFieldIndex**).

	Data Type	Explanation
Property value	System.Boolean	true: remaining capacities are used. false: remaining capacities are not used. Default value: true

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.FullUsageOfPlanningUnitsEnabled = True
```

Example Code C#

```
vcGantt1.ResourceScheduler2.FullUsageOfPlanningUnitsEnabled = true;
```

LinkDataTableName

Property of VcResourceScheduler2

This property lets you set or retrieve the name of the linkData table, that holds links. If you do not set this name, links will not be taken into account during the run of the resource scheduling module.

The image shows a Windows-style dialog box titled "Administrative Data Tables". It contains two main sections: "Data Tables" and "Data Table Fields".

Data Tables Section:

Name	Status	Multiple primary keys allowed	Description
Maindata		<input checked="" type="checkbox"/>	
Relations		<input type="checkbox"/>	
Task		<input type="checkbox"/>	
Operation		<input type="checkbox"/>	
Assignment		<input type="checkbox"/>	
Link		<input type="checkbox"/>	

Data Table Fields Section:

Index	Name	Primary key	Type	Date format	Editable	Hidden	Relationship
0	ID	<input type="checkbox"/>	Integer		<input type="checkbox"/>	<input type="checkbox"/>	
1	Predecessor	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
2	Successor	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
3	Duration	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

At the bottom of the dialog are buttons for "OK", "Cancel", "Apply", and "Help".

	Data Type	Explanation
Property value	System.String	Name of the link data table Default value: Empty string

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.NodeDataTableName = "Node"
```

Example Code C#

```
vcGantt1.ResourceScheduler2.LinkDataTableName("Link");
```

LinkDurationFieldIndex

Property of VcResourceScheduler2

This property lets you set or retrieve the index of a data field in the link data table in which a minimum temporal distance between predecessor and successor can be stored. This distance can also be negative. Unit: as set by the method BaseTimeUnit. In the picture referring to **LinkDataTableName**, the field index for example is 3.

As a limit, when applying the planning strategy ASAP, a successor cannot start earlier than a predecessor ; when applying the planning strategy JIT, a predecessor cannot finish later than a successor.

	Data Type	Explanation
Property value	System.Int32	Index of the data field in the link data table that is designated to hold the values on the duration. {-1...NumberOfFieldsInLinkDataTable -1}. By setting the index to -1, no data field of the link data table will be assigned to this property. Default value: -1

LinkPredecessorOperationIDFieldIndex

Read Only Property of VcResourceScheduler2

This property lets you set or retrieve the index of a data field in the link data table the values of which contain the ID of the predecessor operation. As the resource scheduling module is only able to draw links between tasks, this property facilitates the use of links in XGantt which currently can only be displayed between operations. Thus the links are internally always created between the tasks of the operations specified by the ID.

When using a link data table, it is mandatory to set this property to a value not equal to -1 unless the VcResourceScheduler2 property **LinkPredecessorTaskIDFieldIndex** is used.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the link data table that is designated to hold the IDs of the predecessor operation.</p> <p>{-1...NumberOfFieldsInLinkDataTable -1}. By setting the index to -1, no data field of the link data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.LinkPredecessorOperationIDFieldIndex = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.LinkPredecessorOperationIDFieldIndex = 1;
```

LinkPredecessorTaskIDFieldIndex**Property of VcResourceScheduler2**

This property lets you set or retrieve the index of a data field in the link data table that holds the ID of the predecessor task. In the picture referring to **LinkDataTableName**, the field index for example is 1.

When using a link data table, it is mandatory to set this property to a value not equal to -1 unless the VcResourceScheduler2 property **LinkPredecessorOperationIDFieldIndex** is used.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the link data table that is designated to hold the IDs of the predecessor task.</p> <p>{-1...NumberOfFieldsInLinkDataTable -1}. By setting the index to -1, no data field of the link data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.NodePredecessorTaskIDFieldIndex = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.LinkPredecessorTaskIDFieldIndex = 1;
```

LinkSuccessorOperationIDFieldIndex

Read Only Property of VcResourceScheduler2

This property lets you set or retrieve the index of a data field in the link data table the values of which contain the ID of the successor operation. As the resource scheduling module is only able to draw links between tasks, this property facilitates the use of links in XGantt which currently can only be displayed between operations. Thus the links are internally always created between the tasks of the operations specified by the ID.

When using a link data table, it is mandatory to set this property to a value not equal to -1 unless the VcResourceScheduler2 property **LinkPredecessorTaskIDFieldIndex** is used.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the link data table that is designated to hold the IDs of the successor operation.</p> <p>{-1...NumberOfFieldsInLinkDataTable -1}. By setting the index to -1, no data field of the link data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.LinkSuccessorOperationIDFieldIndex = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.LinkSuccessorOperationIDFieldIndex = 1;
```

LinkSuccessorTaskIDFieldIndex

Property of VcResourceScheduler2

This property lets you set or retrieve the index of a data field in the link data table that contains the ID of the successor task. In the picture referring to **LinkDataTableName**, the field index for example is 2.

When using a link data table, it is mandatory to set this property to a value not equal to -1 unless the VcResourceScheduler2 property **LinkSuccessorOperationIDFieldIndex** is used.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the link data table that is designated to hold the IDs of successor tasks.</p> <p>{-1...NumberOfFieldsInLinkDataTable -1}. By setting the index to -1, no data field of the link data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.NodeSuccessorTaskIDFieldIndex = 2
```

Example Code C#

```
vcGantt1.ResourceScheduler2.LinkSuccessorTaskIDFieldIndex = 2;
```

OperationDataTableName

Property of VcResourceScheduler2

This property lets you set or retrieve the name of the operation data table that holds data of the operations. Setting this name is mandatory.

Administrate Data Tables

Data Tables

Name	Status	Multiple primary keys allowed	Description
Maindata		<input checked="" type="checkbox"/>	
Relations		<input type="checkbox"/>	
Task		<input type="checkbox"/>	
Operation	!	<input type="checkbox"/>	
Assignment		<input type="checkbox"/>	
Link		<input type="checkbox"/>	

Data Table Fields

Index	Name	Primary key	Type	Date format	Editable	Hidden	Relationship
0	ID	<input checked="" type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
1	Name	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
2	TaskID	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	Task:ID
3	StartDate	<input type="checkbox"/>	Date/Time	DD/MM/YY	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
4	EndDate	<input type="checkbox"/>	Date/Time	DD/MM/YY	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
5	StartLockDate	<input type="checkbox"/>	Date/Time	DD/MM/YY	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
6	MaxInterruption	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
7	LoadPerItem	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
8	OverlapQuantity	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
9	PreparationLoad	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
10	PostLoad	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
11	Route	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
12	WorkInProgress	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
13	ResultStartDate	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
14	ResultEndDate	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
15	ResultProcessingTime	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
16	ResultStatus	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

OK Cancel Apply Help

	Data Type	Explanation
Property value	System.String	Name of the operation data table Default value: Empty string

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationDataTableName = "Operation"
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationDataTableName("Operation");
```


OperationLoadPerItemFieldIndex

Property of VcResourceScheduler2

This property lets you set or retrieve the index of a data field in the operation data table that holds the load of a timing resource per item. To receive the total load on the timing resource, the value in the data field specified will be multiplied with the number specified by the task. If the data field holds an invalid value or if this property is set to -1, a value of 0 will be assumed.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the values of the load.</p> <p>{-1...NumberOfFieldsInOperationsDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationLoadPerItemFieldIndex = 10
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationLoadPerItemFieldIndex = 10;
```

OperationMaximumInterruptionTimeFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the operation data table to which a maximum time span is stored, for which the operation is allowed be interrupted. In the picture referring to **OperationDataTableName**, the field index for example is 9.

An interruption is a period free of activity on a resource that was fully loaded and allocated to an operation. It differs from a "break" by not being caused by a pre-defined workfree time.

The content of this field is a number that represents base time units (see property **BaseTimeUnit**).

If this property is set to -1 or if the value of the field equals zero or is empty, the value set by the property **DefaultOperationMaximumInterruptionTime** will be used. If the latter also equals 0, an interruption is not allowed. If the value is < 0, an interruption also is not allowed, even if the property **DefaultOperationMaximumInterruptionTime** does not equal 0.

This property will be disabled by setting the maximum load to less than 100% (see property **AssignmentMaximumLoadFieldIndex**).

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the maximum interruption time.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationMaximumInterruptionTimeFieldIndex = 9
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationMaximumInterruptionTimeFieldIndex = 9;
```

OperationMinimumSupplementTimeFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the operation data table to which a minimum supplement time of the operation is stored. During supplement time, the resources affected by this operation will not be occupied, so this time span can be used for standby or idle times.

The content of the designated field is a number that represents base time units (s. property **BaseTimeUnit**). In the picture referring to **OperationDataTableName**, the field index for example is 7.

Please also see **OperationMaximumSupplementLoadFieldIndex**, **OperationPreparationLoadFieldIndex** and **OperationPostLoadFieldIndex**

BaseCalendarUsageForSupplementTimes.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the minimum supplement time.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationMinimumSupplementTimeFieldIndex = 7
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationMinimumSupplementTimeFieldIndex = 7;
```

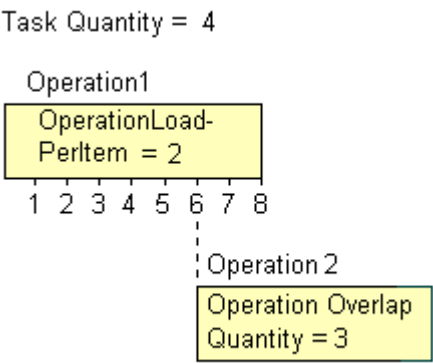
OperationOverlapQuantityFieldIndex

Property of VcResourceScheduler2

This property lets you set or retrieve the index of a data field in the operation data table that holds the 'overlap' quantity of an operation. Overlapping can only occur in tasks that were scheduled according to the strategy ASAP. This is the field to make succeeding resources overlap, which is useful if the succeeding operation does not have to wait for the preceding one to finish.

The quantity specified in the data field refers to the quantity of the task, set by the property **TaskQuantityFieldIndex**. The succeeding operation starts earliest after the preceding one has worked off the quantity specified (or later, optionally), overlapping the preceding one.

In the example below the value of the overlap field equals 3. It refers to the quantity of 4. After 3 units of those 4 units were worked off by operation1, operation2 will start. A possibly defined load per item for operation1 (in the below example =2) will be multiplied by the overlap value: $3 \times 2 = 6$. Therefore operation2 starts after operation has reached the value of 6.



Scenario sample: 4 candle sticks are to be produced, each one holding 3 candles. 2 candle sticks and 6 candles are put in a package. After 6 candles were produced by operation1, operation2 starts packing.

If the index set by the property is empty or if it contains a value = 0, the operation will not overlap the preceding one; if the value equals -1, the operation will start at the same time as the preceding one.

If a preparation time was defined, it will be taken into consideration within the overlapping period. So probably, the preparation time needs to be divided by the load per item of the operation (see **OperationLoadPerItemFieldIndex**) and added to the overlapping quantity. This property should not be used simultaneously with the property **ResourceEfficiencyFieldIndex**; the same is valid for **AssignmentMaximumLoadFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the values of the 'overlap' quantity.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationOverlapQuantityFieldIndex = 11
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationOverlapQuantityFieldIndex = 11;
```

OperationPostLoadFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the operation data table to which a post time of the operation is stored. During the post time, the resources affected by this operation will be occupied.

The content of the designated field is a number that represents the required capacity. In the picture referring to Please also see **OperationPreparation-TimeFieldIndex**, **OperationMaximumInterruptionTimeFieldIndex** and **OperationMinimumSupplementTimeFieldIndex**.

If you want to define post times that resource-independent you can use the property **OperationPostOffsetFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the post time.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationPostLoadFieldIndex = 13
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationPostLoadFieldIndex = 13;
```

OperationPostOffsetFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the operation data table to which a follow-up time of each operation is stored. If this field contains positive integers (in the current base time unit), the follow-up time of the operations is resource-independent. If the index equals -1, there are no follow-up times. This also applies if the index refers to a data field that contains a non-valid number or a 0 in the according operation.

Please also see **OperationPreparationOffsetFieldIndex**.

If you want to define resource-dependent post times you can use the property **OperationPostLoadFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that specifies whether the follow-up time of an operation is to be resource-independent.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. If the index is set to -1, there are no follow-up times.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationPostOffsetFieldIndex = 8
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationPostOffsetFieldIndex = 8;
```

OperationPreparationLoadFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the operation data table to which a preparation time of the operation is stored. During the preparation time, the resources affected by this operation will be occupied.

The content of the designated field is a number that represents the required capacity. In the picture referring to **OperationDataTableName**, the field index for example is 12.

Please also see **OperationPostLoadFieldIndex**, **OperationMaximum-InterruptionTimeFieldIndex** and **OperationMinimumSupplementTime-FieldIndex**.

If you want to define resource-independent post times that you can use the property **OperationPreparationOffsetFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the preparation time.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationPreparationLoadFieldIndex = 12
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationPreparationLoadFieldIndex = 12;
```

OperationPreparationOffsetFieldIndex**Property of VcResourceScheduler2**

The index specifies a data field in the operation data table to which a lead time of each operation is stored. If the field contains positive integers (in the current base time unit), the lead time of the operations is resource-independent. If the index equals -1, there are no lead times. This also applies if the index refers to a data field that contains a non-valid number or a 0 in the according operation.

Please also see **OperationPostOffsetFieldIndex**.

If you want to define resource-dependent preparation times you can use the property **OperationPreparationLoadFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	Index of the data field in the operation data table that specifies whether the lead time of an operation is to be resource-independent. {-1...NumberOfFieldsInOperationDataTable -1}. If the index is set to -1, there are no lead times. Default value: -1

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationPreparationOffsetFieldIndex = 8
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationPreparationOffsetFieldIndex = 8;
```

OperationResultEndDateFieldIndex**Property of VcResourceScheduler2**

This property lets you set or retrieve the index of a data field in the operation data table to which the calculated finish date of the operation is stored. In the picture referring to **OperationDataTableName**, the field index for example is 17.

To receive sensible results for the scheduling procedure, at least two out of the three properties **OperationResultStartDateFieldIndex**, **Operation-**

ResultProcessingTimeFieldIndex and **OperationResultEndDateFieldIndex** need to be set to a value unequal to -1.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the values of the end date.</p> <p>{-1...NumberOfFieldsInOperationsDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationResultEndDateFieldIndex = 17
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationResultEndDateFieldIndex = 17;
```

OperationResultPostEndDateFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the operation data table which holds the scheduled end date of the post time of an operation. If this phase is 0, the date is identical to the value in the data field which is referred to by the property **OperationResultEndDateFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the end date of the post time.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationResultPostEndDateFieldIndex = 15
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationPostEndDateFieldIndex = 15;
```


OperationResultPreparationStartDateFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the operation data table which holds the scheduled start date of the preparation phase of an operation. If the preparation phase is 0, this date is identical to the value in the data field which is referred to by the property **OperationResultStartDateFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the start date of the preparation phase.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationResultPreparationStartDateFieldIndex = 10
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationResultPreparationStartDateFieldIndex = 10;
```

OperationResultProcessingTimeFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the operation data table to which the calculated duration of the operation is stored. In the picture referring to **OperationDataTableName**, the field index for example is 18.

To receive sensible results for the scheduling procedure, at least two out of the three properties **OperationResultStartDateFieldIndex**, **OperationResultProcessingTimeFieldIndex** and **OperationResultEndDateFieldIndex** need to be set to a value unequal to -1.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the values of the processing time.</p> <p>{-1...NumberOfFieldsInOperationsDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationResultProcessingTimeFieldIndex = 18
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationResultProcessingTimeFieldIndex = 18;
```

OperationResultSelectedTimingResourceIDFieldIndex**Property of VcResourceScheduler2**

The index specifies a data field in the operation data table which holds the scheduled ID of a timing resource that was selected by the module. Thus in a table or a layer annotation the assigned resource can easily be shown graphically.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the ID of the timing resource that was selected by the module.</p> <p>{0...NumberOfFieldsInOperationsDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.OperationResultSelectedTimingResourceFieldIndex = 8
```

Example Code C#

```
vcGantt1.OperationResultSelectedTimingResourceFieldIndex = 8;
```

OperationResultStartDateFieldIndex**Property of VcResourceScheduler2**

The index specifies a data field in the operations table to which the calculated start date of the operation is stored. In the picture referring to **Operation-DataTableName**, the field index for example is 16.

To receive sensible results for the scheduling procedure, at least two out of the three properties **OperationResultStartDateFieldIndex**, **Operation-ResultProcessingTimeFieldIndex** and **OperationResultEndDateField-Index** need to be set to a value unequal to -1.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the values of the start date.</p> <p>{-1...NumberOfFieldsInOperationsDataTable -1}. By setting the index to -1, no data field of the operations table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationResultStartDateFieldIndex = 16
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationResultStartDateFieldIndex = 16;
```

OperationResultStatusFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the operation data table to which an error or a warning on scheduling the operation is stored. In the picture referring to **OperationDataTableName**, the field index for example is 19.

Possible values stored by the scheduling procedure:

0: the operation was scheduled

1: the operation was not scheduled because the scheduling procedure selected a different route of the task. This case can only occur if the property **OperationRouteFieldIndex** was set to a value unequal to -1.

1000: the operation was not scheduled

1001: the operation was not scheduled and it was an operation of a task causing the task not to be scheduled. The reasons for this can be various.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the error values.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationResultStatusFieldIndex = 19
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationResultStatusFieldIndex = 19;
```

OperationRouteFieldIndex

Property of VcResourceScheduler2

This property lets you set or retrieve the index of a data field in the operation data table the values of which assign operations to routes. In the picture referring to **OperationDataTableName**, the field index for example is 14.

Operations of the same content in this field belong to the same route. The content of this field also represents the name of the route.

Routes represent alternative ways to execute a task. The scheduling procedure checks the routes available and selects one for the task. This way, you can define several alternative operation sequences for the same task. Not more than 10 routes can be defined per task. The routes are selected in the sequence of their occurrence by the operations.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the name of the route.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationRouteFieldIndex = 14
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationRouteFieldIndex = 14;
```

OperationSequenceNumberFieldIndex

Property of VcResourceScheduler2

This property lets you set or retrieve the index of a data field in the operation data table the values of which define the sequence of the operations associated with a task. In the picture referring to **OperationDataTableName**, the field index for example is 6.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the sequence values.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationSequenceNumberFieldIndex = 6
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationSequenceNumberFieldIndex = 6;
```

OperationStartLockDateFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the operation data table that holds a start date for each operation in case of ASAP planning strategy (see property **PlanningStrategy**). In the picture referring to **OperationDataTableName**, the field index for example is 5.

If the data field contains a valid date, the task will be locked in the place of that start date and will not be moved by the scheduling procedure, which makes sense in particular for tasks already started. Please also see the property **OperationWorkInProgressFieldIndex**).

By the property **ToleranceTimeOnStartLockDates** you can set an allowance by which an operation may differ, i.e. a delay by which the lock date may be belated. Please mind that tasks that have operations with locked start dates are not scheduled automatically by first priority. If you wish this to happen, you need to calculate priorities of the tasks manually (see property **TaskPriorityFieldIndex**).

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the lock date.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationStartLockDateFieldIndex = 5
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationStartLockDateFieldIndex = 5;
```

OperationTaskIDFieldIndex

Property of VcResourceScheduler2

This property lets you set or retrieve the index of a data field in the operations table which holds the ID of the task that the operation belongs to. In the picture referring to **OperationDataTableName**, the field index for example is 2.

To have the operation scheduled, this property needs to be set to a value different from -1. The data field allows to assign several operations to a task. The sequence in which the operations of a task are scheduled depends on the value of the data field, the index of which is set by the property **OperationSequenceNumberDataFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the operation data table that is designated to hold the task ID.</p> <p>{-1...NumberOfFieldsInOperationDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationTaskIDFieldIndex = 2
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationTaskIDFieldIndex = 2;
```

OperationWorkInProcessFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the operation data table that contains a field which holds the degree of completion of an operation in percent, i.e. a value between 0 and 100. In the picture referring to **OperationDataTableName**, the field index for example is 15.

If the data field index was found to be -1 or no valid value can be provided by the field, 0% ("not started") will be assumed.

	Data Type	Explanation
Property value	System.Int32	Index of the data field in the operation data table that is designated to hold the degree of completion. {-1...NumberOfFieldsInOperationsDataTable -1}. By setting the index to -1, no data field of the operation data table will be assigned to this property. Default value: -1

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationWorkInProgressFieldIndex = 15
```

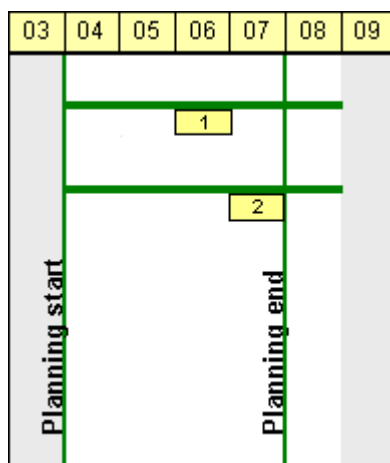
Example Code C#

```
vcGantt1.ResourceScheduler2.OperationWorkInProgressFieldIndex = 15;
```

PlanningEndDate

Property of VcResourceScheduler2

By this property you can set or retrieve the end date of the scheduling period. If you do not set this date, the end date will be taken from the end of the time scale, set by the property **VcGantt.TimeScaleEnd**. The start of the scheduling period can be set by **PlanningStartDate**.



Limited scheduling period

	Data Type	Explanation
Property value	System.DateTime	End date of the scheduling period Default value: DateTime.MinValue

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.PlanningEndDate = VcGantt1.TimeScaleEnd
```

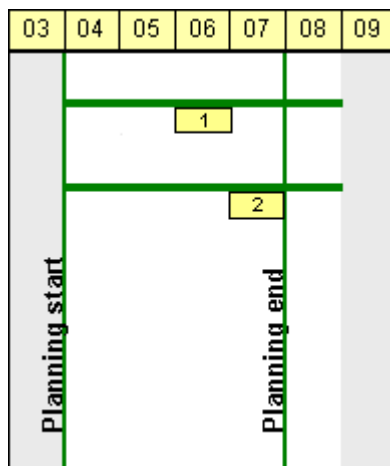
Example Code C#

```
vcGantt1.ResourceScheduler2.PlanningEndDate = vcGantt1.TimeScaleEnd;
```

PlanningStartDate

Property of VcResourceScheduler2

By this property you can set or retrieve the start date of the scheduling period. If you do not set this date, the start date will be taken from the start of the time scale, set by the property **VcGantt.TimeScaleStart**. The end of the scheduling period can be set by **PlanningEndDate**.



Limited scheduling period

	Data Type	Explanation
Property value	System.DateTime	Start date of the scheduling period Default value: DateTime.MinValue

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.PlanningStartDate = VcGantt1.TimeScaleStart
```

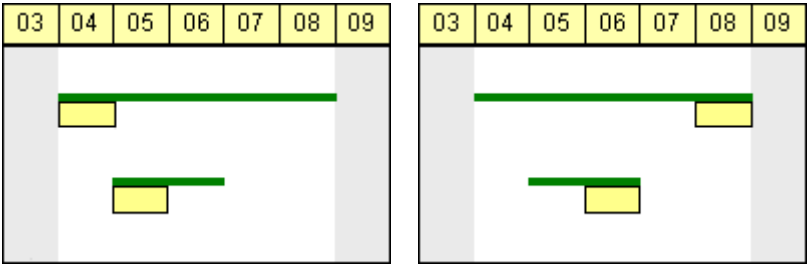
Example Code C#

```
vcGantt1.ResourceScheduler2.PlanningStartDate = vcGantt1.TimeScaleStart;
```


PlanningStrategy

Property of VcResourceScheduler2

This property specifies the planning strategy for tasks. Two options exist for planning strategies: One strategy aims at working off tasks as fast as possible to achieve a high turnover in the production system. Therefore, tasks start as soon as possible (ASAP). The other strategy aims at finishing tasks duely, for example to keep stocks low. Therefore, tasks finish just in time (JIT).



So in the ASAP strategy the start is early (picture left), while in the JIT strategy the finish is late (picture right). The long slim bars show the available period to complete a task, while the short big bars represent the actually allocated time for completion. So ASAP tasks tend to appear at the left end of the available period of completion, while JIT tasks tend to appear at its right end.

If an individual setting of the planning strategy per task is required, you can assign a data field by **TaskPlanningStrategyFieldIndex** to individually overwrite settings of **PlanningStrategy**.

	Data Type	Explanation
Property value	VcResourceSchedulingPlanningStrategy	Planning strategy
		Default value: vcResSchedPSASAP
	Possible Values: .vcResSchedPSASAP -1 .vcResSchedPSJIT 0	As soon as possible Just in time

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.PlanningStrategy =  
VcResourceSchedulingPlanningStrategy.vcResSchedPSASAP
```

Example Code C#

```
vcGantt1.ResourceScheduler2.PlanningStrategy =  
VcResourceSchedulingPlanningStrategy.vcResSchedPSASAP;
```

ResourceCalendarNameFieldIndex

Property of VcResourceScheduler2

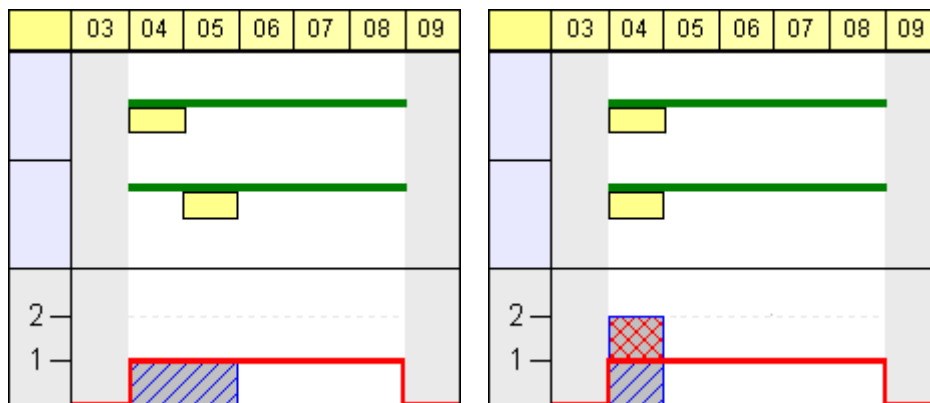
The index passed as the property value specifies a data field in the resource data table that defines the name of a calendar for a resource of the type **TimingResource** or **WorkResource**. If the field of the resource is empty, if it contains an invalid name or if this property is set to -1, as a substitute the name of the resource will be used for the calendar name, as set by the property **ResourceNameFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	Index of the data field in the resource data table that defines a calendar name for the resource of the type TimingResource or WorkResource . Default value: -1

ResourceCapacityType

Property of VcResourceScheduler2

This property sets the capacity type for all resources, if it is not set individually for each resource by **ResourceCapacityTypeFieldIndex**.



Finite capacities (left) may require tasks to be allocated sequentially while infinite capacities (right) allow to schedule them simultaneously.

	Data Type	Explanation
Parameter: ⇒ resourceTableIndex	System.Int16	Index of the resource data table. {0...24}
Property value	VcResourceCapacityType	Capacity types Default value: vcResSchedCTFinite

Possible Values:	
.vcResSchedCTFinite -1	Finite capacities
.vcResSchedCTInfinite 0	Infinite capacities

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ResourceCapacityType =  
VcResourceSchedulingCapacityType.vcResSchedCTFinite
```

Example Code C#

```
vcGantt1.ResourceScheduler2.ResourceCapacityType =  
VcResourceSchedulingCapacityType.vcResSchedCTFinite;
```

ResourceCapacityTypeFieldIndex

Property of VcResourceScheduler2

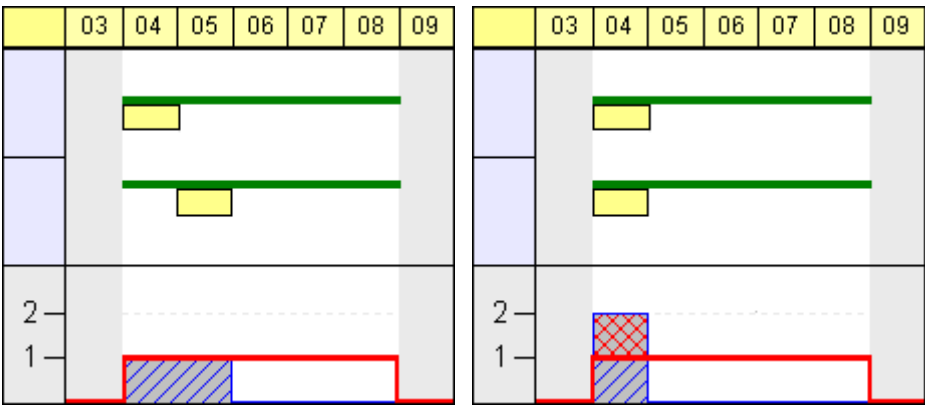
This property lets you set or retrieve the index of a field in a resource data table that holds the capacity type of a single timing resource. In the picture referring to **ResourceDataTableName**, the field index for example is 2.

The index passed as a parameter denotes one out of 25 resource tables. The one to be used is defined by the indexed property **ResourceDataTableName**.

Permitted values of the data field content:

1 finite capacity

2 infinite capacity



Finite capacities (left) may require tasks to be allocated sequentially while infinite capacities (right) allow to work them off simultaneously. By the property **ResourceCapacityType** you can set the capacity for all data records of a resource table. The latter property is overwritten by this property if set.

If a resource belongs to more than one group, it has to have the same capacity type in all groups.

	Data Type	Explanation
Parameter: ⇒ resourceTableIndex	System.Int16	Index of the resource data table {0...24}
Property value	System.Int32	Index of the data field in the resource data table that is designated to hold the capacity type. {-1...NumberOfFieldsInResourceDataTable -1}. By setting the index to -1, no data field of the resource data table will be assigned to this property. Default value: -1

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ResourceCapacityTypeFieldIndex(0) = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.set_ResourceCapacityTypeFieldIndex(0,1);
```

ResourceConstraintTypeFieldIndex

Property of VcResourceScheduler2

The index passed as the property value specifies a data field in the resource data table that holds a constraint for a single work or material resource.

Among the 25 possibly existing resource tables the one sought for is referred to by the index passed as the parameter.

As types, the values 0,1 or 3 or no value may be specified. The values "" or "1" or no field indicate, that the given capacity of the resource is truly valid (this is what is called a "hard" resource).

The value "0" indicates, that the given capacity of the resource may be ignored if there is an increasing demand for it, since it then would be available by an unlimited capacity ("soft" resource).

The value "3" indicates that the resource is "hard", but workfree periods will be taken into account which do not cause interruptions when the operation is scheduled.

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the resource table {0...24}
Property value	System.Int32	Index of the data field in the resource data table that is designated to hold the constraint data. {-1...NumberOfFieldsInResourceDataTable -1}. By setting the index to -1, no data field of the resource data table will be assigned to this property. Default value: -1

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ResourceConstraintTypeFieldIndex(0) = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.set_ResourceConstraintTypeFieldIndex(0,1);
```

ResourceDataTableName**Property of VcResourceScheduler2**

This property lets you set or retrieve the names of up to 25 resource data tables. The name at the index 0 is to be set by obligation. If more than one name is set, the indices need to be stocked continuously without a gap from 0 onward. For each resource data table set by this property a corresponding field has to be allocated in the assignment data table by the property **AssignmentResourceIDFieldIndex**.

Administrate Data Tables

Data Tables

Name	Status	Multiple primary keys allowed	Description
Maindata		<input checked="" type="checkbox"/>	
Relations		<input type="checkbox"/>	
Task		<input type="checkbox"/>	
Operation		<input type="checkbox"/>	
Assignment		<input type="checkbox"/>	
Link		<input type="checkbox"/>	
TimingResource	!	<input type="checkbox"/>	
WorkResource		<input type="checkbox"/>	

Data Table Fields

Index	Name	Primary key	Type	Date format	Editable	Hidden	Relationship
0	ID	<input checked="" type="checkbox"/>	Integer		<input type="checkbox"/>	<input type="checkbox"/>	
1	Name	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
2	CapacityType	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
3	ResourceGroupID	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

OK Cancel Apply Help

Administrate Data Tables

Data Tables

Name	Status	Multiple primary keys allowed	Description
Maindata		<input checked="" type="checkbox"/>	
Relations		<input type="checkbox"/>	
Task		<input type="checkbox"/>	
Operation		<input type="checkbox"/>	
Assignment		<input type="checkbox"/>	
Link		<input type="checkbox"/>	
TimingResource		<input type="checkbox"/>	
WorkResource		<input type="checkbox"/>	

Data Table Fields

Index	Name	Primary key	Type	Date format	Editable	Hidden	Relationship
0	ID	<input checked="" type="checkbox"/>	Integer		<input type="checkbox"/>	<input type="checkbox"/>	
1	Name	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
2	Constraint	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

OK Cancel Apply Help

	Data Type	Explanation
Parameter: ⇒ resourceTableIndex	System.Int16	Index of the resource table {0...24}
Property value	System.String	Name of the data table Default value: Empty string

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ResourceDataTableName(1) = "Timing Resource"
```

Example Code C#

```
vcGantt1.ResourceScheduler2.set_ResourceDataTableName(1, "Timing Resource");
```

ResourceEfficiencyFieldIndex

Property of VcResourceScheduler2

The index passed as the property value specifies a data field in the resource data table that indicates an efficiency in percent for the resource of the type **TimingResource**. If this field of a resource is empty or if the property is set to -1, the efficiency by default equals 100. If however a value is set, the total of the allocations is multiplied by the efficiency value by assigning before scheduling this resource. So if the efficiency is lower than 100 per cent, an operation assigned to this resource will take longer than the default whereas values above 100 per cent will cause an assigned operation to be worked off faster than could the default. This is particularly interesting regarding the definition of resource groups (please see also **ResourceSelectionStrategy**), where from the available resources the one of greatest efficiency can be selected.

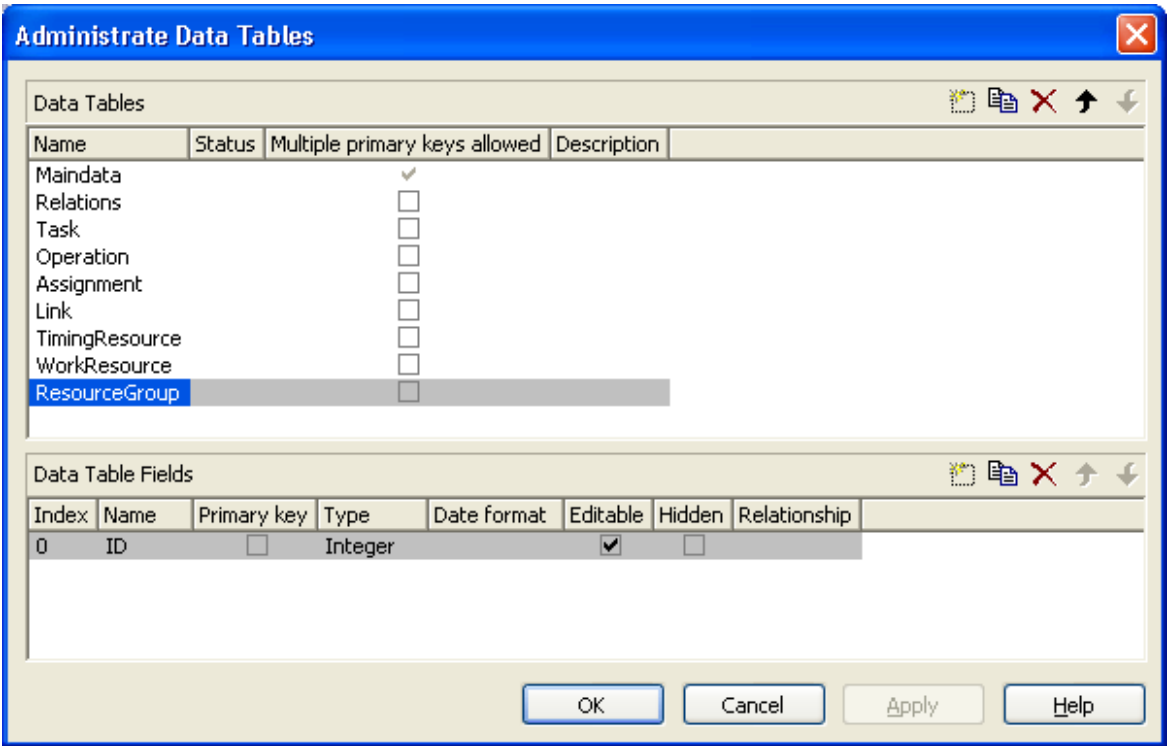
Being a percentage, the values of efficiency in general range between 1 and 100. Values of > 1,000 automatically will be put back to 1,000. The efficiency should NOT be set to a value as high as to reduce the occupation of a resource below 1.

	Data Type	Explanation
Property value	System.Int32	Index of the data field in the resource data table that indicates the efficiency in per cent of a resource of the type Timing Resource .

ResourceGroupDataTableName

Property of VcResourceScheduler2

This indexed property lets you set or retrieve the data table in which the resource groups can be found, of which the IDs are held by fields referred to by **ResourceGroupIDFieldIndex**. So for each field index that you specify by the property **ResourceGroupIDFieldIndex**, you need to set the name of a data field by this property. It uses the same data tables as does **ResourceDataTableName**. The resource data table index passed as the parameter denotes one out of 25 available resource data tables assigned by the indexed property **ResourceDataTableName**.



	Data Type	Explanation
Parameter: ⇒ ResourceGroupTableIndex	System.Int16	Index of the resource group data table. {0...24}
Property value	System.String	Name of the resource group data table

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ResourceGroupDataTableName(1) = "Printer Resource"
```

Example Code C#

```
vcGantt1.ResourceScheduler2.set_ResourceGroupDataTableName(1, "Printer Resource");
```


ResourceGroupIDFieldIndex

Property of VcResourceScheduler2

The index passed as the property value specifies a data field in the resource data table that is designated to hold the ID of a group resource. By setting the ID, the resource is described as one belonging to the group. In the picture referring to **ResourceGroupDataTableName**, the field index for example is 0. If the field index is set to -1 or if the resource data field referred to is empty, the resource will not belong to a group. This property must only be set to timing resources (see property **ResourceType**).

The index passed as a parameter denotes one out of 25 resource tables. They can be set by the indexed property **ResourceDataTableName**.

	Data Type	Explanation
Parameter: ⇨ resourceTableIndex	System.Int16	Index of the resource table. {0...24}
Property value	System.Int32	Index of the data field in the resource data table that is designated to hold the groupID. {-1...NumberOfFieldsInResourceDataTable -1}. By setting the index to -1, no data field of the resource data table will be assigned to this property. Default value: -1

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ResourceGroupIDFieldIndex(0) = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.set_ResourceGroupIDFieldIndex(0,1);
```

ResourceNameFieldIndex

Property of VcResourceScheduler2

The index passed as the property value specifies a data field in the resource data table that holds the names of resources. In the picture referring to **ResourceDataTableName**, the field index for example is 1.

The resource name serves to identify histogram names, curve names and calendar names. Beside, it is used with groups to allocate a resource to several groups simultaneously. For this, a resource needs to be specified in different data records by the same name but by different IDs of the group

resources. If no field index is specified, names of histograms, curves and calendars will be retrieved on the base of the resource ID.

The index passed as a parameter denotes one out of 25 resource tables. The resource tables used can be set by the indexed property **ResourceDataTableName**.

	Data Type	Explanation
Parameter: ⇒ resourceTableIndex	System.Int16	Index the resource table. {0...24}
Property value	System.Int32	Index of the data field in the resource data table that is designated to hold the name. {-1...NumberOfFieldsInResourceDataTable -1}. By setting the index to -1, no data field of the resource data table will be assigned to this property. Default value: -1

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ResourceNameFieldIndex(0) = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.set_ResourceNameFieldIndex(0,1);
```

ResourceResultLoadCurveNamePrefix

Property of VcResourceScheduler2

Prefix for the name of the curve that after the scheduling procedure contains the resource capacity for each timing resource and for each work and material resource.

The curves for the work load need to have been defined before invoking the method **Process**, otherwise they cannot be visualized. The resource name (see property **ResourceNameFieldIndex**) or the resource ID will be used to form the remaining part of the name. If a curve is not found, the results of the work load will be lost for the resource affected.

Beside, the property **CurveSource** needs to have been set to **vcSetCurve** for the curves, i.e. assignments must be feasible by the **VcCurve.SetValues** method of the API.

	Data Type	Explanation
Property value	System.String	Character string that contains the prefix Default value: "Load_"

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ResourceResultLoadCurveNamePrefix = "LoadCurve_"
```

Example Code C#

```
vcGantt1.ResourceScheduler2.ResourceResultLoadCurveNamePrefix = "LoadCurve_";
```

ResourceResultStockCurveNamePrefix

Property of VcResourceScheduler2

Prefix for the name of the curve that after the scheduling procedure contains the available stock of each material resource.

The stock curves need to have been defined before invoking the method **Process**, otherwise they cannot be visualized. The resource name or the resource ID will be used to form the remaining part of the name.

If a curve is not found, the results of the stock will be lost for the resource affected. The available stock is calculated from the cumulation of material supply (that is, from the supply curve that has to be put up before the scheduling procedure starts) and from the utilization by the operations that were assigned to the resource.

Beside, the property **CurveSource** needs to have been set to **vcSetCurve** for the curves, i.e. assignments must be feasible by the **VcCurve.SetValues** method of the API.

	Data Type	Explanation
Property value	System.String	Character string that contains the prefix Default value: "Stock_"

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ResourceResultStockCurveNamePrefix = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.ResourceResultStockCurveNamePrefix = 1;
```

ResourceSelectionStrategy

Property of VcResourceScheduler2

This property specifies the selection strategy of the scheduling process for resources to be selected from a group (therefore for timing resources only).

	Data Type	Explanation
Property value	VcResourceSchedulingResourceSelectionStrategy	Selection types
		Default value: VcResSchedRSSequential
	Possible Values:	
	.vcResSchedRSFirstAvailable 6	The resource which is first available when the scheduling is performed will be selected if its available capacity permits. When using this constant, the selection merely depends on the first timing resource. Other assignments of the operation are not taken into account. So when using material and work resources, the results may not turn out satisfactory.
	.vcResSchedRSHighestEfficiency 2	The resource most efficient when the scheduling is performed will be selected (makes sense only if the property ResourceEfficiencyFieldIndex is used) if its available capacity permits.
	.vcResSchedRSLeastLoaded 0	The resource least loaded when the scheduling is performed will be selected, if its available capacity permits. This strategy is useful if the workload is to be distributed evenly between resources. This value entails consecutive adding of resource occupation that forms the base for selecting the resource least loaded. So if planning periods of tasks differ widely or if both planning strategies are applied, the results may not prove satisfactory.
	.vcResSchedRSMostLoaded 1	The resource most loaded when the scheduling is performed will be selected, if its available capacity permits. This strategy is useful if the workload is to be concentrated on as few resources as possible. This value entails consecutive adding of resource occupation that forms the base for selecting the resource most loaded. So if planning periods of tasks differ widely or if both planning strategies are applied, the results may not prove satisfactory.
	.vcResSchedRSSequential -1	The resources are tried to be used in the sequence defined.

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ResourceSelectionStrategy =  
VcResourceSchedulingResourceSelectionStrategy.vcResSchedRSLeastLoaded
```

Example Code C#

```
vcGantt1.ResourceScheduler2.ResourceSelectionStrategy =  
VcResourceSchedulingResourceSelectionStrategy.vcResSchedRSLeastLoaded;
```

ResourceType

Property of VcResourceScheduler2

This property lets you set or retrieve the type of a resource data table. The index passed specifies one of the 25 possibly existing resource data tables. Three possible resource types exist:

1. Timing Resources

For a resource to time an operation, the operation needs to be assigned to exactly one resource. Both, finite and infinite capacity types are permitted (s. property **ResourceCapacityTypeFieldIndex**). Resources of this type can be grouped (s. properties **ResourceGroupDataTableName** and **ResourceGroupIDFieldIndex**). Beside, the work load of the resource can be limited (s. properties **AssignmentMinimumLoadFieldIndex** and **AssignmentMaximumLoadFieldIndex**). A timing resource requires capacity curves as an indirect resource information and uses work load curves to put the results (s. properties **ResourceNameFieldIndex** and **ResourceResultLoadCurvePrefix**).

2. Work Resources

This resource type shows two particular features. An operation can be assigned to more than one resource of this type. As the timing type, the work type requires capacity curves as an indirect source of information and uses work load curves to put the results (s. properties **ResourceNameFieldIndex** and **ResourceResultLoadCurvePrefix**).

3. Material Resources

The material resource also shows two characteristic features. An operation can be assigned to more than one resource of this type. The material resource differs by its source of indirect information and the result output: it requires supply curves as an indirect source of information and uses stock curves to

put the results (s. properties **ResourceNameFieldIndex** and **Resource-ResultStockCurvePrefix**).

	Data Type	Explanation
Parameter: ⇒ resourceTableIndex	System.Int16	Index of the resource data table {0...24}.
Property value	VcResourceSchedulingResourceType Possible Values: .vcMaterial 1 .vcTiming -1 .vcWork 0	Type of the resource data table Default value: vcTiming The resource type is "material". The resource type is "timing". The resource type is "work".

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ResourceType(0) =
VcResourceSchedulingResourceType.vcResSchedTiming
```

Example Code C#

```
vcGantt1.ResourceScheduler2.set_ResourceType(0,
VcResourceSchedulingResourceType.vcResSchedTiming);
```

ResultProcessingStepCount

Property of VcResourceScheduler2

This property provides the number of scheduled tasks in the chart after a scheduling procedure.

	Data Type	Explanation
Property value	System.Int32	Number of tasks Default value: 0

Example Code VB.NET

```
Dim i As Integer
i = VcGantt1.ResourceScheduler2.ResultProcessingStepCount()
```

Example Code C#

```
int i = vcGantt1.ResourceScheduler2.ResultProcessingStepCount;
```

TaskDataTableName

Property of VcResourceScheduler2

This property lets you set or retrieve the name of the task data table. A valid table name has to be used with the property.

Administrate Data Tables

Data Tables

Name	Status	Multiple primary keys allowed	Description
Maindata		<input checked="" type="checkbox"/>	
Relations		<input type="checkbox"/>	
Task	!	<input type="checkbox"/>	
Operation		<input type="checkbox"/>	
Assignment		<input type="checkbox"/>	
Link		<input type="checkbox"/>	
TimingResource		<input type="checkbox"/>	
WorkResource		<input type="checkbox"/>	
ResourceGroup		<input type="checkbox"/>	

Data Table Fields

Index	Name	Primary key	Type	Date format	Editable	Hidden	Relationship
0	ID	<input checked="" type="checkbox"/>	Integer		<input type="checkbox"/>	<input type="checkbox"/>	
1	Name	<input type="checkbox"/>	String		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
2	Release date	<input type="checkbox"/>	Date/Time	DD/MM/YY	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
3	Due date	<input type="checkbox"/>	Date/Time	DD/MM/YY	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
4	Quantity	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
5	Priority	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
6	PlanningStrategy	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
7	ResultStartDate	<input type="checkbox"/>	Date/Time	DD/MM/YY	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
8	ResultEndDate	<input type="checkbox"/>	Date/Time	DD/MM/YY	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
9	ResultRoute	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
10	ResultProcessingTime	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
11	ResultProcessingStep	<input type="checkbox"/>	Integer		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

OK Cancel Apply Help

	Data Type	Explanation
Property value	System.String	Name of the task data table Default value: Empty string

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.TaskDataTableName = "Task"
```

Example Code C#

```
vcGantt1.ResourceScheduler2.TaskDataTableName("Task");
```

TaskDueDateFieldIndex**Property of VcResourceScheduler2**

The index specifies a data field in the tasks data table which holds the due date at which a task must be finished. If no valid value is found in the data field, the value set by the VcGantt property **TimeScaleEnd** will be used. If you wish the task to be scheduled, the value of this property must not be set to -1. In the picture referring to **TaskDataTableName**, the field index for example is 3.

To due dates, a general allowance can be set by the property **ToleranceTime-OnASAPDueDates**. Please mind that tasks that have a close due date or only a short period between the release date and the due date are not scheduled automatically by first priority. If you wish this to happen, you need to calculate the priorities of the tasks manually (see property **TaskPriority-FieldIndex**).

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the task data table that is designated to hold the due date.</p> <p>{-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.TaskDueDateFieldIndex = 3
```

Example Code C#

```
vcGantt1.ResourceScheduler2.TaskDueDateFieldIndex = 3;
```

TaskPlanningStrategyFieldIndex**Property of VcResourceScheduler2**

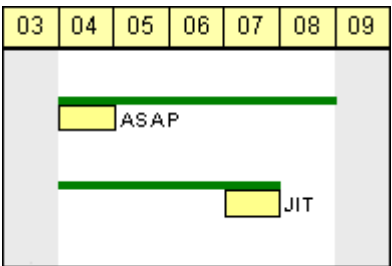
The index specifies a data field which holds an individual planning strategy for a task.

If no value is set or if the value is < 1 or > 2, the value set by the property **Planning Strategy** will be used. In the picture referring to **TaskDataTable-Name**, the field index for example is 6.

Defined values of data fields {1...2}:

1 - ASAP: as soon as possible

2 - JIT: just in time



In the ASAP strategy a task is scheduled early, while in the JIT strategy it is scheduled late. The long slim bars show the available period to complete a task, while the short big bars represent the actually allocated time for completion. So ASAP tasks tend to appear at the left end of the available period of completion, while JIT tasks tend to appear at its right end.

	Data Type	Explanation
Property value	System.Int32	Index of the data field in the task data table that is designated to hold the data of the planning strategy. {-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property. Default value: -1

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.TaskPlanningStrategyFieldIndex = 6
```

Example Code C#

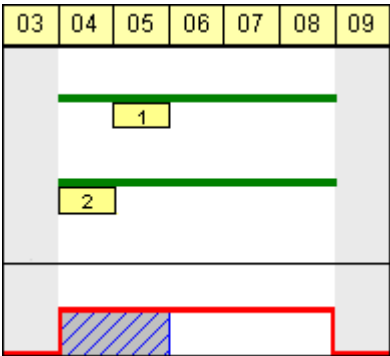
```
vcGantt1.ResourceScheduler2.TaskPlanningStrategyFieldIndex = 6;
```

TaskPriorityFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the task data table which holds a priority for a task.

The higher the priority value, the better the activity is positioned in the queue of scheduling.



A priority 2 task will be scheduled before a priority 1 task.

Please note: If tasks are linked, their priorities should be set very carefully. When using the ASAP strategy, predecessors should have the same priority as their successors; when using the JIT strategy, predecessors should have at least the same priority as their successors. Tasks can be grouped by their priorities. For example, when grouping tasks of equal priority, preparation and cleaning times of the device may be saved.

	Data Type	Explanation
Property value	System.Int32	Index of the data field in the task data table that is designated to hold the priority. {-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property. Default value: -1

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.TaskPriorityFieldIndex = 5
```

Example Code C#

```
vcGantt1.ResourceScheduler2.TaskPriorityFieldIndex = 5;
```

TaskQuantityFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the tasks data table which holds the quantity to be worked off by a task. The value of this property must not be set to -1.

The quantity indirectly influences the amount of time required by the task to finish. The amount of time can also be influenced by the efficiency of the resources (see **ResourceEfficiencyFieldIndex**), by multipliers of operations (see **OperationLoadPerItemFieldIndex**) and of assignments (see **AssignmentLoadOrConsumptionPerItemFieldIndex**).

If no valid value is found in the data field, a quantity of 1 will be assumed. In the picture referring to **TaskDataTableName**, the field index for example is 4.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the task data table that is designated to hold the quantity.</p> <p>{-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.TaskQuantityFieldIndex = 4
```

Example Code C#

```
vcGantt1.ResourceScheduler2.TaskQuantityFieldIndex = 4;
```

TaskReleaseDateFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the tasks data table which holds the release date from which onward a task can be scheduled. The value of this property must not be set to -1.

If no valid value is found in the data field, the value set by the VcGantt property **TimeScaleStart** will be used. In the picture referring to **TaskDataTableName**, the field index for example is 2.

You can set a general allowance to release dates by the property **Tolerance-TimeOnJITReleaseDates**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the task data table that is designated to hold the release date.</p> <p>{-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.TaskReleaseDateFieldIndex = 2
```

Example Code C#

```
vcGantt1.ResourceScheduler2 .TaskReleaseDateFieldIndex = 2;
```

TaskResultEndDateFieldIndex**Property of VcResourceScheduler2**

The index specifies a data field in the tasks data table which holds the calculated end date of the latest operation scheduled that is part of the task. In the picture referring to **TaskDataTableName**, the field index for example is 8.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the task data table that is designated to hold the end date.</p> <p>{-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.TaskResultEndDateFieldIndex = 8
```

Example Code C#

```
vcGantt1.ResourceScheduler2.TaskResultEndDateFieldIndex = 8;
```

TaskResultPostEndDateFieldIndex**Property of VcResourceScheduler2**

The index specifies a data field in the task data table which holds the scheduled end date of the post time of an operation. If the post time is 0, the date is identical to the value in the data field which is referred to by the property **TaskResultEndDateFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the task data table that is designated to hold the end date of the post time.</p> <p>{-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationResultPostEndDateFieldIndex = 15
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationPostEndDateFieldIndex = 15;
```

TaskResultPreparationStartDateFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the task data table which holds the scheduled start date of the preparation phase of a task. If the preparation phase is 0, the date is identical to the value in the data field which is referred to by the property **TaskResultStartDateFieldIndex**.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the task data table that is designated to hold the start date of the preparation phase.</p> <p>{-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.OperationResultPreparationStartDateFieldIndex = 10
```

Example Code C#

```
vcGantt1.ResourceScheduler2.OperationResultPreparationStartDateFieldIndex = 10;
```

TaskResultProcessingStepFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the tasks data table which holds a sequence number by which the task was scheduled. This value is useful to recognize the first task that cannot be scheduled due to resource bottlenecks.

The task scheduled first will receive 0, the tasks following will receive the consecutive numbers in ascending order. In the picture referring to **TaskDataTableName**, the field index for example is 11.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the task data table that is designated to hold the sequence number.</p> <p>{-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.TaskResultProcessingStepFieldIndex = 11
```

Example Code C#

```
vcGantt1.ResourceScheduler2.TaskResultProcessingStepFieldIndex = 11;
```

TaskResultProcessingTimeFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the tasks data table which holds the calculated total processing time of the operations that form the task and that were scheduled. It is the time span between the start date of the first operation and the final date of the last operation. Units: as set by the base time unit. In the picture referring to **TaskDataTableName**, the field index for example is 10.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the task data table that is designated to hold the processing time.</p> <p>{-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.TaskResultProcessingTimeFieldIndex = 10
```

Example Code C#

```
vcGantt1.ResourceScheduler2.TaskResultProcessingTimeFieldIndex = 10;
```

TaskResultRouteFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the tasks data table which holds the name of a route that was selected for the task by the scheduling procedure.

The value of this property should be set to a value different from -1, if the property **OperationRouteFieldIndex** is also used. In the picture referring to **TaskDataTableName**, the field index for example is 9.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the task data table that is designated to hold the name of the route.</p> <p>{-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.TaskResultRouteFieldIndex = 9
```

Example Code C#

```
vcGantt1.ResourceScheduler2.TaskResultRouteFieldIndex = 9;
```

TaskResultStartDateFieldIndex

Property of VcResourceScheduler2

The index specifies a data field in the tasks data table which holds the calculated start date of the earliest operation scheduled that is part of the task. In the picture referring to **TaskDataTableName**, the field index for example is 7.

	Data Type	Explanation
Property value	System.Int32	<p>Index of the data field in the task data table that is designated to hold the start date.</p> <p>{-1...NumberOfFieldsInTaskDataTable -1}. By setting the index to -1, no data field of the task data table will be assigned to this property.</p> <p>Default value: -1</p>

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.TaskResultStartDateFieldIndex = 7
```

Example Code C#

```
vcGantt1.ResourceScheduler2.TaskResultStartDateFieldIndex = 7;
```

ToleranceTimeOnASAPDueDates**Property of VcResourceScheduler2**

By this property you can set or retrieve an allowance to due dates. It only works with the ASAP planning strategy. The unit equals the one set by the property **BaseTimeUnit**.

During the scheduling procedure, the due dates of the tasks are postponed by the number of units set by this property, prolonging the period of time allowed to a task. This property is useful to detect whether after enlarging the scheduling period all operations and tasks could be scheduled. It saves you from modifying and testing tasks individually.

Please also see **ToleranceTimeOnJITReleaseDates**.

	Data Type	Explanation
Property value	System.Int32	Number of base time units {>=0} Default value: 0

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ToleranceTimeOnASAPDueDates = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.ToleranceTimeOnASAPDueDates = 1;
```

ToleranceTimeOnJITReleaseDates**Property of VcResourceScheduler2**

By this property you can set or retrieve a variation allowed to release dates. This setting only works if the JIT planning strategy is set. The unit equals what was set by the property **BaseTimeUnit**.

During the scheduling procedure, the release dates of the tasks are put earlier by the number of units set by this property, prolonging the period of time allowed to a task. This property is useful to detect what scheduling periods are needed for all tasks to be scheduled. It saves you from modifying the release dates of tasks individually.

Please also see **ToleranceTimeOnASAPDueDates**.

	Data Type	Explanation
Property value	System.Int32	Number of base time units {>=0} Default value: 0

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ToleranceTimeOnJITReleaseDates = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.ToleranceTimeOnJITReleaseDates = 1;
```

ToleranceTimeOnStartLockDates

Property of VcResourceScheduler2

By this property you can set or retrieve an allowance to a locked start date of an operation (see **OperationStartLockDateFieldIndex**). Its unit equals the one set by the property **BaseTimeUnit**.

During the scheduling procedure, an operation can be postponed by the number of units set by this property, if the resources to be occupied are not available at the lock start date.

	Data Type	Explanation
Property value	System.Int32	Number of base time units {>=0} Default value: 0

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.ToleranceTimeOnStartLockDates = 1
```

Example Code C#

```
vcGantt1.ResourceScheduler2.ToleranceTimeOnStartLockDates = 1;
```

WorkInProgressType

Property of VcResourceScheduler2

This property sets the unit to specify the degree of completion (please see **OperationWorkInProgressFieldIndex**).

	Data Type	Explanation
Property value	VcResourceSchedulingWorkInProgressType	Unit of the degree of completion Default value: vcResSchedWIPPercentage
	Possible Values:	

```
.vcResSchedWIPCompleted 0
.vcResSchedWIPPercentage -1
.vcResSchedWIPRemaining 1
```

```
Unit: quantity already completed
Unit: percentage (0...100)
Unit: quantity to be completed
```

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.WorkInProcessType =
VcResourceSchedulingWorkInProcessType.vcResSchedWIPCompleted
```

Example Code C#

```
vcGantt1.ResourceScheduler2.WorkInProcessType =
VcResourceSchedulingWorkInProcessType.vcResSchedWIPCompleted;
```

WritingDebugFilesEnabled

Property of VcResourceScheduler2

If this property is set to **true**, a debug file named **OPS_debug.txt** will be stored to the current directory, which may be useful for error analysis.

	Data Type	Explanation
Property value	System.Boolean	true : debug files can be written into the current directory. false : debug files cannot be written into the current directory. Default value : false

Example Code VB.NET

```
VcGantt1.ResourceScheduler2.WritingDebugFilesEnabled = True
```

Example Code C#

```
vcGantt1.ResourceScheduler2.WritingDebugFilesEnabled = true;
```

Methods

DetermineIDOfFirstOperationByTaskID

Method of VcResourceScheduler2

This method determines the ID of the first operation of a task by the given TaskID and helps the developer updating the data field of a link which contains the first operation of a task.

For further information please see the description of the VcResourceScheduler2 properties **LinkPredecessorOperationIDFieldIndex** and **LinkSuccessorOperationIDFieldIndex**.

	Data Type	Explanation
Parameter: ⇒ taskID	System.String	ID of a task of the corresponding data table which was set by the VcResourceScheduler2 property TaskDataTableName .
Return value	System.String	ID of the first operation of the corresponding data table which was set by the VcResourceScheduler2 property OperationDataTableName .

DetermineIDOfLastOperationByTaskID

Method of VcResourceScheduler2

This method determines the ID of the last operation of a task by the given TaskID and helps the developer updating the data field of a link which contains the last operation of a task.

For further information please see the description of the VcResourceScheduler2 properties **LinkPredecessorOperationIDFieldIndex** and **LinkSuccessorOperationIDFieldIndex**.

	Data Type	Explanation
Parameter: taskID	System.String	ID of a task of the corresponding data table which was set by the VcResourceScheduler2 property TaskDataTableName .
Return value	System.String	ID of the last operation of the corresponding data table which was set by the VcResourceScheduler2 property OperationDataTableName .

Process

Method of VcResourceScheduler2

This method starts the scheduling procedure after the desired properties were set. For messages on the progress please also see **ResourceScheduling-Progressing**. **OnResourceSchedulingProgress**. Beside, warnings are put out by **ResourceSchedulingWarning**.

	Data Type	Explanation
Return value	System.Boolean	<p>true: No error occurred during the scheduling procedure.</p> <p>false: An error occurred or the scheduling procedure was abandoned.</p> <p>If the settings allow, error codes may have been stored for each job by the data field addressed by the property OperationResultStatusFieldIndex .</p>

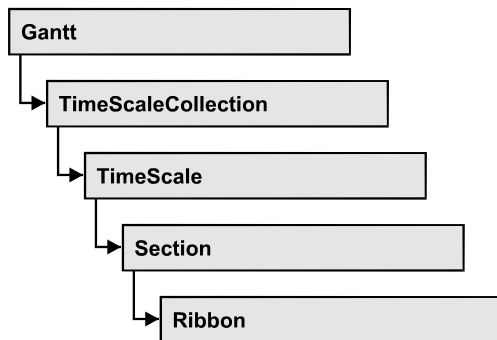
Example Code VB.NET

```
VcGantt1.ResourceScheduler2.Process()
```

Example Code C#

```
vcGantt1.ResourceScheduler2.Process();
```

7.66 VcRibbon



An object of the type VcRibbon represents a defined ribbon in the time scale of homogeneous units and scaling. You can set the background color, the type of unit separation, font type, color, size, alignment and other attributes to a ribbon.

Properties

- CalendarName
- DateOutputFormat
- Font
- FontColor
- MajorTicks
- MinorTicks
- ObserveDST
- PatternBackgroundColorAsARGB
- PatternColorAsARGB
- PatternEx
- Position
- ReferenceDate
- TextAlignment
- TickColor
- TickPosition
- Type
- UnitSeparation
- UseReferenceDate

Properties

CalendarName

Property of VcRibbon

This property lets you set or retrieve the calendar name.

	Data Type	Explanation
Property value	System.String	Calendar name

DateOutputFormat

Property of VcRibbon

This property lets you specify the date output format of a ribbon. To compose the date you can use the below codes:

- D: first letter of the day of the week (not adjustable)
- TD: Day of the Week (adjustable by using the event **VcTextEntrySupplying**)
- DD: two-digit figure for the day of the month: 01-31
- DDD: first three letters of the day of the week (not adjustable)
- M: first letter of the name of the month (not adjustable)
- TM: name of the month (adjustable by using the event **VcTextEntrySupplying**)
- MM: two-digit figure for the month: 01-12
- MMM: first three letters of the name of the month (not adjustable)
- YY: two-digit figure for the year
- YYYY: four-digit figure for the year
- WW: two-digit figure for the number of the calendar week: 01-53
- TW: text for "calendar week" (adjustable by using the event **VcTextEntrySupplying**)
- Q: one-digit figure for the quarter: 1-4
- TQ: name of quarter (adjustable by using the event **VcTextEntrySupplying**)

- hh: two-digit figure for the hour in 24 hours format: 00-23
- HH: two-digit figure for the hour in 12 hours format: 01-12
- Th: Text of "o' clock" (adjustable by using the event **VcTextEntrySupplying**)
- TH: "am" or "pm" (adjustable by using the event **VcTextEntrySupplying**)
- mm two-digit figure for the minute: 00-59
- ss: two-digit figure for the second: 00-59
- TS: short date format, as defined in the regional settings of the windows control panel
- TL: long date format, as defined in the regional settings of the windows control panel
- TT: time format, as defined in the regional settings of the windows control panel
- xC/XC: You can set a maximum ten-place, simple upward counting from a reference date onward, for example "15:05:07:16:00", which equals 15 months, 5 days, 7 hours, 16 minutes, 0 seconds. The notation is: **xC44:C33:C22:C11:C00**. In written language: Show at least 2 digits for the counters 4...0 and a preceding "-" symbol if the value is negative. The separators are variable and can be replaced by other separators symbols. "x" means: Display a preceding "-" symbol if the value is negative, but no "+" symbol if it is positive. "X" means: Display a preceding "-" symbol if the value is negative and a "+" symbol for positive values. In the dialog **Edit Time Scale Section...** the check boxes **Use reference date** and **Adjust major ticks to reference date** need to be ticked, also, the parameter **Serial annotation** has to be set to **No**. In the application the reference date is set at run time by the call **VcRibbon.set ReferenceDate**, overriding any settings in the dialog.

Note: Characters which are not to be interpreted as part of the date should be preceded by a backslash '\'. '\\' for instance results in \'. The special characters: ':, /, -' and **blank** don't need '\' as prefix.

	Data Type	Explanation
Property value	System.String	Date format {DMYhms:;/{}

Example Code VB.NET

```
Dim timeScale As VcTimeScale
Dim ribbon As VcRibbon

timeScale = VcGantt1.TimeScaleCollection.Active
ribbon = timeScale.Ribbon(0, 0)
ribbon.DateOutputFormat = "DD.MMM.YYYY hh:mm:ss"
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcRibbon ribbon = timeScale.get_Ribbon(0,0);
ribbon.DateOutputFormat = "DD.MMM.YYYY hh:mm:ss";
```

Font

Property of VcRibbon

This property lets you set or retrieve all font attributes of the ribbon.

	Data Type	Explanation
Property value	System.Drawing.Font	Font attributes of the ribbon

Example Code VB.NET

```
Dim timeScale As VcTimeScale
Dim ribbon As VcRibbon
Dim newFont As Font

timeScale = VcGantt1.TimeScaleCollection.Active
ribbon = timeScale.Ribbon(0, 0)
newFont = New Font("Times New Roman", 14, FontStyle.Italic)
ribbon.Font = newFont
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcRibbon ribbon = timeScale.get_Ribbon(0,0);
Font newFont = new Font("Times New Roman", 14, FontStyle.Italic);
ribbon.Font = newFont;
```

FontColor

Property of VcRibbon

This property lets you set or retrieve the font color of the ribbon.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255}

Example Code VB.NET

```
Dim timeScale As VcTimeScale
Dim ribbon As VcRibbon

timeScale = VcGantt1.TimeScaleCollection.Active
ribbon = timeScale.Ribbon(0, 0)
ribbon.FontColor = Color.Blue
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcRibbon ribbon = timeScale.get_Ribbon(0,0);
ribbon.FontColor = Color.LightSteelBlue;
```

MajorTicks

Property of VcRibbon

This property lets you set or retrieve after how many time units a major tick is drawn. The time unit depends on the ribbon type used. The major ticks are labelled when there is enough space. This property you can also set in the **Edit Time Scale Section** dialog.

	Data Type	Explanation
Property value	System.Int32	Number of units between two major ticks

Example Code VB.NET

```
Dim timeScale As VcTimeScale
Dim ribbon As VcRibbon

timeScale = VcGantt1.TimeScaleCollection.Active
ribbon = timeScale.Ribbon(0, 0)
ribbon.MajorTicks = 7
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcRibbon ribbon = timeScale.get_Ribbon(0,0);
ribbon.MajorTicks = 7;
```

MinorTicks

Property of VcRibbon

This property lets you set or retrieve after how many time units a minor tick is drawn. The time unit depends on the ribbon type used. The minor ticks are

not labelled. This property you can also set in the **Edit Time Scale Section** dialog.

	Data Type	Explanation
Property value	System.Int32	Number of units between two minor ticks

Example Code VB.NET

```
Dim timeScale As VcTimeScale
Dim ribbon As VcRibbon

timeScale = VcGantt1.TimeScaleCollection.Active
ribbon = timeScale.Ribbon(0, 0)
ribbon.MinorTicks = 1
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcRibbon ribbon = timeScale.get_Ribbon(0,0);
ribbon.MinorTicks = 1;
```

ObserveDST

Property of VcRibbon

This property lets you set or retrieve whether for this ribbon daylight saving time is considered or not.

	Data Type	Explanation
Property value	VcRibbonObserveDST	Daylight saving time is/is not considered.
	Possible Values: .vcGODDefault 9999 .vcRODNo 0 .vcRODYes 1	Default setting from .INI file is used Daylight saving time is not considered Daylight saving time is considered

PatternBackgroundColorAsARGB

Property of VcRibbon

This property lets you set or retrieve the background color of the ribbon. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color. When casting an RGB value on an ARGB value, an alpha value of 255 has to be added.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255})

Example Code VB.NET

```
Dim timeScale As VcTimeScale
Dim ribbon As VcRibbon

Set timeScale = VcGantt1.TimeScaleCollection.Active
Set ribbon = timeScale.Ribbon(0, 0)
ribbon.PatternBackgroundColorAsARGB = &h88FF0A06
```

PatternColorAsARGB**Property of VcRibbon**

This property lets you set or retrieve the pattern color of the ribbon. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color. When casting an RGB value on an ARGB value, an alpha value of 255 has to be added.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255})

Example Code VB.NET









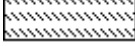
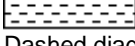
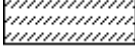



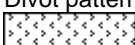
```
Dim timeScale As VcTimeScale
Dim ribbon As VcRibbon

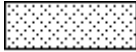
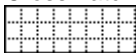
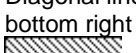
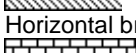
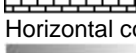
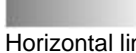



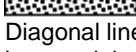

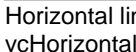
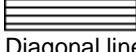
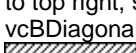
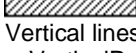


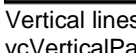

Set timeScale = VcGantt1.TimeScaleCollection.Active
Set ribbon = timeScale.Ribbon(0, 0)
ribbon.PatternColorAsARGB = &h88FF0A06
```

PatternEx**Property of VcRibbon**

This property lets you set or retrieve the pattern of the ribbon background.

	Data Type	Explanation
Property value	VcFillPattern	Pattern type
	Possible Values:	

.vc05PercentPattern...	Dots in foreground color on background color, the density of the foreground color increasing with the percentage
.vc90PercentPattern 01 - 11	
.vcAeroGlassPattern 44	Vertical color gradient in the color of the fill pattern 
.vcBDiagonalPattern 5	Diagonal lines slanting from bottom left to top right 
.vcCrossPattern 6	Cross-hatch pattern 
.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width 
.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 
.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width 
.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 
.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
.vcDashedHorizontalPattern 2026	Dashed horizontal lines 
.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 
.vcDashedVerticalPattern 2027	Dashed vertical lines 
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small 
.vcDiagonalBrickPattern 2032	Diagonal brick pattern 
.vcDivotPattern 2036	Divot pattern 

.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines 
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines 
.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right 
.vcHorizontalBrickPattern 2033	Horizontal brick pattern 
.vcHorizontalGradientPattern 52	Horizontal color gradient 
.vcHorizontalPattern 3	Horizontal lines 
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern 
.vcLargeConfettiPattern 2029	Confetti pattern, large 
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern 
.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern 
.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern 
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern 
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern 
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern 
.vcNoPattern 1276	No fill pattern 
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large 
.vcPlaidPattern 2035	Plaid pattern 
.vcShinglePattern 2039	Diagonal shingle pattern 
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern 

.vcSmallConfettiPattern 2028	Confetti pattern
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern
.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares
.vcSpherePattern 2041	Checkerboard of spheres
.vcTrellisPattern 2040	Trellis pattern
.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright
.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright
.vcVerticalGradientPattern 62	Vertical color gradient
.vcVerticalPattern 2	Vertical lines
.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark
.vcWavePattern 2031	Horizontal waves pattern
.vcWeavePattern 2034	Interwoven stripes pattern
.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern
.vcWideUpwardDiagonalPattern 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of vcBDiagonalPattern
.vcZigZagPattern 2030	Horizontal zig-zag lines

Position

Property of VcRibbon

This property lets you set or retrieve the position of the ribbon.

	Data Type	Explanation
Property value	VcRibbonPosition Possible Values: .vcRPBottom 2 .vcRPNone 0 .vcRPTop 1	Ribbon position bottom none top

ReferenceDate

Property of VcRibbon

This property lets you set or retrieve the reference date.

	Data Type	Explanation
Property value	System.DateTime	Reference date

TextAlignment

Property of VcRibbon

This property lets you set or retrieve the alignment of the major ticks of the ribbon.

	Data Type	Explanation
Property value	VcHorizontalRibbonTextAlignment Possible Values: .vcRTAtTickAligned 1039 .vcRTHorCenterAligned -1 .vcRTLeftAligned -3 .vcRTRightAligned -2	Positioned above the tick, centered between two ticks, left aligned, right aligned Text placed at tick Text horizontally centered between two major ticks Text left aligned between two major ticks Text right aligned between two major ticks

Example Code VB.NET

```

Dim timeScale As VcTimeScale
Dim ribbon As VcRibbon

timeScale = VcGantt1.TimeScaleCollection.Active
ribbon = timeScale.Ribbon(0, 0)
ribbon.TextAlignment = VcHorizontalRibbonTextAlignment.vcRTLLeftAligned

```

Example Code C#

```

VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcRibbon ribbon = timeScale.get_Ribbon(0,0);
ribbon.TextAlignment = VcHorizontalRibbonTextAlignment.vcRTLLeftAligned;

```

TickColor

Property of VcRibbon

This property lets you set or retrieve the color of ticks.

	Data Type	Explanation
Property value	System.Drawing.Color RGB ({0...255},{0...255},{0...255})	RGB color values ({0...255},{0...255},{0...255}) Default value: 0,0,0

TickPosition

Property of VcRibbon

This property lets you set or retrieve the tick position.

	Data Type	Explanation
Property value	VcRibbonTickPosition Possible Values: .vcTPAbove 1044 .vcTPBelow 1045	Tick position above below

Type

Property of VcRibbon

This property lets you set or retrieve the ribbon type. The types available are listed below.

	Data Type	Explanation
Property value	VcRibbonType	Ribbon type
	Possible Values: .vcDayRibbon 5 .vcFiscalQuarterRibbon 3002 .vcFiscalYearRibbon 3001 .vcHourRibbon 6 .vcMinuteRibbon 7 .vcMonthRibbon 3 .vcQuarterRibbon 10 .vcSecondRibbon 9 .vcShiftRibbon 8 .vcWeekRibbon 4 .vcYearRibbon 1	Ribbon showing days' units Ribbon showing fiscal quarters' units Ribbon showing fiscal years' units Ribbon showing hours' units Ribbon showing minutes' units Ribbon showing months' units Ribbon showing quarters' units Ribbon showing seconds' units Ribbon showing shifts Ribbon showing weeks' units Ribbon showing years' units

Example Code VB.NET

```
Dim timeScale As VcTimeScale
Dim ribbon As VcRibbon

timeScale = VcGantt1.TimeScaleCollection.Active
ribbon = timeScale.Ribbon(0, 0)
ribbon.Type = VcRibbonType.vcWeekRibbon
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcRibbon ribbon = timeScale.get_Ribbon(0,0);
ribbon.Type = VcRibbonType.vcWeekRibbon;
```

UnitSeparation**Property of VcRibbon**

This property lets you set or retrieve the appearance of the major ticks of the ribbon. A full line, a tick and no line are the features available.

	Data Type	Explanation
Property value	VcUnitSeparation	Appearance of the major tick
	Possible Values: .vcUSFullLine 4 .vcUSNone 1 .vcUSTick 1035	Units separated by full lines Units not separated Units separated by ticks

Example Code VB.NET

```
Dim timeScale As VcTimeScale
Dim ribbon As VcRibbon

timeScale = VcGantt1.TimeScaleCollection.Active
ribbon = timeScale.Ribbon(0, 0)
ribbon.UnitSeparation = VcUnitSeparation.vcUSTick
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;  
VcRibbon ribbon = timeScale.get_Ribbon(0,0);  
ribbon.UnitSeparation = VcUnitSeparation.vcUSTick;
```

UseReferenceDate**Property of VcRibbon**

This property lets you set or retrieve whether the ribbon uses a reference date.

	Data Type	Explanation
Property value	System.Boolean	The ribbon uses (True) / does not use (False) reference date Default value: False

7.67 VcScheduler

Scheduler

An object of the type **VcScheduler** represents a module for calculating simple project data, such as the early end of a project or its early start (if calculations are performed backward), or its free float and total float.

Properties

- ActualEndDateDataFieldIndex
- ActualStartDateDataFieldIndex
- AutomaticSchedulingEnabled
- DurationDataFieldIndex
- EarlyEndDateDataFieldIndex
- EarlyStartDateDataFieldIndex
- EndDateForAutomaticScheduling
- EndDateNotLaterThanDataFieldIndex
- FreeFloatDataFieldIndex
- LateEndDateDataFieldIndex
- LateStartDateDataFieldIndex
- LinkDurationDataFieldIndex
- ScheduledProjectEndDate
- ScheduledProjectStartDate
- ScheduleSuccessorsOnlyEnabled
- StartDateForAutomaticScheduling
- StartDateNotEarlierThanDataFieldIndex
- TotalFloatDataFieldIndex

Methods

- ScheduleProject

Properties

ActualEndDateDataFieldIndex

Property of VcScheduler

With this property you can set/retrieve the index of the data field which contains the actual end date of the activity. This is only possible as long as no data has been loaded.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the actual end date

ActualStartDateDataFieldIndex

Property of VcScheduler

With this property you can set/retrieve the index of the data field which contains the actual start date of the activity. This is only possible as long as no data has been loaded.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the currently valid start date

AutomaticSchedulingEnabled

Property of VcScheduler

This property lets you set or retrieve whether automatic time scheduling is switched on or off.

	Data Type	Explanation
Property value	System.Boolean	Automatic time scheduling is switched on (true) or off (false) Default value: false

DurationDataFieldIndex

Property of VcScheduler

With this property you can set/retrieve the index of the data field which contains the duration of the activity. This is only possible as long as no data has been loaded.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the duration of the activity

EarlyEndDateDataFieldIndex

Property of VcScheduler

With this property you can set/retrieve the index of the data field which contains the earliest possible end date of the activity. This is only possible as long as no data has been loaded.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the earliest possible end date of an activity

EarlyStartDateDataFieldIndex

Property of VcScheduler

With this property you can set/retrieve the index of the data field which contains the earliest possible start date of the activity. This is only possible as long as no data has been loaded.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the earliest possible start date of an activity

EndDateForAutomaticScheduling

Property of VcScheduler

In case **Automatic scheduling** is activated, this property lets you set or retrieve the end date of the project.

	Data Type	Explanation
Property value	System.DateTime	Desired end date for automatic scheduling

EndDateNotLaterThanDataFieldIndex

Property of VcScheduler

With this property you can set/retrieve the index of the data field which contains the desired latest end date of the activity. This is only possible as long as no data has been loaded.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the desired late end date

FreeFloatDataFieldIndex

Property of VcScheduler

With this property you can set/retrieve the index of the data field which contains the calculated free float of the activity. This is only possible as long as no data has been loaded.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the free float

LateEndDateDataFieldIndex

Property of VcScheduler

With this property you can set/retrieve the index of the data field which contains the calculated latest possible end date of the project. This is only possible as long as no data has been loaded.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the latest possible end date of an activity

LateStartDateDataFieldIndex

Property of VcScheduler

With this property you can set/retrieve the index of the data field which contains the calculated latest possible start date of the project.activity. This is only possible as long as no data has been loaded.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the latest possible start date of an activity

LinkDurationDataFieldIndex

Property of VcScheduler

This property lets you set or retrieve the index of a data field in the project in which a minimum temporal distance between predecessor and successor can be stored. This is only possible as long as no data has been loaded.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the minimum time space between a predecessor and a successor

ScheduledProjectEndDate

Read Only Property of VcScheduler

This property returns the data **Early end** of a project after having calculated the project dates by **VcScheduler.ScheduleProject** if the end date was set before.

This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.DateTime	Index of the data field which holds the calculated end date of the project

ScheduledProjectStartDate

Read Only Property of VcScheduler

This property returns the **Late start** of a project after having calculated the project dates by **VcScheduler.ScheduleProject** if the start date was set before.

This property can also be set on the **General** property page.

	Data Type	Explanation
Property value	System.DateTime	Index of the data field which holds the calculated start date of the project

ScheduleSuccessorsOnlyEnabled

Property of VcScheduler

With this property you can set/retrieve whether the scheduling of only those nodes that have a predecessor node is switched on or off; otherwise all nodes will be scheduled. A "project start" will thus be ignored.

	Data Type	Explanation
Property value	System.Boolean	Scheduling of nodes only with predecessors is switched on/off

StartDateForAutomaticScheduling

Property of VcScheduler

In case **Automatic scheduling** is activated, this property lets you set or retrieve the start date of the project.

	Data Type	Explanation
Property value	System.DateTime	Desired start date for automatic scheduling

StartDateNotEarlierThanDataFieldIndex

Property of VcScheduler

With this property you can set/retrieve the index of the data field which contains the desired earliest start date of the activity.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the desired early start date

TotalFloatDataFieldIndex

Property of VcScheduler

With this property you can set/retrieve the index of the data field which contains the calculated total float of the activity.

	Data Type	Explanation
Property value	System.Int32	Index of the data field which holds the total float

Methods

ScheduleProject

Method of VcScheduler

This method lets you calculate the dates of a project (early / late start, early / late end, free float, total float) of a project. The desired start and end date can be set by this method. By passing only the end date, the project start will be calculated, by passing only the start date, the project end will be calculated. You can pass both dates, which will add the corresponding float to the activities. (This only works with matching dates, which means that the end date for example should not be within the project time period.) At least one date must be passed, otherwise an error message will occur. If a cycle amongst the nodes and links is identified, the ones affected will be marked.

The results will be stored to fields that you can set by the properties **EarlyStartDateDataFieldIndex**, **LateStartDateDataFieldIndex**, **EarlyEndDateDataFieldIndex**, **LateEndDateDataFieldIndex**, **FreeFloatDataFieldIndex** and **TotalFloatDataFieldIndex**.

	Data Type	Explanation
Parameter:		
⇒ startDate	System.DateTime	Desired start date
⇒ endDate	System.DateTime	Desired end date

Return value	System.Boolean	The project data were successfully calculated (true) / were not calculated (False)
---------------------	----------------	--

Example Code VB.NET

```
' Vorwärtsberechnung (ASAP)
VcScheduler.ScheduleProject(2.5.2017, newDate(0))

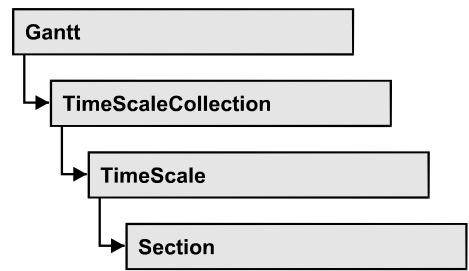
' Rückwärtsberechnung (JIT)
VcScheduler.ScheduleProject(newDate(0), 2.5.2017)
```

Example Code C#

```
// Vorwärtsberechnung (ASAP)
vcScheduler.ScheduleProject(2.5.2017, newDate(0));

// Rückwärtsberechnung (JIT)
vcScheduler.ScheduleProject(newDate(0), 2.5.2017);
```

7.68 VcSection



An object of the type VcSection represents a section of the time scale.

Properties

- CalendarGrid
- DateLineGrid
- LineColor
- NonWorkIntervalsCollapsed
- Ribbon
- StartDate
- TimeUnit
- UnitWidth
- UnitWidthEx

Properties

CalendarGrid

Read Only Property of VcSection

This property lets you retrieve one of the calendar grids used in the section.

The property is an Indexed Property, which in C# is addressed by the method `get_CalendarGrid (gridIndex)`.

	Data Type	Explanation
Parameter: ⇒ gridIndex	System.Int16	Index of the calendar grid
Property value	VcCalendarGrid	CalendarGrid object

DateLineGrid

Read Only Property of VcSection

This property gives you access to the DateLineGrid object, that lets you mark time periods such as days, weeks or months by vertical lines.

The property DateLineGrid is an Indexed Property, which in C# is addressed by the method `get_DateLineGrid (gridIndex)`.

	Data Type	Explanation
Parameter: ⇒ gridIndex	System.Int16	Index of the date line grid
Property value	VcDateLineGrid	DateLine object

Example Code VB.NET

```
Dim timescale As VcTimeScale
Dim section As VcSection
Dim dateLineGrid As VcDateLineGrid

timescale = VcGantt1.TimeScaleCollection.Active
section = timescale.Section(0)
dateLineGrid = section.DateLineGrid(0)
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcSection section = timeScale.get_Section(0);
VcDateLineGrid dateLineGrid = section.get_DateLineGrid(0);
```

LineColor

Read Only Property of VcSection

This property lets you set or retrieve the color of the (border) lines of **all** time scale sections and returns the color of the first time scale section. It is not possible to set a color for each section.

	Data Type	Explanation
Parameter: ⇒ Rückgabewert	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255})
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255})

NonWorkIntervalsCollapsed

Property of VcSection

This property lets you set or retrieve whether workfree periods of this section are to be collapsed. This property can also be set in the subdialog **Edit time scale section** of the **Specify Time Scale** dialog which you can reach by the **Time scales...** button on the property page **Objects**.

Tip Please note that the visible time scale section will be shifted when you modify the property value at runtime. If you want to make sure that always the same reference date is displayed on the left , please call the following method:

```
Set_NonWorkIntervalsCollapsed(vcGantt1, true);
```

```
private static void Set_NonWorkIntervalsCollapsed(VcGantt gantt, bool
collapse)
```

```
{
```

```
    DateTime dt_left = new DateTime();
```

```
    DateTime dt_right = new DateTime();
```

```
    gantt.GetCurrentViewDates(ref dt_left, ref dt_right);
```

```
gantt.TimeScaleCollection.Active.get_Section(0).NonWorkIntervalsCollapse
d = collapse;
```

```
    gantt.ScrollToDate(dt_left,    VcHorizontalAlignment.vcLeftAligned,
0);
```

```
}
```

	Data Type	Explanation
Property value	System.Boolean	Workfree periods are/are not collapsed

Example Code VB.NET

```

Dim timeScaleCltn As VcTimeScaleCollection
Dim timeScale As VcTimeScale
Dim section As VcSection

timeScaleCltn = VcGantt1.TimeScaleCollection
timeScale = timeScaleCltn.Active
section = timeScale.Section(1)
section.NonWorkIntervalsCollapsed = True

```

Example Code C#

```

VcTimeScaleCollection timeScaleCltn = vcGantt1.TimeScaleCollection;
VcTimeScale timeScale = timeScaleCltn.Active;
VcSection section = timeScale.get_Section(1);
section.NonWorkIntervalsCollapsed = true;

```

Ribbon

Property of VcSection

This property lets you access the ribbons of a section.

The property Ribbon is an Indexed Property, which in C# is addressed by the methods `sset_Ribbon (ribbonIndex, pvn)` and `get_Ribbon (ribbonIndex)`.

	Data Type	Explanation
Parameter: ⇒ ribbonIndex	System.Int16	Index of the ribbon
Property value	VcRibbon	Ribbon object

Example Code VB.NET

```

Dim timescale As VcTimeScale
Dim section As VcSection
Dim ribbon As VcRibbon

timescale = VcGantt1.TimeScaleCollection.Active
section = timescale.Section(0)
ribbon = section.Ribbon(0)

```

Example Code C#

```

VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcSection section = timeScale.get_Section(0);
VcRibbon ribbon = section.get_Ribbon(0);

```

StartDate

Property of VcSection

This property lets you set or retrieve the start date of a time scale section. The start date of the first section (Section 0) is automatically set by the project start. It cannot be set here, but can merely be retrieved. Besides, a start date beyond the time scale must not be set.

	Data Type	Explanation
Property value	System.DateTime	Start date of the time scale section

Example Code VB.NET

```
Dim timescale As VcTimeScale
Dim section As VcSection

timescale = VcGantt1.TimeScaleCollection.Active
section = timescale.Section(0)
section.StartDate = "21.06.14"
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcSection section = timeScale.get_Section(0);
section.StartDate = Convert.ToDateTime("21.06.14");
```

TimeUnit

Property of VcSection

This property lets you set or retrieve the time unit that a section is based on.

	Data Type	Explanation
Property value	VcTimeUnit	Time unit of the section
	Possible Values: .vcDay 5 .vcHour 6 .vcMinute 7 .vcSecond 8	Time unit day Time unit hour Time unit minute Time unit second

Example Code VB.NET

```
Dim timescale As VcTimeScale
Dim section As VcSection

timescale = VcGantt1.TimeScaleCollection.Active
section = timescale.Section(0)
section.TimeUnit = VcTimeUnit.vcHour
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcSection section = timeScale.get_Section(0);
section.TimeUnit = VcTimeUnit.vcHour;
```

UnitWidth**Property of VcSection**

This property lets you set or retrieve the unit width of a section (in 1/100 mm). This property also can be set in the **Specify Time Scale** dialog.

	Data Type	Explanation
Property value	System.Int32	unit width (1/100 mm)

Example Code VB.NET

```
Dim timescale As VcTimeScale
Dim section As VcSection

timescale = VcGantt1.TimeScaleCollection.Active
section = timescale.Section(0)
section.UnitWidth = 660
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcSection section = timeScale.get_Section(0);
section.UnitWidth = 660;
```

UnitWidthEx**Property of VcSection**

This property only differs from the property **UnitWidth** by the data type **Double** that is more exact than the data type **Long**.

	Data Type	Explanation
Property value	System.Double	unit width (1/100 mm)

Example Code VB.NET

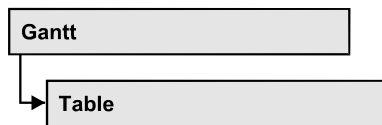
```
Dim timescale As VcTimeScale
Dim section As VcSection

timescale = VcGantt1.TimeScaleCollection.Active
section = timescale.Section(0)
section.UnitWidthEx = 660.0
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcSection section = timeScale.get_Section(0);
section.UnitWidthEx = 660.0;
```


7.69 VcTable



An object of the type VcTable object controls the graphical design of the table section of the diagram: the table heading, column widths and the available formats.

Properties

- ColumnTitle
- ColumnWidth
- Name
- NoOfColumns
- Position
- TableFormatCollection
- UpdateBehaviorName
- Visible

Methods

- IdentifyFormatField
- OptimizeColumnWidth

Properties

ColumnTitle

Property of VcTable

This property lets you specify the caption for each table column. This property also can be set in the **Edit Table** dialog.

The property ColumnTitle is an Indexed Property, which in C# is addressed by the methods set_ColumnTitle (colNumber, pvn) and get_ColumnTitle (colNumber).

Note: The index starts at 1.

	Data Type	Explanation
Parameter: ⇒ colNumber	System.Int16	Number of table column
Property value	System.String	Column title

Example Code VB.NET

```
Dim table As VcTable

table = VcGantt1.LeftTable
table.ColumnTitle(2) = "ID"
```

Example Code C#

```
VcTable table = vcGantt1.LeftTable;
table.set_ColumnTitle(2, "ID");
```

ColumnWidth

Property of VcTable

This property lets you specify the width of each table column. This property also can be set in the **Edit Table** dialog.

The property ColumnWidth is an Indexed Property, which in C# is addressed by the methods set_ColumnWidth (colNumber, pvn) and get_ColumnWidth (colNumber).

	Data Type	Explanation
Parameter: ⇒ colNumber	System.Int16	Number of table column
Property value	System.Int32	Column width in units of 1/100 mm

Example Code VB.NET

```
Dim table As VcTable

table = VcGantt1.LeftTable
table.ColumnWidth(1) = 1500
```

Example Code C#

```
VcTable table = vcGantt1.LeftTable;
table.set_ColumnWidth(1, 1500);
```

Name

Property of VcTable

This property lets you set or retrieve a name for the table. This property also can be set in the **Edit Table** dialog.

	Data Type	Explanation
Property value	System.String	Name of the table

NoOfColumns

Read Only Property of VcTable

This property lets you retrieve the number of columns of the table.

	Data Type	Explanation
Property value	System.Int16	Number of table columns

Position

Read Only Property of VcTable

This property lets you enquire whether the table is displayed left or right of the diagram.

	Data Type	Explanation
Property value	VcTablePosition Possible Values: .vcLeftTable 0 .vcRightTable 1	Position of the table Table on the left of the diagram Table on the right of the diagram

Example Code VB.NET

```
Dim table As VcTable

table = VcGantt1.LeftTable
MsgBox(table.Position)
```

Example Code C#

```
VcTable table = vcGantt1.LeftTable;
MessageBox.Show(table.Position.ToString());
```

TableFormatCollection

Read Only Property of VcTable

This property lets you access the TableFormatCollection object that contains all table formats available.

	Data Type	Explanation
Property value	VcTableFormatCollection	TableFormatCollection object

Example Code VB.NET

```
Dim table As VcTable
Dim formatCltn As VcTableFormatCollection

table = VcGantt1.LeftTable
formatCltn = table.TableFormatCollection
```

Example Code C#

```
VcTable table = vcGantt1.LeftTable;
VcTableFormatCollection formatCltn = table.TableFormatCollection;
```

UpdateBehaviorName

Property of VcTable

This property lets you set or retrieve the name of the UpdateBehavior.

	Data Type	Explanation
Property value	System.String	Name of the UpdateBehavior

Visible

Property of VcTable

This property lets you set or retrieve whether the table is visible or not.

	Data Type	Explanation
Property value	System.Boolean	Table visible/invisible

Example Code VB.NET

```
Dim table As VcTable

table = VcGantt1.LeftTable
table.Visible = True
```

Example Code C#

```
VcTable table = vcGantt1.LeftTable;
table.Visible = true;
```

Methods

IdentifyFormatField

Method of VcTable

This method lets you retrieve the index of the format field at the specified position. If there is a field at the position specified, **True** will be returned, if there isn't, the method will deliver **False**.

	Data Type	Explanation
Parameter:		
⇒ x	System.Int32	X coordinate of the position
⇒ y	System.Int32	Y coordinate of the position
⇐ format	VcTableFormat	Identified format
⇐ formatFieldIndex	System.Int16	Index of the format field
Return value	System.Boolean	A format field exists/does not exist at the position specified

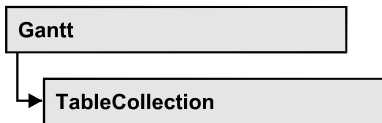
OptimizeColumnWidth

Method of VcTable

This method lets you calculate the optimized width of a column. It depends on the length of the longest text in the column. The setting ColumnNo = 0 optimizes all columns.

	Data Type	Explanation
Parameter:		
⇒ columnNo	System.Int16	Column number
Return value	Void	

7.70 VcTableCollection



An object of the type VcTableCollection contains all available tables. You can access all objects in an iterative loop by **For Each table In TableCollection** or by the methods **First...** and **Next...**. You can access a single table using the methods **TableByName** and **TableByIndex**. The number of tables in the collection object can be retrieved by the property **Count**. By the property **Active** you can set or retrieve the table that is presently active.

Properties

- Active
- Count

Methods

- FirstTable
- GetEnumerator
- NextTable
- TableByIndex
- TableByName

Properties

Active

Property of VcTableCollection

This property lets you set or retrieve the table currently displayed in the diagram.

	Data Type	Explanation
Property value	VcTable	Currently used table

Count

Read Only Property of VcTableCollection

This property lets you retrieve the number of tables in the table collection.

	Data Type	Explanation
Property value	Integer	in

Methods

FirstTable

Method of VcTableCollection

This method can be used to access the initial value, i.e. the first table of a table collection, and to continue in a forward iteration loop by the method **NextTable** for the tables following. If there is no table in the table collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcTable	in

GetEnumerator

Method of VcTableCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the table objects included.

	Data Type	Explanation
Return value	VcObject	in

NextTable

Method of VcTableCollection

This method can be used in a forward iteration loop to retrieve subsequent tables from a table collection after initializing the loop by the method

FirstTable. If there is no table left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcTable	in

TableByIndex

Method of VcTableCollection

This method lets you access a table by its index. If a table does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the table
Return value	VcTable	Table object returned

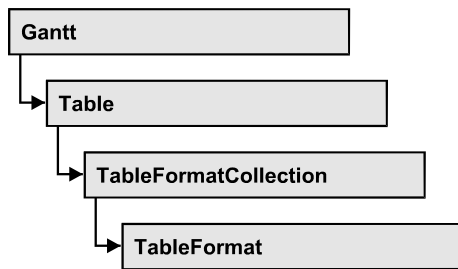
TableByName

Method of VcTableCollection

This method retrieves a table object by its name. If a table of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: tableName	System.String	Name of the table
Return value	VcTable	in

7.71 VcTableFormat



An object of the type VcTableFormat defines the content and the appearance of a table row. A table row contains either the activity data or the group headings. In a table format, you can specify the data field contained in a table field. Each table field is specified by its column. Furthermore, you can specify a font (name, size, body, color), a background color, an horizontal alignment and margins individually for each field.

Available table formats:

- StandardList (for activities that are not summarized)
- ListFormat2 (alternative of StandardList, can be assigned by filters)
- ListFormat3 (alternative of StandardList, can be assigned by filters)
- Subtitle (for group headings when group is expanded)
- Subtitle_n (for multi-level grouping for group headings when group is expanded)
- Collapsed (for group headings when group is collapsed)
- Collapsed_n (for multi-level grouping for group headings when group is collapsed)
- Hierarchy (für summarized activities in a hierarchy)
- HierarchyCollapsed (for collapsed summarized activities in a hierarchy)

Properties

- CollapseColumn
- FieldsSeparatedByLines
- FilterName
- FormatField
- FormatFieldCount
- IndentColumn
- IndentWidth
- Name

- SeparationLineColor
- ThreeDEffect

Methods

- GetEnumerator

Properties

CollapseColumn

Property of VcTableFormat

This property lets you specify whether in a column which contains more than one line + or - for collapsing or showing the lines shall be displayed.

	Data Type	Explanation
Property value	System.Int16	Display of +/- in column switched on

Example Code VB.NET

```
' Display of +/- in the fifth column
VcGantt1.TableCollection.Active.TableFormatCollection.FormatByName("Hierarchy").
CollapseColumn = 5
VcGantt1.TableCollection.Active.TableFormatCollection.FormatByName("HierarchyCol
lapsed").CollapseColumn = 5
```

Example Code C#

```
// Display of +/- in the fifth column
vcGantt1.TableCollection.Active.TableFormatCollection.FormatByName("Hierarchy").
CollapseColumn = 5;
vcGantt1.TableCollection.Active.TableFormatCollection.FormatByName("HierarchyCol
lapsed").CollapseColumn = 5;
```

FieldsSeparatedByLines

Property of VcTableFormat

This property lets you set or retrieve whether the table fields are to be separated by lines.

	Data Type	Explanation
Property value	System.Boolean	Table fields are separated by lines (True)/ are not separated by lines (False).

Example Code VB.NET

```
Dim format As VcTableFormat

format = VcGantt1.LeftTable.TableFormatCollection.FormatByName("StandardList")
format.FieldsSeparatedByLines = True
```

Example Code C#

```
VcTableFormat format =
vcGantt1.LeftTable.TableFormatCollection.FormatByName("StandardList");
format.FieldsSeparatedByLines = true;
```

FilterName

Property of VcTableFormat

This property lets you specify the name of the filter that defines what activities the table format is to apply to.

	Data Type	Explanation
Property value	System.String	Name of the filter

Example Code VB.NET

```
Dim format As VcTableFormat

format = VcGantt1.LeftTable.TableFormatCollection.FormatByName("ListFormat2")
format.FilterName = "Code1NotA"
```

Example Code C#

```
VcTableFormat format =
vcGantt1.LeftTable.TableFormatCollection.FormatByName("ListFormat2")
format.FilterName = "Code1NotA";
```

FormatField

Read Only Property of VcTableFormat

This property gives access to a VcTableFormatField object by its index. The index has to be in the range from 0 to FormatFieldCount-1.

Note to users of versions previous to 3.0: The index does **not** count in the range from 1 to FormatFieldCount as in the versions up to 3.0.

The property FormatField is an Indexed Property, which in C# is addressed by the method get_FormatField (index).

	Data Type	Explanation
Parameter: index	System.Int16	Index of the table format field 0 ... FormatFieldCount-1
Property value	VcTableFormatField	Table format field

FormatFieldCount

Read Only Property of VcTableFormat

This property lets you retrieve the number of table columns of this table format.

	Data Type	Explanation
Property value	System.Int16	Number of table columns

Example Code VB.NET

```
Dim format As VcTableFormat
Dim numberOfColumns As Integer

format = VcGantt1.LeftTable.TableFormatCollection.FormatByName("StandardList")
numberOfColumns = format.FormatFieldCount
```

Example Code C#

```
VcTableFormat format =
vcGantt1.LeftTable.TableFormatCollection.FormatByName("StandardList");
int numberOfColumns = format.FormatFieldCount;
```

IndentColumn

Property of VcTableFormat

This property lets you specify the number of the column which shall be indented.

	Data Type	Explanation
Property value	System.Int16	Number of indented column

Example Code VB.NET

```
' Second column is indented
VcGantt1.TableCollection.Active.TableFormatCollection.FormatByName("StandardList")
.IndentColumn = 2
```

Example Code C#

```
// Second column is indented
vcGantt1.TableCollection.Active.TableFormatCollection.FormatByName("StandardList").IndentColumn = 2;
```

IndentWidth**Property of VcTableFormat**

Specify the measure by which the column shall be indented in mm

	Data Type	Explanation
Property value	System.Int32	Measure of indentation

Example Code VB.NET

```
' Second column is indented by 100 mm
VcGantt1.TableCollection.Active.TableFormatCollection.FormatByName("StandardList").IndentColumn = 2
VcGantt1.TableCollection.Active.TableFormatCollection.FormatByName("StandardList").IndentWidth = 100
```

Example Code C#

```
// Second column is indented by 100 mm
vcGantt1.TableCollection.Active.TableFormatCollection.FormatByName("StandardList").IndentColumn = 2;
vcGantt1.TableCollection.Active.TableFormatCollection.FormatByName("StandardList").IndentWidth = 100;
```

Name**Property of VcTableFormat**

This property lets you set or retrieve the name of the table format.

	Data Type	Explanation
Property value	System.String	Table format name

Example Code VB.NET

```
Dim format As VcTableFormat
Dim formatName As String

format = VcGantt1.LeftTable.TableFormatCollection.FirstFormat
formatName = format.Name
```

Example Code C#

```
VcTableFormat format = vcGantt1.LeftTable.TableFormatCollection.FirstFormat();
string formatName = format.Name;
```

SeparationLineColor

Property of VcTableFormat

This property lets you set or retrieve the color of the separation lines of the table fields. The default color is white.

	Data Type	Explanation
Property value	Color RGB	Color value {0...255},{0...255},{0...255} Default value: RGB(0,0,0)

Example Code VB.NET

```
VcTableFormat.SeparationLineColor = RGB(255, 204, 204)
```

Example Code C#

```
VcTableFormat.SeparationLineColor = RGB(255, 204, 204);
```

ThreeDEffect

Property of VcTableFormat

This property lets you set or retrieve whether this table format will be highlighted by a 3D effect.

	Data Type	Explanation
Property value	System.Boolean	3D effect switched on (True)/switched off (False)

Example Code VB.NET

```
Dim format As VcTableFormat
```

```
format = VcGantt1.LeftTable.TableFormatCollection.FormatByName("StandardList")
format.ThreeDEffect = True
```

Example Code C#

```
VcTableFormat format =
vcGantt1.LeftTable.TableFormatCollection.FormatByName("StandardList");
format.ThreeDEffect = true;
```

Methods

GetEnumerator

Method of VcTableFormat

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the table format fields included.

	Data Type	Explanation
Return value	VcObject	Reference object

Example Code VB.NET

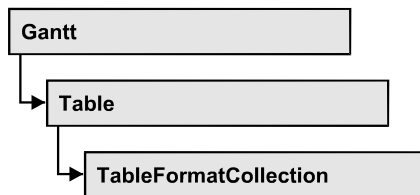
```
Dim format As VcTableFormat
Dim formatField As VcTableFormatField

For Each formatField In format
    Debug.Write(formatField.Index)
Next
```

Example Code C#

```
private void Form1_Load(object sender, System.EventArgs e)
{
    VcTableFormat format;
    foreach (VcTableFormatField formatField in format)
        Console.Writ(formatField.Index);
}
```

7.72 VcTableFormatCollection



An object of the type `VcTableFormatCollection` automatically contains all formats available to the table. You can access all objects in an iterative loop by **For Each format In FormatCollection** or by the methods **First...** and **Next...**. You can access a single format using the methods **FormatByName** and **FormatByIndex**. The number of tables in the collection object can be retrieved by the property **Count**.

Properties

- Count

Methods

- FirstFormat
- FormatByIndex
- FormatByName
- GetEnumerator
- NextFormat

Properties

Count

Read Only Property of VcTableFormatCollection

This property lets you retrieve the number of table formats in the table format collection.

	Data Type	Explanation
Property value	System.Int32	Number of table formats

Example Code VB.NET

```
Dim formatCltn As VcTableFormatCollection
Dim numberOfFormats As Integer

formatCltn = VcGantt1.LeftTable.TableFormatCollection
numberOfFormats = formatCltn.Count
```

Example Code C#

```
VcTableFormatCollection formatCltn = vcGantt1.LeftTable.TableFormatCollection;
int numberOfFormats = formatCltn.Count;
```

Methods

FirstFormat

Method of VcTableFormatCollection

This method can be used to access the initial value, i.e. the first table format of a table format collection and then to continue in a forward iteration loop by the method **NextFormat** for the table formats following. If there is no table format in the table format collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcTableFormat	First table format

Example Code VB.NET

```
Dim format As VcTableFormat

format = VcGantt1.LeftTable.TableFormatCollection.FirstFormat
```

Example Code C#

```
VcTableFormat format = vcGantt1.LeftTable.TableFormatCollection.FirstFormat();
```

FormatByIndex

Method of VcTableFormatCollection

This method lets you access a table format by its index. If a table format does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the table format

Return value	VcTableFormat	Table format object returned
---------------------	---------------	------------------------------

FormatByName

Method of VcTableFormatCollection

By this method you can retrieve a table format by its name. If a table format of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ formatName	System.String	Name of the table format
Return value	VcTableFormat	Table format

Example Code VB.NET

```
Dim format As VcTableFormat

format = VcGantt1.LeftTable.TableFormatCollection.FormatByName("StandardList")
```

Example Code C#

```
VcTableFormat format =
vcGantt1.LeftTable.TableFormatCollection.FormatByName("StandardList");
```

GetEnumerator

Method of VcTableFormatCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the table formats included.

	Data Type	Explanation
Return value	VcObject	Reference object

Example Code VB.NET

```
Dim format As VcTableFormat

For Each format In VcGantt1.LeftTable.TableFormatCollection
    Debug.Write(format.Name)
Next
```

Example Code C#

```
foreach (VcTableFormat format in vcGantt1.LeftTable.TableFormatCollection)
    Console.Write(format.Name);
```

NextFormat

Method of VcTableFormatCollection

This method can be used in a forward iteration loop to retrieve subsequent table formats from a table format collection after initializing the loop by the method **FirstFormat**. If there is no format left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcTableFormat	Subsequent table format

Example Code VB.NET

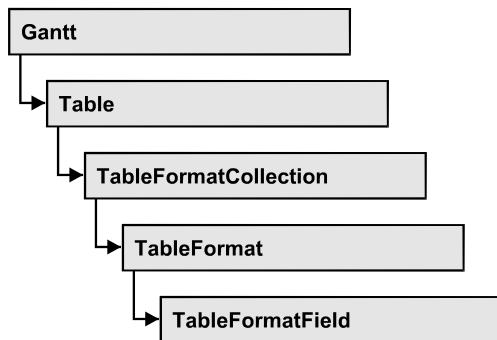
```
Dim formatCltn As VcTableFormatCollection
Dim format As VcTableFormat

formatCltn = VcGantt1.LeftTable.TableFormatCollection
format = formatCltn.FirstFormat
While Not format Is Nothing
    ListBox1.Items.Add(format.Name)
    format = formatCltn.NextFormat
End While
```

Example Code C#

```
VcTableFormatCollection formatCltn = vcGantt1.LeftTable.TableFormatCollection;
VcTableFormat format = formatCltn.FirstFormat();
while (format != null)
{
    listBox1.Items.Add(format.Name);
    format = formatCltn.NextFormat();
}
```

7.73 VcTableFormatField



An object of the type **VcTableFormatField** represents a field of a **VcTableFormat** object. A table format field does not have a name as have many other objects, but is represented by its index that defines its position in the table format. A table can have 100 format fields at maximum.

Properties

- Alignment
- BottomMargin
- ConstantText
- FormatName
- GraphicsFileName
- GraphicsFileNameDataFieldIndex
- GraphicsFileNameMapName
- GraphicsHeight
- Index
- LeftMargin
- MaximumTextLineCount
- MinimumTextLineCount
- MultiState
- PatternBackgroundColorAsARGB
- PatternBackgroundColorDataFieldIndex
- PatternBackgroundColorMapName
- PatternColorAsARGB
- PatternColorDataFieldIndex
- PatternColorMapName
- PatternEx
- PatternExDataFieldIndex
- PatternExMapName
- RightMargin

- TextAndGraphicsCombined
- TextDataFieldIndex
- TextFont
- TextFontColor
- TextFontColorDataFieldIndex
- TextFontColorMapName
- TextFontDataFieldIndex
- TextFontMapName
- TopMargin
- Type

Properties

Alignment

Property of VcTableFormatField

This property lets you set or retrieve the alignment of the content of the table format field.

	Data Type	Explanation
Property value	VcFormatFieldAlignment	Alignment of the field content
	Possible Values: .vcFFABottom 28 .vcFFABottomLeft 27 .vcFFABottomRight 29 .vcFFACenter 25 .vcFFALeft 24 .vcFFARight 26 .vcFFATop 22 .vcFFATopLeft 21 .vcFFATopRight 23	Bottom Bottom left Bottom right Center Left Right Top Top left Top right

BottomMargin

Property of VcTableFormatField

This property lets you set or retrieve the width (in mm) of the bottom margin of the table format field. It can also be set in the **Edit Table Format** dialog box.

	Data Type	Explanation
Property value	System.Int16	Width (in mm) of the bottom margin of the table format field 0...9

ConstantText

Property of VcTableFormatField

This property allows the table format field to display a constant text, if the table format field is of the type *vcFFTText* and if the property **TextDataFieldIndex** was set to **-1**.

	Data Type	Explanation
Property value	System.String	Constant text

FormatName

Read Only Property of VcTableFormatField

This property lets you retrieve the name of the table format to which this field belongs.

	Data Type	Explanation
Property value	System.String	Name of the table format

GraphicsFileName

Property of VcTableFormatField

only for the type vcFFTGraphics: This property lets you set or retrieve the name of a graphics file the content of which is displayed in the table format field. The graphics file name has to be valid. *Available formats:*

- *.BMP (Microsoft Windows Bitmap)
- *.EMF (Enhanced Metafile or Enhanced Metafile Plus)
- *.GIF (Graphics Interchange Format)

- *.JPG (Joint Photographic Experts Group)
- *.PNG (Portable Network Graphics)
- *.TIF (Tagged Image File Format)
- *.VMF (Viewer Metafile)
- *.WMF (Microsoft Windows Metafile, probably with EMF included)

EMF, EMF+, VMF and WMF are vector formats that allow to store a file independent of pixel resolution. All other formats are pixel-oriented and confined to a limited resolution.

The VMF format basically has been deprecated, but it will still be supported for some time to maintain compatibility with existing applications.

	Data Type	Explanation
Property value	System.String	Name of the graphics file

GraphicsFileNameDataFieldIndex

Property of VcTableFormatField

*only for the type **vcFFTGraphics***: This property lets you set or retrieve the data field index that is specified in the property **GraphicsFileNameMapName**. If the property has the value **-1**, in the table format field the graphics that is specified in property **GraphicsFileName** will be displayed. If a valid data field index is specified, but no map is specified, the graphics file name will be read from the specified data field.

	Data Type	Explanation
Property value	System.Int16	Index of the data field

GraphicsFileNameMapName

Property of VcTableFormatField

*only for the type **vcFFTGraphics***: This property lets you set or retrieve the name of a map of the type **vcGraphicsFileMap** or **""**. If a name and additionally a data field index is specified in the property **GraphicsFile-**

NameDataFieldIndex, a graphics of the map will be displayed. If no data field entry applies, the graphics specified in the property **GraphicsFileName** will be displayed.

	Data Type	Explanation
Property value	System.String	Name of the graphics map

GraphicsHeight

Property of VcTableFormatField

This property lets you set or retrieve for the type **vcFFTGraphics** the height of the graphics in the table format field.

	Data Type	Explanation
Property value	System.Int16	Height of the graphics in mm 0 ... 99

Index

Read Only Property of VcTableFormatField

This property lets you retrieve the index of the table format field in the associated table format.

	Data Type	Explanation
Property value	System.Int16	Index of the table format field

LeftMargin

Property of VcTableFormatField

This property lets you set or retrieve the width (in mm) of the left margin of the table format field. It can also be set in the **Edit Table Format** dialog box.

	Data Type	Explanation
Property value	System.Int16	Width (in mm) of the left margin of the table format field 0...9

MaximumTextLineCount

Property of VcTableFormatField

This property lets you set or retrieve the maximum number of lines in the table format field, if the table format field is of the type **vcFFTText**. Also see the property **MinimumTextLineCount**.

	Data Type	Explanation
Property value	System.Int16	Maximum number of lines 0...9

MinimumTextLineCount

Property of VcTableFormatField

This property lets you set or retrieve the minimum number of lines in the table format field, if it is of the type **vcFFTText**. If there is more text than can be taken by the lines, the format field will be enlarged dynamically up to the maximum number of lines. Also see the property **MaximumTextLineCount**. When assigning a value by this property, please also remember to set the **MaximumTextLineCount** value anew, since otherwise the minimum value might overwrite the maximum value.

	Data Type	Explanation
Property value	System.Int16	Minimum number of lines 0...9

MultiState

Property of VcTableFormatField

This property lets you set or retrieve, whether the table format field is a multi-state field. Multi-state fields are used for example to trigger a rotating sequence of different states and of the associated data fields when clicked.

	Data Type	Explanation
Property value	System.Boolean	Multi-state field (True) / no multi-state field (False)

PatternBackgroundColorAsARGB

Property of VcTableFormatField

This property lets you set or retrieve the background color of the table format field. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color.

If the table format field shall have the color of the table format, select the value **-1**.

If in the property **PatternPatternBackgroundColorMapName** a map is specified, the map will set the background color in dependence on the data.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255} Default value: -1

PatternBackgroundColorDataFieldIndex

Property of VcTableFormatField

This property lets you set or retrieve the data field index to be used with a color map specified by the property **PatternBackgroundColorMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

PatternBackgroundColorMapName

Property of VcTableFormatField

This property lets you set or retrieve the name of a color map (type vcColorMap). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **PatternBackgroundColorDataFieldIndex**, then the background color is controlled by the map. If no data field entry applies, the background color that is specified in the property **PatternBackgroundColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the color map

PatternColorAsARGB

Property of VcTableFormatField

This property lets you set or retrieve the pattern color of the table format field. Color values have a transparency or alpha value, followed by a value for a red, a blue and a green partition (ARGB). The values range between 0..255. An alpha value of 0 equals complete transparency, whereas 255 represents a completely solid color. When casting an RGB value on an ARGB value, an alpha value of 255 has to be added.

If the table format field shall have the background color of the table format, select the value **-1**.

	Data Type	Explanation
Property value	System.Drawing.Color	Pattern color of the table format field

PatternColorDataFieldIndex

Property of VcTableFormatField

This property lets you set or retrieve the data field index that has to be specified if the property **PatternColorMapName** is used. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

PatternColorMapName

Property of VcTableFormatField

This property lets you set or retrieve the name of a color map (type vcColorMap). If set to "", no map will be used. Only if a map name and a data field index are specified in the property **PatternColorDataFieldIndex**, the pattern color is controlled by the map. If no data field entry applies, the

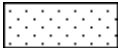






pattern color of the calendar grid that is specified in the property **PatternColor** will be used.



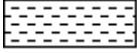



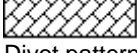
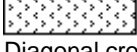
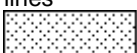
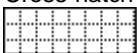

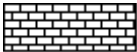

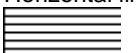



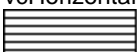
	Data Type	Explanation
Property value	System.String	Name of the color map











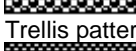
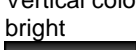
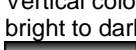
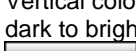





PatternEx

Property of VcTableFormatField

This property lets you set or retrieve the pattern of the field background of the table format field.

	Data Type	Explanation
Property value	VcFillPattern	Pattern type
	Possible Values: .vc05PercentPattern... vc90PercentPattern 01 - 11	Dots in foreground color on background color, the density of the foreground color increasing with the percentage 
	.vcAeroGlassPattern 44	Vertical color gradient in the color of the fill pattern 
	.vcBDiagonalPattern 5	Diagonal lines slanting from bottom left to top right 
	.vcCrossPattern 6	Cross-hatch pattern 
	.vcDarkDownwardDiagonalPattern 2014	Diagonal lines slanting from top left to bottom right; spaced 50% closer than vcFDiagonalPattern and of twice the line width 
	.vcDarkHorizontalPattern 2023	Horizontal lines spaced 50% closer than vcHorizontalPattern and of twice the line width 
	.vcDarkUpwardDiagonalPattern 2015	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern and of twice the line width 

.vcDarkVerticalPattern 2022	Vertical lines spaced 50% closer than vcVerticalPattern and of twice the line width 
.vcDashedDownwardDiagonalPattern 2024	Dashed diagonal lines from top left to bottom right 
.vcDashedHorizontalPattern 2026	Dashed horizontal lines 
.vcDashedUpwardDiagonalPattern 2025	Dashed diagonal lines from bottom left to top right 
.vcDashedVerticalPattern 2027	Dashed vertical lines 
.vcDiagCrossPattern 7	Diagonal cross-hatch pattern, small 
.vcDiagonalBrickPattern 2032	Diagonal brick pattern 
.vcDivotPattern 2036	Divot pattern 
.vcDottedDiamondPattern 2038	Diagonal cross-hatch pattern of dotted lines 
.vcDottedGridPattern 2037	Cross-hatch pattern of dotted lines 
.vcFDiagonalPattern 4	Diagonal lines slanting from top left to bottom right 
.vcHorizontalBrickPattern 2033	Horizontal brick pattern 
.vcHorizontalGradientPattern 52	Horizontal color gradient 
.vcHorizontalPattern 3	Horizontal lines 
.vcLargeCheckerboardPattern 2044	Checkerboard pattern showing squares of twice the size of vcSmallCheckerBoardPattern 
.vcLargeConfettiPattern 2029	Confetti pattern, large 
.vcLightDownwardDiagonalPattern 2012	Diagonal lines slanting to from top left to bottom right; spaced 50% closer than vcBDiagonalPattern 
.vcLightHorizontalPattern 2019	Horizontal lines spaced 50% closer than vcHorizontalPattern 

.vcLightUpwardDiagonalPattern 2013	Diagonal lines slanting from bottom left to top right, spaced 50% closer than vcBDiagonalPattern 
.vcLightVerticalPattern 2018	Vertical lines spaced 50% closer than vcVerticalPattern 
.vcNarrowHorizontalPattern 2021	Horizontal lines spaced 75% closer than vcHorizontalPattern 
.vcNarrowVerticalPattern 2020	Vertical lines spaced 75% closer than vcVerticalPattern 
.vcNoPattern 1276	No fill pattern
.vcOutlinedDiamondPattern 2045	Diagonal cross-hatch pattern, large 
.vcPlaidPattern 2035	Plaid pattern 
.vcShinglePattern 2039	Diagonal shingle pattern 
.vcSmallCheckerBoardPattern 2043	Checkerboard pattern 
.vcSmallConfettiPattern 2028	Confetti pattern 
.vcSmallGridPattern 2042	Cross-hatch pattern spaced 50% closer than vcCrossPattern 
.vcSolidDiamondPattern 2046	Checkerboard pattern showing diagonal squares 
.vcSpherePattern 2041	Checkerboard of spheres 
.vcTrellisPattern 2040	Trellis pattern 
.vcVerticalBottomLightedConvexPattern 43	Vertical color gradient from dark to bright 
.vcVerticalConcavePattern 40	Vertical color gradient from dark to bright to dark 
.vcVerticalConvexPattern 41	Vertical color gradient from bright to dark to bright 
.vcVerticalGradientPattern 62	Vertical color gradient 
.vcVerticalPattern 2	Vertical lines 
.vcVerticalTopLightedConvexPattern 42	Vertical color gradient from bright to dark 

.vcWavePattern 2031	Horizontal waves pattern
.vcWeavePattern 2034	Interwoven stripes pattern
.vcWideDownwardDiagonalPattern 2016	Diagonal lines slanting from top left to bottom right, showing the same spacing but three times the line width of vcF-DiagonalPattern
.vcWideUpwardDiagonalPattern 2017	Diagonal lines slanting from bottom left to top right right, showing the same spacing but three times the line width of vcBDiagonalPattern
.vcZigZagPattern 2030	Horizontal zig-zag lines

PatternExDataFieldIndex

Read Only Property of VcTableFormatField

This property lets you set or retrieve the data field index to be used together with the property **PatternExMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

PatternExMapName

Read Only Property of VcTableFormatField

This property lets you set or retrieve the name of a font map (type vcPatternMap). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **PatternExDataFieldIndex**, then the pattern is controlled by the map. If no data field entry applies, the pattern that is specified in the property **PatternEx** will be used.

	Data Type	Explanation
Property value	System.String	Name of the pattern map

RightMargin

Property of VcTableFormatField

This property lets you set or retrieve the width (in mm) of the right margin of the table format field. It can also be set in the **Edit Table Format** dialog box.

	Data Type	Explanation
Property value	System.Int16	Width (in mm) of the right margin of the table format field 0...9

TextAndGraphicsCombined

Property of VcTableFormatField

This property lets you set or retrieve whether the table field is a combi field. (See also **Edit Table Format** dialog.)

	Data Type	Explanation
Property value	System.Boolean	Combi field (True)/ no combi field (False)

TextDataFieldIndex

Property of VcTableFormatField

This property lets you set or retrieve the index of the data field, the content of which is to be displayed in the table format field. This property only works if the type of the data field is **vcFFTText**. If the value of the index equals **-1**, the content of the property **ConstantText** will be returned instead.

	Data Type	Explanation
Property value	System.Int16	Index of the data field

TextFont

Property of VcTableFormatField

This property lets you set or retrieve the font color of the table format field, if it is of the type **vcFFTText**. If in the property **TextFontMapName** a map was set, the map will control the text font color in dependence of the data.

	Data Type	Explanation
Property value	System.Drawing.Font	Font type of the table format

TextFontColor

Property of VcTableFormatField

This property lets you set or retrieve the font color of the table format field, if it is of the type **vcFFTText**. If a map was set by the property **TextFontMapName**, the map will control the text font color in dependence of the data.

	Data Type	Explanation
Property value	System.Drawing.Color	Font color of the table format

TextFontColorDataFieldIndex

Property of VcTableFormatField

This property lets you set or retrieve the data field index to be used with a font color map specified by the property **TextFontColorMapName**. If you set this property to **-1**, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

TextFontColorMapName

Property of VcTableFormatField

This property lets you set or retrieve the name of a color map (type **vcColorMap**) for the font color, if the format field is of the type **vcFFTText**. If the name of the color map is set to "", no map will be used. If a map name and a data field index are specified by the property **TextFontColorDataFieldIndex**, the font color will be controlled by the map. If no map entry applies, the font color specified in the property **TextFontColor** will be used.

	Data Type	Explanation
Property value	System.String	Name of the font color map

TextFontDataFieldIndex

Property of VcTableFormatField

This property lets you set or retrieve the data field index to be used with a font map specified by the property **TextFontMapName**. If you set this property to -1, no map will be used.

	Data Type	Explanation
Property value	System.Int16	Data field index

TextFontMapName

Property of VcTableFormatField

This property lets you set or retrieve the name of a font map (type vcFontMap). If set to "", no map will be used. If a map name and additionally a data field index is specified in the property **TextFontDataFieldIndex**, then the font is controlled by the map. If no data field entry applies, the font that is specified in the property **TextFont** will be used.

	Data Type	Explanation
Property value	System.String	Name of the font map

TopMargin

Property of VcTableFormatField

This property lets you set or retrieve the width (in mm) of the top margin of the table format field. It can also be set in the **Edit Table Format** dialog box.

	Data Type	Explanation
Property value	System.Int16	Width (in mm) of the top margin of the table format field 0...9

Type

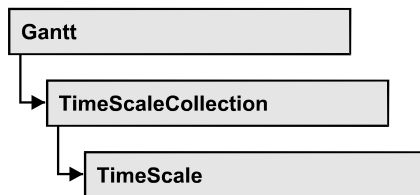
Property of VcTableFormatField

This property lets you set or retrieve the type of the table format field.

1406 API Reference: VcTableFormatField

	Data Type	Explanation
Property value	VcFormatFieldType Possible Values: .vcFFTGraphics 64 .vcFFTText 36	Type of the table format field Graphics Text

7.74 VcTimeScale



The VcTimeScale object represents the time scale at the top of the node area in the diagram. From several time scales that display different units, such as hours or weeks, you can select the time scale that meets your demands. The color and several font attributes can be set as you like. In the settings of the time scale the (vertical) grid lines and possibly the emphasizing of weekends also can be activated.

Properties

- BackgroundColor
- CalendarGridsVisible
- DateGridsVisible
- Font
- FontColor
- Name
- Ribbon
- Section
- ThreeDEffect
- UpdateBehaviorName

Properties

BackgroundColor

Property of VcTimeScale

This property lets you set or retrieve the background color of the time scale.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values {0...255},{0...255},{0...255}

Example Code VB.NET

```
Dim timescale As VcTimeScale

timescale = VcGantt1.TimeScaleCollection.Active
timescale.BackgroundColor = Color.Blue
```

Example Code C#

```
VcTimeScale timescale = vcGantt1.TimeScaleCollection.Active;
timescale.BackgroundColor = Color.LightSteelBlue;
```

CalendarGridsVisible

Property of VcTimeScale

This property lets you set or retrieve whether workfree periods will be marked by gray shadings. This property also can be set in the **Specify Time Scale/ Edit time scale section** dialog.

	Data Type	Explanation
Property value	System.Boolean	Workfree periods are/are not displayed in gray.

Example Code VB.NET

```
Dim timescale As VcTimeScale

timescale = VcGantt1.TimeScaleCollection.Active
timescale.CalendarGridsVisible = True
```

Example Code C#

```
VcTimeScale timescale = vcGantt1.TimeScaleCollection.Active;
timescale.CalendarGridsVisible = true;
```

DateGridsVisible

Property of VcTimeScale

This property lets you set or retrieve whether a (vertical) date grid is displayed. This property also can be set in the **Specify Time Scale** dialog.

	Data Type	Explanation
Property value	System.Boolean	Date grids are/are not displayed.

Example Code VB.NET

```
Dim timescale As VcTimeScale

timescale = VcGantt1.TimeScaleCollection.Active
timescale.DateGridsVisible = True
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
timeScale.DateGridsVisible = true;
```

Font**Property of VcTimeScale**

This property lets you set or retrieve all font attributes of the timescale.

	Data Type	Explanation
Property value	System.Drawing.Font	Font attributes of the timescale

Example Code VB.NET

```
Dim newFont As Font
newFont = VcGantt1.TimeScaleCollection.Active.Font
MsgBox(newFont.ToString())
```

Example Code C#

```
Font newFont = vcGantt1.TimeScaleCollection.Active.Font;
MessageBox.Show(newFont.ToString());
```

FontColor**Property of VcTimeScale**

This property lets you set or retrieve the font color of the time scale.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255})

Example Code VB.NET

```
Dim timescale As VcTimeScale

timescale = VcGantt1.TimeScaleCollection.Active
timescale.FontColor = Color.Blue
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
timeScale.FontColor = Color.LightSteelBlue;
```

Name**Property of VcTimeScale**

This property lets you set or retrieve the name of the time scale.

	Data Type	Explanation
Property value	System.String	Name of the time scale

Example Code VB.NET

```
Dim timescale As VcTimeScale

timescale = VcGantt1.TimeScaleCollection.Active
MsgBox("Active timescale: " + timescale.Name)
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
MessageBox.Show("Active timescale: " + timeScale.Name);
```

Ribbon

Read Only Property of VcTimeScale

This property gives access to the ribbons of a time scale.

The property Ribbon is an Indexed Property, which in C# can be addressed by the method `get_Ribbon (ribbonIndex, sectionIndex)` .

	Data Type	Explanation
Parameter:		
⇒ sectionIndex	System.Int16	Index of the time scale section
⇒ ribbonIndex	System.Int16	Index of the ribbon
Property value	VcRibbon	Ribbon object

Example Code VB.NET

```
Dim timescale As VcTimeScale
Dim ribbon As VcRibbon

timescale = VcGantt1.TimeScaleCollection.Active
ribbon = timescale.Ribbon(0, 0)
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcRibbon ribbon = timeScale.get_Ribbon(0,0);
```

Section

Read Only Property of VcTimeScale

This property gives access to the sections of a timescale.

The property `Section` is an Indexed Property, which in C# is addressed by the method `get_Section` (`sectionIndex`).

	Data Type	Explanation
Parameter: ⇒ <code>sectionIndex</code>	<code>System.Int16</code>	Index of the section
Property value	<code>VcSection</code>	Section object

Example Code VB.NET

```
Dim timescale As VcTimeScale
Dim section As VcSection

timescale = VcGantt1.TimeScaleCollection.Active
section = timescale.Section(0)
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
VcSection section = timeScale.get_Section(0);
```

ThreeDEffect

Property of VcTimeScale

This property lets you set or retrieve whether the time scale should have or has a three-dimensional appearance. This property also can be set in the **Specify Time Scale** dialog.

	Data Type	Explanation
Property value	<code>System.Boolean</code>	3D effect switched on (True)/switched off (False)

Example Code VB.NET

```
Dim timescale As VcTimeScale

timescale = VcGantt1.TimeScaleCollection.Active
timeScale.ThreeDEffect = False
```

Example Code C#

```
VcTimeScale timeScale = vcGantt1.TimeScaleCollection.Active;
timeScale.ThreeDEffect = false;
```

UpdateBehaviorName

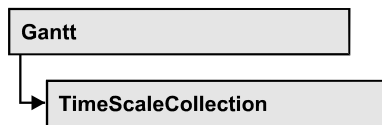
Property of VcTimeScale

This property lets you set or retrieve the name of the UpdateBehavior.

1412 API Reference: VcTimeScale

	Data Type	Explanation
Property value	System.String	Name of the UpdateBehavior

7.75 VcTimeScaleCollection



The VcTimeScaleCollection object contains all available time scales. You can access all objects in an iterative loop by **For Each timeScale In TimeScaleCollection** or by the methods **First...** and **Next...**. You can access a single time scale using the methods **TimeScaleByName** and **TimeScaleByIndex**. The number of time scales in the collection object can be retrieved by the property **Count**. By the property **Active** you can set or retrieve the time scale that is presently active.

Properties

- Active
- Count

Methods

- FirstTimeScale
- GetEnumerator
- NextTimeScale
- TimeScaleByIndex
- TimeScaleByName

Properties

Active

Property of VcTimeScaleCollection

This method lets you set or retrieve the current time scale.

	Data Type	Explanation
Property value	VcTimeScale	Currently displayed timescale

Example Code VB.NET

```
Dim timeScaleCltn As VcTimeScaleCollection
Dim timeScale As VcTimeScale

timeScaleCltn = VcGantt1.TimeScaleCollection
timeScale = timeScaleCltn.Active
```

Example Code C#

```
VcTimeScaleCollection timeScaleCltn = vcGantt1.TimeScaleCollection;
VcTimeScale timeScale = timeScaleCltn.Active;
```

Count

Read Only Property of VcTimeScaleCollection

This property lets you retrieve the number of time scales in the TimeScaleCollection object.

	Data Type	Explanation
Property value	System.Int32	Number of time scales

Example Code VB.NET

```
Dim numberOfTimeScales As Integer

numberOfTimeScales = VcGantt1.TimeScaleCollection.Count
```

Example Code C#

```
int numberOfTimeScales = vcGantt1.TimeScaleCollection.Count;
```

Methods

FirstTimeScale

Method of VcTimeScaleCollection

This method can be used to access the initial value, i.e. the first time scale of a time scale collection, and then to continue in a forward iteration loop by the method **NextTimeScale** for the scales following. If there is no scale in the time scale collection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcTimeScale	First time scale

Example Code VB.NET

```
Dim timeScaleCltn As VcTimeScaleCollection
Dim timeScale As VcTimeScale

timeScaleCltn = VcGantt1.TimeScaleCollection
timeScale = timeScaleCltn.FirstTimeScale
```

Example Code C#

```
VcTimeScaleCollection timeScaleCltn = vcGantt1.TimeScaleCollection;
VcTimeScale timeScale = timeScaleCltn.FirstTimeScale();
```

GetEnumerator**Method of VcTimeScaleCollection**

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the time scale objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

NextTimeScale**Method of VcTimeScaleCollection**

This method can be used in a forward iteration loop to retrieve subsequent time scales from a time scale collection after initializing the loop by the method **FirstTimeScale**. If there is no time scale left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcTimeScale	Succeeding time scale

Example Code VB.NET

```
Dim timeScaleCltn As VcTimeScaleCollection
Dim timeScale As VcTimeScale

timeScaleCltn = VcGantt1.TimeScaleCollection
timeScale = timeScaleCltn.FirstTimeScale
While Not timeScale Is Nothing
    ListBox1.Items.Add(timeScale.Name)
    timeScale = timeScaleCltn.NextTimeScale
End While
```

Example Code C#

```
VcTimeScaleCollection timeScaleCltn = vcGantt1.TimeScaleCollection;
VcTimeScale timeScale = timeScaleCltn.FirstTimeScale();
while (timeScale != null)
{
    listBox1.Items.Add(timeScale.Name);
    timeScale = timeScaleCltn.NextTimeScale();
}
```

TimeScaleByIndex**Method of VcTimeScaleCollection**

This method lets you access a time scale by its index. If a time scale does not exist at the index specified, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ index	System.Int16	Index of the time scale
Return value	VcTimeScale	Time scale object returned

TimeScaleByName**Method of VcTimeScaleCollection**

By this method you can retrieve a time scale by its name. If a time scale of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter: ⇒ timeScaleName	System.String	Name of the time scale
Return value	VcTimeScale	Time scale

Example Code VB.NET

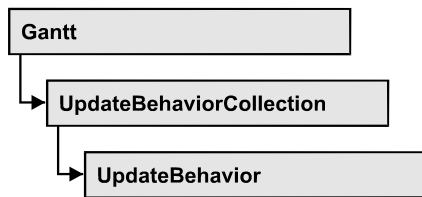
```
Dim timeScaleCltn As VcTimeScaleCollection

timeScaleCltn = VcGantt1.TimeScaleCollection
timeScaleCltn.Active = timeScaleCltn.TimeScaleByName("Days")
```

Example Code C#

```
VcTimeScaleCollection timeScaleCltn = vcGantt1.TimeScaleCollection;
timeScaleCltn.Active = timeScaleCltn.TimeScaleByName("Days");
```

7.76 VcUpdateBehavior



An object of the type **VcUpdateBehavior** contains a set of properties and methods that control the live update behavior of those objects on the screen to which it was assigned.

Properties

- IsEditable
- Name
- Specification

Methods

- PutInOrderAfter

Properties

IsEditable

Property of VcUpdateBehavior

This property lets you set or retrieve whether the update behavior should be editable at run time.

	Data Type	Explanation
Property value	System.Boolean	Update behavior editable (True) / not editable (False) Default value: True

Example Code VB.NET

```

Dim updBeh As VcUpdateBehavior

updBeh = UpdateBehaviorCollection.UpdateBehaviorByName("Immediate")
updBeh.IsEditable = False
  
```

Example Code C#

```
VcUpdateBehavior updBeh =
UpdateBehavior.Collection.UpdateBehaviorByName("Immediate");
updBeh.IsEditable = false;
```

Name**Property of VcUpdateBehavior**

This property lets you set or retrieve the name of an update behavior.

	Data Type	Explanation
Property value	System.String	Name of the update behavior

Example Code VB.NET

```
Dim updBehCltn As VcUpdateBehaviorCollection
Dim updBeh As VcUpdateBehavior

updBehCltn = VcGantt1.UpdateBehaviorCollection
For Each updBeh In updBehCltn
    ComboBox1.Items.Add(updBeh.Name)
Next
```

Example Code C#

```
VcUpdateBehaviorCollection updBehCltn = vcGantt1.UpdateBehaviorCollection;
foreach (VcUpdateBehavior updBeh in updBehCltn)
    comboBox1.Items.Add(updBeh.Name);
```

Specification**Read Only Property of VcUpdateBehavior**

This property lets you retrieve the specification of an update behavior. A specification is a string that contains legible ASCII characters from 32 to 127 only, so it can be stored smoothly to text files or data bases. This allows for persistency. A specification can be used to create an update behavior by the method **VcUpdateBehaviorCollection.AddBySpecification**.

	Data Type	Explanation
Property value	System.String	Specification of the update behavior

Example Code VB.NET

```
Dim updateBehaviorCltn As VcUpdateBehaviorCollection
Dim updateBehavior As VcUpdateBehavior

updateBehaviorCltn = VcGantt1.UpdateBehaviorCollection
updateBehavior = updateBehaviorCltn.FirstUpdateBehavior
MsgBox(updateBehavior.Specification)
```

Example Code C#

```
VcUpdateBehaviorCollection boxCltn = vcGantt1.UpdateBehaviorCollection;
VcUpdateBehavior updateBehavior = updateBehaviorCltn.FirstUpdateBehavior();
MessageBox.Show(updateBehavior.Specification);
```

Methods

PutInOrderAfter

Method of VcUpdateBehavior

This method lets you set the update behavior behind a link appearance specified by name, within the UpdateBehaviorCollection. If you set the name to "", the update behavior will be put in the first position. The order of the update behaviors within the collection determines the order by which they apply to the objects they were assigned to.

	Data Type	Explanation
Parameter: refUpdateBehaviorName	System.String	Name of the update behavior behind which the current update behavior is to be put.

Example Code VB.NET

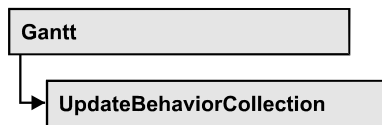
```
Dim updBehCltn As VcUpdateBehaviorCollection
Dim updBeh1 As VcUpdateBehavior
Dim updBeh2 As VcUpdateBehavior

updBehCltn = VcGantt1.UpdateBehaviorCollection()
updBeh1 = updBehCltn.Add("updBeh1")
updBeh2 = updBehCltn.Add("updBeh2")
updBeh1.PutInOrderAfter("updBeh2")
updBehCltn.Update()
```

Example Code C#

```
VcUpdateBehaviorCollection updBehCltn = vcGantt1.UpdateBehaviorCollection;
VcUpdateBehavior updBeh1 = updBehCltn.Add("updBeh1");
VcUpdateBehavior updBeh2 = updBehCltn.Add("updBeh2");
updBeh1.PutInOrderAfter("updBeh2");
updBehCltn.Update();
```


7.77 VcUpdateBehaviorCollection



The VcUpdateBehaviorCollection object contains all update behaviors available. You can access all objects in an iterative loop by **For Each updateBehavior In UpdateBehaviorCollection** or by the methods **First...** and **Next...**. You can access a single update behavior by the methods **UpdateBehaviorByName** and **UpdateBehaviorByIndex**. The number of update behaviors in the collection object can be retrieved by the property **Count**. The methods **Add**, **Copy** and **Remove** allow to handle the update behaviors in the corresponding way.

Properties

- Active
- Count

Methods

- Add
- AddBySpecification
- Copy
- FirstUpdateBehavior
- GetEnumerator
- NextUpdateBehavior
- Remove
- UpdateBehaviorByIndex
- UpdateBehaviorByName

Properties

Active

Read Only Property of VcUpdateBehaviorCollection

This property lets you set or retrieve the update behavior that currently is in effect.

	Data Type	Explanation
Property value	VcUpdateBehavior	Currently used update behavior

Example Code VB.NET

```
Dim updBehCltn As VcUpdateBehaviorCollection
Dim updBeh As VcUpdateBehavior

Set updBehCltn = VcGantt1.UpdateBehaviorCollection
Set updBeh = UpdateBehaviorCltn.Active
```

Example Code C#

```
VcUpdateBehaviorCollection updBehCltn = vcGantt1.UpdateBehaviorCollection;
VcUpdateBehavior updBeh = updBehCltn.UpdateBehaviorByIndex(0);
updBehCltn.Remove(updBeh.Name);
```

Count

Read Only Property of VcUpdateBehaviorCollection

This property lets you retrieve the number of update behaviors in the UpdateBehaviorCollection object.

	Data Type	Explanation
Property value	System.Int32	Number of update behaviors

Example Code VB.NET

```
Dim numberOfUpdateBehavior As Integer

numberOfUpdateBehavior = VcGantt1.UpdateBehaviorCollection.Count
```

Example Code C#

```
int numberOfUpdateBehvior = vcGantt1.UpdateBehaviorCollection.Count;
```

Methods

Add

Method of VcUpdateBehaviorCollection

This method lets you create an update behavior as a member of the UpdateBehaviorCollection. If the name was not used before, the new update behavior object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter: ⇒ updateBehaviorName	System.String	Update behavior name
Return value	VcUpdateBehavior	New update behavior object

Example Code VB.NET

```
newUpdateBehavior = VcGantt1.UpdateBehaviorCollection.Add("updBeh1")
```

Example Code C#

```
newUpdateBehavior = vcGantt1.UpdateBehaviorCollection.Add("updBeh1");
```

AddBySpecification

Method of VcUpdateBehaviorCollection

This method lets you create an update behavior by using a date line grid specification. This way of creating allows update behavior objects to become persistent. The specification of an update behavior can be saved and re-loaded (see VcUpdateBehavior property **Specification**). In a subsequent session the update behavior can be created again from the specification and is identified by its name.

	Data Type	Explanation
Parameter: ⇒ updateBehaviorSpecification	System.String	Update behavior specification
Return value	VcUpdateBehavior	New update behavior object

Example Code VB.NET

```
Dim updBehCltn As VcUpdateBehaviorCollection

updBehCltn = VcGantt1.UpdateBehaviorCollection
updBehCltn.AddBySpecification(textSpecification)
```

Example Code C#

```
VcUpdateBehaviorCollection updBehCltn = vcGantt1.UpdateBehaviorCollection;
updBehCltn.AddBySpecification(textSpecification);
```

Copy

Method of VcUpdateBehaviorCollection

By this method you can copy an update behavior. If the update behavior that is to be copied exists, and if the name of the new update behavior does not

yet exist, the new update behavior object will be returned. Otherwise "Nothing" (in Visual Basic) or "0" (other languages) will be returned.

	Data Type	Explanation
Parameter:		
⇒ updateBehaviorName	System.String	Name of the update behavior to be copied
⇒ newUpdateBehaviorName	System.String	Name of the new update behavior
Return value	VcUpdateBehavior	Update behavior object

Example Code VB.NET

```
Dim updBehCltn As VcUpdateBehaviorCollection

updBehCltn = VcGantt1.UpdateBehaviorCollection
updBehCltn.Copy("UpdateBehaviorOne", "NewUpdateBehavior")
updBehCltn.Update()
```

Example Code C#

```
VcUpdateBehaviorCollection updBehCltn = vcGantt1.UpdateBehaviorCollection;
updBehCltn.Copy("UpdateBehaviorOne", "NewUpdateBehavior");
```

FirstUpdateBehavior

Method of VcUpdateBehaviorCollection

This method can be used to access the initial value, i.e. the first update behavior of an update behavior collection and then to continue in a forward iteration loop by the method **NextUpdateBehavior** for the update behaviors following. If there is no update behavior in the UpdateBehaviorCollection, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcUpdateBehavior	First update behavior

Example Code VB.NET

```
Dim updBehCltn As VcUpdateBehaviorCollection

updBehCltn = VcGantt1.UpdateBehaviorCollection
updBeh = updBehCltn.FirstUpdateBehavior
```

Example Code C#

```
VcUpdateBehaviorCollection updBehCltn = vcGantt1.UpdateBehaviorCollection;
VcUpdateBehavior updBeh = updBehCltn.FirstUpdateBehavior();
```

GetEnumerator

Method of VcUpdateBehaviorCollection

This method returns an Enumerator object which supports the iteration by language specific elements. It is implied in the For...Each construct of Visual Basic and C#. This object allows to iterate over the group objects included.

	Data Type	Explanation
Return value	VcObject	Reference object

Example Code VB.NET

```
Dim updBeh As VcUpdateBehavior

For Each updBeh In VcGantt1.UpdateBehaviorCollection
    Debug.Print updBeh.Name
Next
```

Example Code C#

```
VcUpdateBehavior updBehCltn = vcGantt1.UpdateBehaviorCollection;
foreach (VcUpdateBehavior updBeh in updBehCltn)
    listBox1.Items.Add(updBeh.Name);
```

NextUpdateBehavior

Method of VcUpdateBehaviorCollection

This method can be used in a forward iteration loop to retrieve subsequent update behaviors from an UpdateBehaviorCollection after initializing the loop by the method **FirstUpdateBehavior**. If there is no update behavior left, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Return value	VcUpdateBehavior	Subsequent update behavior

Example Code VB.NET

```
Dim updBehCltn As VcUpdateBehaviorCollection
Dim updBeh As VcUpdateBehavior

updBehCltn = VcGantt1.UpdateBehaviorCollection
updBeh = updBehCltn.FirstUpdateBehavior

While Not updBeh Is Nothing
    ListBox1.Items.Add(updBeh.Name)
    updBeh = updBehCltn.NextUpdateBehavior
End While
```

Example Code C#

```
VcUpdateBehaviorCollection updBehCltn = vcGantt1.UpdateBehaviorCollection;
VcUpdateBehavior updBeh = updBehCltn.FirstUpdateBehavior();

while (updBeh != null)
{
    ListBox.Items.Add(updBeh.Name);
    updBeh = updBehCltn.NextUpdateBehavior();
}
```

Remove**Method of VcUpdateBehaviorCollection**

This method lets you delete an update behavior. If the update behavior is used in another object, it cannot be deleted. Then False will be returned, otherwise True.

	Data Type	Explanation
Parameter:		
⇒ updateBehaviorName	System.String	Update behavior name
Return value	System.Boolean	Update behavior deleted (True)/not deleted (False)

Example Code VB.NET

```
Dim updBehCltn As VcUpdateBehaviorCollection
Dim updBeh As VcUpdateBehavior

updBehCltn = VcGantt1.UpdateBehaviorCollection
updBeh = updBehCltn.UpdateBehaviorByIndex(0)
updBehCltn.Remove(updBeh.Name)
```

Example Code C#

```
VcUpdateBehaviorCollection updBehCltn = vcGantt1.UpdateBehaviorCollection;
VcUpdateBehavior updBeh = updBehCltn.UpdateBehaviorByIndex(0);
updBehCltn.Remove(updBeh.Name);
```

UpdateBehaviorByIndex**Method of VcUpdateBehaviorCollection**

This method lets you access a update behavior by its index. If an update behavior of the specified index does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter:		
⇒ index	System.Int16	Index of the update behavior
Return value	VcUpdateBehavior	Update behavior object returned

Example Code VB.NET

```
Dim updBehCltn As VcUpdateBehaviorCollection
Dim upBeh As VcUpdateBehavior

updBehCltn = VcGantt1.UpdateBehaviorCollection
updBeh = updBehCltn.UpdateBehaviorByIndex(0)
MsgBox(updBeh.Name)
```

Example Code C#

```
VcUpdateBehaviorCollection updBehCltn = vcGantt1.UpdateBehaviorCollection;
VcUpdateBehavior updBeh = updBehCltn.UpdateBehaviorByIndex(0);
MessageBox.Show(updBeh.Name);
```

UpdateBehaviorByName

Method of VcUpdateBehaviorCollection

This method can be used to access an update behavior by its name. If an update behavior of the specified name does not exist, a **none** object will be returned (**Nothing** in Visual Basic).

	Data Type	Explanation
Parameter:		
⇒ updateBehaviorName	System.String	Name of the update behavior
Return value	VcUpdateBehavior	Update behavior

Example Code VB.NET

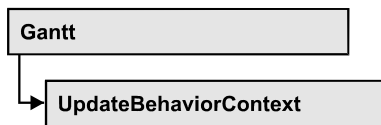
```
Dim updBehCltn As VcUpdateBehaviorCollection
Dim updBeh As VcUpdateBehavior

updBehCltn = VcGantt1.UpdateBehaviorCollection
updBeh = updBehCltn.UpdateBehaviorByName("UpdateBehaviorOne")
MsgBox(updBeh.Name)
```

Example Code C#

```
VcUpdateBehaviorCollection updBehCltn = vcGantt1.UpdateBehaviorCollection;
VcUpdateBehavior updBeh = updBehCltn.UpdateBehaviorByName("UpdateBehaviorOne");
MessageBox.Show(updBeh.Name);
```

7.78 VcUpdateBehaviorContext



An object of the type **VcUpdateBehaviorContext** comprises the context of the update behavior, that is, the behavior of all other objects that are affected by a live update and that can be configured by a user.

Properties

- DelayTime
- IsEditable
- Type
- UpdateMode

Properties

DelayTime

Property of VcUpdateBehaviorContext

This property lets you set the delay time after which the modified objects of the live update visually are to appear while the mouse cursor is moving. Setting this property makes sense only if the property **UpdateMode** was set to **OnPauseWhileMouseMoving**.

	Data Type	Explanation
Property value	System.Int16	Number of milliseconds Default value: 500

Example Code VB.NET

```

Dim updBehCtx As VcUpdateBehaviorContext
Dim delTim As Integer

delTim = VcGantt1.updBehCtx.DelayTime
  
```

Example Code C#

```

int numOfMS = VcUpdateBehaviorContext.DelayTime;
  
```


IsEditable

Property of VcUpdateBehaviorContext

This property lets you set or retrieve whether the context of the update behavior should be editable at run time.

	Data Type	Explanation
Property value	System.Boolean	Context of the update behavior editable (True) / not editable (False) Default value: True

Example Code VB.NET

```
Dim updBehCtx As VcUpdateBehaviorContext
updBehCtx.Editable = False
```

Example Code C#

```
VcUpdateBehaviorContext updBehCtx.Editable = false;
```

Type

Read Only Property of VcUpdateBehaviorContext

This property lets you retrieve defined areas (context types) that are affected by the live update and to which the properties **Editable**, **UpdateMode** und **DelayTime** can be applied.

	Data Type	Explanation
Property value	VcUpdateBehaviorContextType Possible Values: .vcBoxesChangeAnchorNode 1403 .vcBoxesChangePosition 1402 .vcBoxesChangeSize 1401 .vcCurvesChangeValue 1302 .vcCurvesChangeXAndYValue 1303 .vcCurvesChangeXValue 1301 .vcDateLinesChangeDate 801 .vcGroupLevelLayoutsAutoCollapseGroups 705 .vcGroupLevelLayoutsAutoExpandTargetGroup 707 .vcGroupLevelLayoutsChangeGroupsSortingOrder 701 .vcGroupLevelLayoutsNodesOptimization 702 .vcGroupLevelLayoutsOverlappingNodesSorting 704	Available update areas (types): Boxes change anchor node Boxes change position Boxes change size Curves change Y-value Curves change X- and Y-value Curves change X-value Date lines change date In the group level layout the groups are collapsed automatically In the group level layout target groups are expanded automatically Group level layouts change the sorting order of groups Group level layouts optimize nodes Group level layouts sort overlapping nodes

.vcGroupLevelLayoutsRestoreAutoCollapsedGroups 706	In the group level layout the automatically collapsed groups are restored automatically.
.vcGroupLevelLayoutsRestoreAutoExpandedGroups 708	In the group level layout automatically expanded target groups are restored
.vcGroupLevelLayoutsSummaryBarsCalculation 703	Group level layouts calculate summary bars
.vcHierarchyLevelLayoutAutoCollapseGroups 302	In the hierarchy level layout the groups are collapsed automatically
.vcHierarchyLevelLayoutAutoExpandTargetGroups 304	In the hierarchy level layout target groups are expanded automatically
.vcHierarchyLevelLayoutRestoreAutoCollapsedGroups 303	In the hierarchy level layout the automatically collapsed groups are restored automatically.
.vcHierarchyLevelLayoutRestoreAutoExpandedGroups 305	In the hierarchy level layout automatically expanded target groups are restored.
.vcHierarchyLevelLayoutSummaryBarsCalculation 301	The hierarchy level layout calculates summary bars
.vcHistogramsLayerSourceCurvesCalculations 1101	Curve calculation from layer data in histograms
.vcLinksChangeSuccessorNode 402	Links change their successor node
.vcNodeLevelLayoutsChangeNodesSortingOrder 201	Node level layouts change the sorting order of nodes
.vcNodesAutoScheduling 105	Nodes are automatically scheduled
.vcNodesChangeDatesDuration 101	Nodes change their dates or their duration
.vcNodesFiltering 102	Nodes are filtered
.vcNodesGrouping 104	Nodes are grouped
.vcNumericScalesChangeUnitWidth 1201	Numeric scales change unit width
.vcSashesChangePosition 1501	Sashes change position
.vcTablesChangeColumnWidth 901	Tables change column width
.vcTimeScalesChangeSectionStartDate 1002	Time scales change the start date of a section
.vcTimeScalesChangeSectionStartDate 1002	Time scales change the start date of a section
.vcTimeScalesChangeUnitWidth 1001	Time scales change unit width

UpdateMode

Property of VcUpdateBehaviorContext

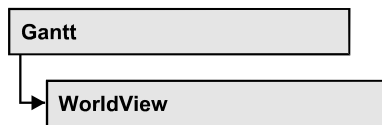
In a self-created update behavior this property lets you set or retrieve a cursor action on which the live update is to take place. If this property was set to **OnPauseWhileMouseMoving**, you can set the desired delay time

by the **DelayTime** property.

1430 API Reference: VcUpdateBehaviorContext

	Data Type	Explanation
Property value	VcUpdateMode	Available actions of the cursor: Default value: vcOnMouseMove
	Possible Values: .vcOnMouseMove 1 .vcOnMouseUp 0 .vcOnPauseWhileMouseMove 2	The update is displayed when the mouse cursor moves The update is displayed when the left mouse button is released The update is displayed when a pause occurs during the movements of the mouse cursor

7.79 VcWorldView



An object of the type **VcWorldView** designates the world view window.

Properties

- Border
- BorderColor
- Height
- HeightActualValue
- Left
- LeftActualValue
- MarkingColor
- Mode
- ScrollBarMode
- Top
- TopActualValue
- UpdateBehaviorName
- Visible
- Width
- WidthActualValue

Properties

Border

Property of VcWorldView

This property lets you set or retrieve whether the world view has a frame (not valid for the **vcPopupWindow** mode). The color of the frame is **Color.Black**. This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Boolean	World view with a border line (True)/without border line (False) Default value: True

Example Code VB.NET

```
VcGantt1.WorldView.Mode = VcWorldViewMode.vcNotFixed
VcGantt1.WorldView.Border = True
```

Example Code C#

```
vcGantt1.WorldView.Mode = VcWorldViewMode.vcNotFixed;
vcGantt1.WorldView.Border = true;
```

BorderColor

Property of VcWorldView

This property lets you set/retrieve the color of the frame that may be visible.

	Data Type	Explanation
Property value	System.Drawing.Color RGB ({0...255},{0...255},{0...255})	RGB color values ({0...255},{0...255},{0...255}) Default value: 0,0,0

Height

Property of VcWorldView

This property lets you retrieve the vertical extension of the world view. It can also be set in the modes **vcFixedAtTop** and **vcFixedAtBottom**.

The coordinates are to be specified as pixels, referring to the screen.

This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Int32	Height of the world view Default value: 100

Example Code VB.NET

```
VcGantt1.WorldView.Height = 100
```

Example Code C#

```
vcGantt1.WorldView.Height = 100;
```

HeightActualValue**Read Only Property of VcWorldView**

This property lets you retrieve the vertical extension of the world view which actually is displayed. In the modes **b!vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** the actual value may differ from the one that was set because in these modes either the height or the width is preset.

	Data Type	Explanation
Property value	System.Int32	Actual height of the world view {0, ...} Default value: 100

Example Code VB.NET

```
VcGantt1.LegendView.Height = 300
```

Example Code C#

```
vcGantt1.LegendView.Height = 100;
```

Left**Property of VcWorldView**

This property lets you retrieve the left position of the world view. It can also be set in the modes **vcNotFixed** and **vcPopupWindow**.

The coordinates are to be specified as pixels, referring to the screen.

This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Int32	Left position of the world view Default value: 0

Example Code VB.NET

```
VcGantt1.WorldView.Left = 200
```

Example Code C#

```
vcGantt1.WorldView.Left = 200;
```

LeftActualValue

Read Only Property of VcWorldView

This property lets you retrieve the left position of the world view which actually is displayed. In the modes **b!vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** the actual value may differ from the one that was set because in these modes either the height or the width is preset.

	Data Type	Explanation
Property value	System.Int32	Actual left position of the world view {0, ...} Default value: 0

Example Code VB.NET

```
VcGantt1.LegendView.LeftActualValue = 150
```

Example Code C#

```
vcGantt1.LegendView.LeftActualValue = 150;
```

MarkingColor

Property of VcWorldView

This property lets you set or retrieve the line color of the rectangle that indicates the selected section in the World View. This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Drawing.Color	RGB color values ({0...255},{0...255},{0...255}) Default value: RGB(0, 0, 255)

Example Code VB.NET

```
VcGantt1.WorldView.MarkingColor = Color.Red
```

Example Code C#

```
vcGantt1.WorldView.MarkingColor = Color.Red;
```

Mode

Property of VcWorldView

This property lets you set or retrieve the world view mode. This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	VcWorldViewMode	Mode of the world view Default value: vcPopupWindow
	Possible Values:	
	.vcFixedAtBottom 4	The world view is displayed on the bottom of the control window. The reference system of the coordinates is the control. With this value set, the height can be specified, whereas the position and the width are fixed.
	.vcFixedAtLeft 1	The world view is displayed on the left side of the VARCHART .NET control window. Then the width can be specified, whereas the position and the height are fixed.
	.vcFixedAtRight 2	The world view is displayed on the right side of the control window. The reference system of the coordinates is the control. With this value set, the width can be specified, whereas the position and the height are fixed.
	.vcFixedAtTop 3	The world view is displayed on the top of the control window. The reference system of the coordinates is the control. With this value set, the height can be specified, whereas the position and the width are fixed.
	.vcPopupWindow 6	The world view is a popup window with its own frame. The reference system of the coordinates is the screen. The user can modify its position and extension, open it via the default context menu, and close it via the Close button in the frame.

Example Code VB.NET

```
VcGantt1.WorldView.Mode = VcWorldViewMode.vcFixedAtBottom
```

Example Code C#

```
vcGantt1.WorldView.Mode = VcWorldViewMode.vcFixedAtBottom;
```

ScrollBarMode

Property of VcWorldView

This property lets you set or retrieve the scroll bar mode of the world view. This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	VcWorldViewScrollBarMode	Scrollbarmode Default value: NoScrollBar
	Possible Values:	
	.vcAutomaticScrollBar 3	Display of a horizontal or vertical scrollbar if required.
	.vcHorizontalScrollBar 1	Display of a horizontal scrollbar if required.
	.vcNoScrollBar 0	The chart is always displayed completely without scrollbars.
	.vcVerticalScrollBar 2	Display of a vertical scrollbar if required.

Example Code VB.NET

```
VcGantt1.WorldView.ScrollBarMode = vcAutomaticScrollbar
```

Example Code C#

```
vcGantt1.WorldView.ScrollBarMode = vcAutomaticScrollBar;
```

Top

Property of VcWorldView

This property lets you retrieve the top position of the world view. It can also be set in the modes **vcNotFixed** and **vcPopupWindow**.

The coordinates are to be specified as pixels, referring to the screen.

This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Int32	Top position of the world view Default value: 0

Example Code VB.NET

```
VcGantt1.WorldView.Top = 20
```

Example Code C#

```
vcGantt1.WorldView.Top = 20;
```

TopActualValue

Read Only Property of VcWorldView

This property lets you enquire the top position of the world view which actually is displayed. In the modes **b!vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** the actual value

may differ from the one that was set because in these modes either the height or the width is preset.

	Data Type	Explanation
Property value	System.Int32	Actual top position of the world view {0, ...}

Example Code VB.NET

```
VcGantt1.LegendView.TopActualValue = 40
```

Example Code C#

```
vcGantt1.LegendView.TopActualValue = 40;
```

UpdateBehaviorName

Read Only Property of VcWorldView

This property lets you set or retrieve the name of the UpdateBehavior.

	Data Type	Explanation
Property value	System.String	Name of the UpdateBehavior

Visible

Property of VcWorldView

This property lets you enquire/set whether the world view is visible or not. This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Boolean	World view visible (True)/not visible (False) Default value: False

Example Code VB.NET

```
VcGantt1.WorldView.Visible = True
```

Example Code C#

```
vcGantt1.WorldView.Visible = true;
```

Width

Property of VcWorldView

This property lets you retrieve the horizontal extent of the world view. It can also be set in the modes **vcFixedAtLeft**, **vcFixedAtRight**, **vcNotFixed** and **vcPopupWindow**.

The coordinates are to be specified as pixels, referring to the screen.

This property also can be set on the **Additional Views** property page.

	Data Type	Explanation
Property value	System.Int32	Horizontal extension of the world view Default value: 100

Example Code VB.NET

```
VcGantt1.WorldView.Width = 200
```

Example Code C#

```
vcGantt1.WorldView.Width = 200;
```

WidthActualValue

Read Only Property of VcWorldView

This property lets you retrieve the horizontal extent of the legend view which actually is displayed. In the modes **b!vcLVFixedAtBottom**, **vcLVFixedAtLeft**, **vcLVFixedAtRight**, **vcLVFixedAtTop** the actual value may differ from the one that was set because in these modes either the height or width is preset.

	Data Type	Explanation
Property value	System.Int32	Actual horizontal extension of the world view {0, ...} Default value: 100

Example Code VB.NET

```
VcGantt1.LegendView.WidthActualValue = 600
```

Example Code C#

```
vcGantt1.LegendView.WidthActualValue = 600;
```

8 Index

A

AbsoluteBottomMarginInInches

Property of
VcPrinter 1250

AbsoluteLeftMarginInCM

Property of
VcPrinter 1251

AbsoluteLeftMarginInInches

Property of
VcPrinter 1251

AbsoluteRightMarginInCM

Property of
VcPrinter 1252

AbsoluteRightMarginInInches

Property of
VcPrinter 1252

AbsoluteTopMarginInCM

Property of
VcPrinter 1253

AbsoluteTopMarginInInches

Property of
VcPrinter 1253

Active

Property of
VcCalendarCollection 514
VcHistogramCollection 1032
VcNumericScaleCollection 1244
VcTableCollection 1377
VcTimeScaleCollection 1413
VcUpdateBehaviorCollection 1420

ActiveNodeFilter

Property of
VcGantt 710

Activities

hierarchical arrangement 711

Activity 185

create 41
delete 42
edit data 42
modify duration 41
move 41
saving and reloading order 440

ActualEndDateDataFieldIndex

Property of
VcScheduler 1359

ActualStartDateDataFieldIndex

Property of
VcScheduler 1359

Add

Method of
VcBoxCollection 479
VcBoxFormatCollection 491
VcCalendarCollection 516
VcCalendarGridCollection 539
VcCalendarProfileCollection 549
VcCurveCollection 589
VcDataRecordCollection 614
VcDataTableCollection 626
VcDataTableFieldCollection 639
VcDateLineCollection 657
VcDateLineGridCollection 677
VcFilterCollection 691
VcGroupLevelLayoutCollection 1010
VcIntervalCollection 1061
VcLayerCollection 1109
VcLineFormatCollection 1137

VcLinkAppearanceCollection 1170
 VcMapCollection 1188
 VcUpdateBehaviorCollection 1421

AddBySpecification

Method of
 VcBoxCollection 479
 VcBoxFormatCollection 491
 VcCalendarCollection 516
 VcCalendarGridCollection 539
 VcCalendarProfileCollection 549
 VcCurveCollection 590
 VcDateLineCollection 657
 VcDateLineGridCollection 677
 VcFilterCollection 692
 VcGroupLevelLayoutCollection 1010
 VcIntervalCollection 1061
 VcLayerCollection 1109
 VcLineFormatCollection 1137
 VcLinkAppearanceCollection 1171
 VcMapCollection 1188
 VcUpdateBehaviorCollection 1422

AddDuration

Method of
 VcCalendar 508

Addend

Property of
 VcCurve 555

Additional text 411

AddSubCondition

Method of
 VcFilter 686

AdjustToReferenceDate

Property of
 VcDateLineGrid 665

Administrate calendar profiles

dialog 338

Alignment

Property of
 VcBoundingBox 454
 VcBoxFormatField 497
 VcLayerFormatField 1118
 VcLineFormatField 1143
 VcPrinter 1254
 VcTableFormatField 1392

AllBorderBoxesShownOnCombinedControls

Property of
 VcPrinter 1254

AllData

Property of
 VcDataRecord 606
 VcLink 1154
 VcNode 1205

AllLayersMovingTogether

Property of
 VcGantt 710

AllLayersMovingTogetherAlways

Property of
 VcGantt 711

AlwaysCurrentDate

Property of
 VcDateLine 645

AnchoringInteractionsAllowed

Property of
 VcBox 463

AnchoringLineVisible

Property of
 VcBox 463

AnchorToNode

Method of
 VcBox 473

AnnotationAtBottom

Property of

- VcDateLineGrid 665
- AnnotationAtCenter**
 - Property of
 - VcDateLineGrid 665
- AnnotationAtTop**
 - Property of
 - VcDateLineGrid 666
- Arrangement**
 - Property of
 - VcGantt 711
- ArrowKeyMode**
 - Property of
 - VcGantt 712
- ArrowKeyStepSizeMultiplier**
 - Property of
 - VcGantt 713
- AssignmentDataTableName**
 - Property of
 - VcResourceScheduler2 1281
- AssignmentIsResultFieldIndex**
 - Property of
 - VcResourceScheduler2 1283
- AssignmentIsVisibleFieldIndex**
 - Property of
 - VcResourceScheduler2 1283
- AssignmentLoadOrConsumptionPerItemFieldIndex**
 - Property of
 - VcResourceScheduler2 1284
- AssignmentMaximumLoadFieldIndex**
 - Property of
 - VcResourceScheduler2 1285
- AssignmentMinimumLoadFieldIndex**
 - Property of
 - VcResourceScheduler2 1285
- AssignmentMinimumMaximumLoadType**
 - Property of
 - VcResourceScheduler2 1286
- AssignmentOperationIDFieldIndex**
 - Property of
 - VcResourceScheduler2 1287
- AssignmentResourceIDFieldIndex**
 - Property of
 - VcResourceScheduler2 1287
- AssignmentResourceSelectionStrategyFieldIndex**
 - Property of
 - VcResourceScheduler2 1288
- AutoCollapseGroups**
 - Property of
 - VcGroupLevelLayout 987
 - VcHierarchyLevelLayout 1014
- AutoExpandTargetGroup**
 - Property of
 - VcGroupLevelLayout 988
 - VcHierarchyLevelLayout 1015
- AutomaticSchedulingEnabled**
 - Property of
 - VcScheduler 1359
- Autoschedule 256**

B

Background color

selected row 250

BackgroundColor

Property of

- VcCalendarGrid 522
- VcInterval 1046
- VcLayer 1067
- VcTimeScale 1407

BackgroundColorDataFieldIndex

Property of

- VcCalendarGrid 523

- VcLayer 1067
- BackgroundColorMapName**
 - Property of
 - VcCalendarGrid 523
 - VcLayer 1069
- Bars**
 - moving into visible area 438
- BaseCalendarUsageForSupplementTimes**
 - Property of
 - VcResourceScheduler2 1289
- BaseTimeUnit**
 - Property of
 - VcResourceScheduler2 1290
- BaseTimeUnitsPerStep**
 - Property of
 - VcResourceScheduler2 1290
- BodiesCollapsed**
 - Property of
 - VcGroupLevelLayout 988
 - VcHierarchyLevelLayout 1015
- BodiesCollapsedDataFieldIndex**
 - Property of
 - VcGroupLevelLayout 988
 - VcHierarchyLevelLayout 1015
- BodiesCollapsedMapName**
 - Property of
 - VcGroupLevelLayout 989
 - VcHierarchyLevelLayout 1016
- BodyCollapsed**
 - Property of
 - VcGroup 972
- Border**
 - Property of
 - VcLegendView 1125
 - VcWorldView 1431
- BorderArea**
 - Property of
 - VcGantt 714
 - see also
 - VcBorderArea 452
- BorderBox**
 - alignment 454
 - Method of
 - VcBorderArea 452
 - see also
 - VcBorderBox 454
- BorderColor**
 - Property of
 - VcLegendView 1126
 - VcWorldView 1432
- Bottom**
 - Property of
 - VcRect 1274
- BottomMargin**
 - Property of
 - VcLayerFormatField 1118
 - VcTableFormatField 1392
- Box 71**
 - allow multiple marking 230
 - allow new ones 229
 - anchor to node 474
 - by index 480
 - context menu is/is not enabled 717
 - ID of the node the box is tied to 469
 - marking 468
 - see also
 - VcBox 462
 - show anchoring line 463
 - tie to node 463
- Box format**
 - by index 493
- Box format field**
 - alignment 497

background color 502
font color 503
font type 503
height of graphics 499
index 499
maximum number of lines 500
minimum number of lines 500
minimum width 501
format name 498
pattern color 502
type 504

BoxByIndex

Method of
VcBoxCollection 480

BoxByName

Method of
VcBoxCollection 481

BoxCollection

Property of
VcGantt 715
see also
VcBoxCollection 478

BoxCreationAllowed

Property of
VcGantt 715

Boxes

actual extent 474
convert pixel to offset 476

BoxFormat

see also
VcBoxFormat 485

BoxFormatCollection

Property of
VcGantt 715
see also
VcBoxFormatCollection 490

BoxFormatField

see also
VcBoxFormatField 497

Browser 215**C****CalcDuration**

Method of
VcCalendar 509

CalculateCurrentWidth

Method of
VcLayer 1106

CalculateLineCount

Method of
VcLayerFormatField 1124

Calendar 131

by index 517
name 545
number of seconds of a workday 507
see also
VcCalendar 505

Calendar grid 354

background color 523
background color 523
line color 525, 526
line color map 526
line thickness 527
line type 527
name 528
specification 534

Calendar Grid

Snap target at date 534
Start date as snap target 524, 535

Calendar GridLayer as snap target 1071**Calendar Grids**

administrate 298

calendar profile

- number 549
- Calendar profile**
 - by index 550
 - order 547
 - retrieving a calendar profile by its name 550
 - type 546
- CalendarByIndex**
 - Method of
 - VcCalendarCollection 517
- CalendarByName**
 - Method of
 - VcCalendarCollection 517
- CalendarCollection**
 - Property of
 - VcGantt 716
 - see also
 - VcCalendarCollection 514
- CalendarGrid**
 - data field index of the calendar map 524
 - data field index of the visibility map 536, 674
 - name of the calendar map 524
 - name of the visibility map 536, 675
 - Property of
 - VcSection 1366
 - see also
 - VcCalendarGrid 521
- CalendarGridByIndex**
 - Method of
 - VcCalendarGridCollection 540
- CalendarGridByName**
 - Method of
 - VcCalendarGridCollection 541
- CalendarGridCollection**
 - Property of
 - VcGantt 716
- see also
 - VcCalendarGridCollection 538
- CalendarGridName**
 - Property of
 - VcGroupLevelLayout 989
 - VcNodeLevelLayout 1221
- CalendarGridsVisible**
 - Property of
 - VcGroupLevelLayout 989
 - VcHistogram 1022
 - VcNodeLevelLayout 1221
 - VcTimeScale 1408
- CalendarGridsWithChildGroups**
 - Property of
 - VcGroupLevelLayout 989
- CalendarName**
 - Property of
 - VcCalendarGrid 523
 - VcHistogram 1022
 - VcRibbon 1345
- CalendarNameDataFieldIndex**
 - Property of
 - VcCalendarGrid 524
 - VcGroupLevelLayout 990
- CalendarNameMapName**
 - Property of
 - VcCalendarGrid 524
- CalendarProfile**
 - see also
 - VcCalendarProfile 545
- CalendarProfileByIndex**
 - Method of
 - VcCalendarProfileCollection 550
- CalendarProfileByName**
 - Method of
 - VcCalendarProfileCollection 550
- CalendarProfileCollection**

- Property of
 - VcCalendar 506
 - VcGantt 717
- see also
 - VcCalendarProfileCollection 548
- CalendarProfileName**
 - Property of
 - VcInterval 1046
- Clear**
 - Method of
 - VcCalendar 509
 - VcCurve 578
- CollapseColumn**
 - Property of
 - VcTableFormat 1381
- Color**
 - Property of
 - VcMapEntry 1194
- ColumnTitle**
 - Property of
 - VcTable 1372
- ColumnWidth**
 - Property of
 - VcTable 1373
- CombiningControlsEnabled**
 - Property of
 - VcPrinter 1255
- ComparisonValueAsString**
 - Property of
 - VcFilterSubCondition 696
- CompletionDataFieldIndex**
 - Property of
 - VcLayer 1070
- Configuration 70, 226**
 - save 791
- ConnectionOperator**
 - Property of
 - VcFilterSubCondition 697
- ConsiderFilterEntries**
 - Property of
 - VcMap 1180
- ConsiderLinkRelationTypesOnNodeDragging**
 - Property of
 - VcGantt 717
- ConstantText**
 - Property of
 - VcLayerFormatField 1118
 - VcLineFormatField 1143
 - VcTableFormatField 1393
- Context menu**
 - disable 442
 - for groups 423
 - for links 422
 - for nodes 420
 - for the diagram 415
 - for the histogram 426
 - for the timescale 425
 - of boxes 429
- Context menu for boxes 230**
- Context of the update behavior**
 - delay time 1427
 - editable 1428
 - type 1428
 - update mode 1429
- ContextMenuForBoxesEnabled**
 - Property of
 - VcGantt 717
- ConvertDistance**
 - Method of
 - VcGantt 788
- Copy**
 - Method of
 - VcBoxCollection 481

- VcBoxFormatCollection 492
- VcCalendarCollection 517
- VcCalendarGridCollection 541
- VcCalendarProfileCollection 550
- VcCurveCollection 590
- VcDataTableCollection 626
- VcDataTableFieldCollection 639
- VcDateLineCollection 658
- VcDateLineGridCollection 678
- VcFilterCollection 692
- VcGroupLevelLayoutCollection 1011
- VcIntervalCollection 1062
- VcLayerCollection 1110
- VcLineFormatCollection 1138
- VcLinkAppearanceCollection 1171
- VcMapCollection 1189
- VcUpdateBehaviorCollection 1422
- CopyFormatField**
 - Method of
 - VcBoxFormat 488
 - VcLayerFormat 1115
 - VcLineFormat 1135
- CopySubCondition**
 - Method of
 - VcFilter 687
- Count**
 - Property of
 - VcBoxCollection 478
 - VcBoxFormatCollection 490
 - VcCalendarCollection 515
 - VcCalendarGridCollection 538
 - VcCalendarProfileCollection 549
 - VcCurveCollection 588
 - VcDataDefinitionTable 600
 - VcDataRecordCollection 613
 - VcDataTableCollection 625
 - VcDataTableFieldCollection 638
 - VcDateLineCollection 656
 - VcDateLineGridCollection 676
 - VcFilterCollection 690
 - VcGroupCollection 982
 - VcGroupLevelLayoutCollection 1009
 - VcHistogramCollection 1033
 - VcIntervalCollection 1061
 - VcLayerCollection 1108
 - VcLineFormatCollection 1136
 - VcLinkAppearanceCollection 1169
 - VcLinkCollection 1176
 - VcMap 1181
 - VcMapCollection 1187
 - VcNodeCollection 1216
 - VcNumericScaleCollection 1245
 - VcTableCollection 1378
 - VcTableFormatCollection 1387
 - VcTimeScaleCollection 1414
 - VcUpdateBehaviorCollection 1421
- CreateDataDefinitionField**
 - Method of
 - VcDataDefinitionTable 601
- CreateEntry**
 - Method of
 - VcMap 1183
- CreateHistogram**
 - Method of
 - VcHistogramCollection 1033
- CtrlCXVProcessingEnabled**
 - Property of
 - VcGantt 718
- Ctrl-X, -C, -V 229**
- Current scroll date**
 - graphical element 798, 821
- CurrentHorizontalPagesCount**

Property of
 VcPrinter 1255

CurrentVerticalPagesCount
 Property of
 VcPrinter 1256

CurrentZoomFactor
 Property of
 VcPrinter 1256

Curve
 by index 591
 see also
 VcCurve 554

CurveByIndex
 Method of
 VcCurveCollection 591

CurveByName
 Method of
 VcCurveCollection 591

CurveCollection
 Property of
 VcHistogram 1022
 see also
 VcCurveCollection 588

Cutting marks 410

CuttingMarks
 Property of
 VcPrinter 1256

D

Data
 editing 420

Data binding 27

Data field
 for tooltip text 238
 node 392

Data fields
 node 391

Data modification
 without analysis 724

Data record
 add to collection 614
 all data 606
 by ID 616
 data field 607
 deleting 609
 depending data record not found 869
 enumerator object 617
 ID 609
 Iteration, initial value 616
 iteration, subsequent values 618
 name of associated table 608
 number in collection 613
 related data record 611
 remove from collection 619
 unique ID 618
 update 620
 updating 611

Data recorddata-based object 610

Data table
 add to collection 626
 by index 627
 by name 628
 copy within collection 627
 data record collection 622
 data table field collection 623
 description 623
 enumerator object 604, 629
 Extended data tables 726
 Iteration, primary value 628
 Iteration, subsequent values 629
 name 790
 name 624
 number in collection 625
 update 630

Data table field

- add to collection 639
- associated date table 631
- by index 640
- by name 641
- copying 639
- data type 637
- date format 632
- editable 633
- enumerator object 642
- index 791
- index 634
- iteration, initial value 641
- iteration, subsequent values 642
- name 790
- name 634
- number in collection 638
- primary key 635
- related field index 635

Data tables

- extended 227

Data Tables 75

DataDefinition

- Property of
- VcGantt 718

DataDefinitionField

- see also
- VcDataDefinitionField 595

DataDefinitionFieldByIndex

- Method of
- VcDataDefinitionTable 602

DataDefinitionFieldByName

- Method of
- VcDataDefinitionTable 602

DataDefinitionTable

- Property of
- VcFilter 684

see also

VcDataDefinitionTable 600

DataField

- Property of
- VcDataRecord 607
- VcGroup 972
- VcLink 1155
- VcNode 1205

DataFieldIndex

- Property of
- VcFilterSubCondition 697

DataFieldValue

- Property of
- VcMapEntry 1195

DataRecord

- Method of
- VcGroup 979
- VcLink 1157
- VcNode 1210

see also

VcDataRecord 606

DataRecordByID

- Method of
- VcDataRecordCollection 616

DataRecordCollection

- Property of
- VcDataTable 622
- see also
- VcDataRecordCollection 613

DataRecordEventsEnabled

- Property of
- VcResourceScheduler2 1291

DataTable

- see also
- VcDataTable 622

DataTableByIndex

- Method of

- VcDataTableCollection 627
- DataTableByName**
 - Method of
 - VcDataTableCollection 628
- DataTableCollection**
 - Property of
 - VcGantt 719
 - see also
 - VcDataTableCollection 625
- DataTableField**
 - see also
 - VcDataTableField 631
- DataTableFieldByIndex**
 - Method of
 - VcDataTableFieldCollection 640
- DataTableFieldByName**
 - Method of
 - VcDataTableFieldCollection 641
- DataTableFieldCollection**
 - Property of
 - VcDataTable 623
 - see also
 - VcDataTableFieldCollection 638
- DataTableName**
 - Property of
 - VcDataRecord 608
 - VcDataTableField 631
- Date**
 - corresponding to a x coordinate 800
 - Property of
 - VcDateLine 645
- Date line 87, 425**
 - by index 659
 - DateLineCollection 656
 - font attributes 646
 - font color 646
 - label position 647
 - turning of annotation by 90 degrees 653
- Date Line**
 - Snap target at date 651
- Date line grid**
 - consider daylight saving time 670, 1349
 - line color 667
 - line color map 667, 1081
 - reference date 672, 673
- Date Line Grid**
 - Snap target at date 672
- Date line, individual**
 - data field 646, 654
- Date lines**
 - order 654
- Date Lines**
 - moving 406
- Date output format 225**
- DateDataFieldIndex**
 - Property of
 - VcDateLine 646
- DateFormat**
 - Property of
 - VcDataDefinitionField 595
 - VcDataTableField 632
 - VcPrinter 1257
- DateGridsVisible**
 - Property of
 - VcTimeScale 1408
- DateLine**
 - see also
 - VcDateLine 644
- DateLineByIndex**
 - Method of
 - VcDateLineCollection 659
- DateLineByName**

- Method of
 - VcDateLineCollection 659
- DateLineCollection**
 - Property of
 - VcGantt 719
 - see also
 - VcDateLineCollection 656
- DateLineGrid**
 - Property of
 - VcSection 1367
 - see also
 - VcDateLineGrid 664
- DateLineGridByIndex**
 - Method of
 - VcDateLineGridCollection 679
- DateLineGridByName**
 - Method of
 - VcDateLineGridCollection 679
- DateLineGridCollection**
 - see also
 - VcDateLineGridCollection 676
- DateLineGridName**
 - Property of
 - VcGroupLevelLayout 990
- DateLineGridsVisible**
 - Property of
 - VcGroupLevelLayout 990
- DateLineGridsWithChildGroups**
 - Property of
 - VcGroupLevelLayout 991
- DateLineName**
 - Property of
 - VcGroupLevelLayout 991
 - VcNodeLevelLayout 1221
- DateLinesVisible**
 - Property of
 - VcGroupLevelLayout 991
- VcNodeLevelLayout 1221
- DateLinesWithChildGroups**
 - Property of
 - VcGroupLevelLayout 991
- DateOutputFormat**
 - Property of
 - VcGantt 720
 - VcLineFormatField 1143
 - VcRibbon 1345
- DatesWithHourAndMinute**
 - Property of
 - VcFilter 684
- DayInEndMonth**
 - Property of
 - VcInterval 1046
- DayInStartMonth**
 - Property of
 - VcInterval 1047
- DefaultOperationMaximumInterruptionTime**
 - Property of
 - VcResourceScheduler2 1291
- DefaultPrinterName**
 - Property of
 - VcPrinter 1258
- DefaultResourceCalendarName**
 - Property of
 - VcResourceScheduler2 1292
- DelayTime**
 - Property of
 - VcUpdateBehaviorContext 1427
- Delete**
 - Method of
 - VcDataRecord 609
 - VcGroup 979
 - VcHistogramCollection 1034
 - VcLink 1157

- VcNode 1211
- DeleteEntry**
 - Method of
 - VcMap 1184
- DeleteLinkRecord**
 - Method of
 - VcGantt 789
- DeleteNodeRecord**
 - Method of
 - VcGantt 789
- DeletePoint**
 - Method of
 - VcCurve 579
- Description**
 - Property of
 - VcDataTable 623
- DetectDataTableFieldName**
 - Method of
 - VcGantt 790
- DetectDataTableName**
 - Method of
 - VcGantt 790
- DetectFieldIndex**
 - Method of
 - VcGantt 791
- DetermineIDOfFirstOperationByTaskID**
 - Method of
 - VcResourceScheduler2 1341
- DetermineIDOfLastOperationByTaskID**
 - Method of
 - VcResourceScheduler2 1342
- Diagram**
 - alignment 410
 - background color 722, 786
 - export 69
 - repeat title/table/timescale 408
 - second background color for alternating lines 721
 - tracking space background color 779
- Diagram background color 249**
- DiagramAlternatingRowBackgroundColor**
 - Property of
 - VcGantt 721
- DiagramBackgroundColor**
 - Property of
 - VcGantt 722
- DiagramEnabled**
 - Property of
 - VcPrinter 1258, 1259
- DiagramHistogramHeightRatio**
 - Property of
 - VcGantt 722
- DiagramHistogramHeightRatioEx**
 - Property of
 - VcGantt 723
- DiagramVisible**
 - Property of
 - VcGantt 723
- Dialog**
 - Edit Link 393
 - Page Preview 412
 - Page Setup 407
- Dialog box**
 - Administrate Box Formats 314
 - Administrate Boxes 308
 - Administrate Data Tables 260
 - Administrate Filters 276
 - Administrate Histograms 356
 - Administrate Line formats 282, 284
 - Administrate Maps 303
 - Administrate Update behaviors 257

- Configure Mapping 307
- Edit Box Format 316
- Edit Boxes 313
- Edit Date Line 368
- Edit Filter 278
- Edit Histogram 358
- Edit Layer 267
- Edit Layer Format 272
- Edit Map 305
- Edit Table 325
- Edit Table Format 327
- Edit Time Scale Section 350
- Edit Update behaviors 258
- grouping 288
- Licensing 375
- Line Attributes 332
- Pattern 333
- Select Curve Data Source 362
- Select Ribbon Type 363
- Specification of Texts, Graphics and Legend 370
- Specify Bar Appearance 263
- Specify Calendars 334
- Specify Date Lines 365
- Specify Table 323
- Specify Time Scale 347
- DialogFont**
 - Property of VcGantt 723
- DirectDataWritingModeEnabled**
 - Property of VcGantt 724
- DocumentName**
 - Property of VcPrinter 1259
- Double output format 226**
- Double-click**

- on group 397, 398
- on node 391
- on timescale 403

DoubleOutputFormat

- Property of
 - VcGantt 724
 - VcNumericScale 1230

Drag & Drop 94

Dragging nodes

- consider link types 233

DST 92

DumpConfiguration

- Method of
 - VcGantt 791

Duration

- Property of
 - VcInterval 1047

DurationDataFieldIndex

- Property of
 - VcLayer 1070
 - VcScheduler 1360

E

EarlyEndDateDataFieldIndex

- Property of
 - VcScheduler 1360

EarlyStartDateDataFieldIndex

- Property of
 - VcScheduler 1360

Edit Data

- groups 396

Editable

- Property of
 - VcDataDefinitionField 596
 - VcDataTableField 633

Enabled

- Property of

- VcGantt 725
- End date**
 - calculate 33
- EndDataFieldIndex**
 - Property of
 - VcLayer 1071
- EndDateForAutomaticScheduling**
 - Property of
 - VcGantt 725
 - VcScheduler 1360
- EndDateNotLaterThanDataFieldIndex**
 - Property of
 - VcScheduler 1361
- EndDateTime**
 - Property of
 - VcInterval 1047
- EndLoading**
 - Method of
 - VcGantt 792
- EndMonth**
 - Property of
 - VcInterval 1048
- EndSnapTarget**
 - Property of
 - VcCalendarGrid 524
 - VcLayer 1071
- EndTime**
 - Property of
 - VcInterval 1048
- EndWeekday**
 - Property of
 - VcInterval 1048
- Evaluate**
 - Method of
 - VcFilter 687
- Event argument objects**
 - VcBoxClickingEventArgs 839, 840, 843
 - VcBoxCreatedEventArgs 838
 - VcBoxCreatingEventArgs 839
 - VcBoxModifiedEventArgs 841
 - VcBoxModifyingEventArgs 842
 - VcCalendarGridClickingEventArgs 844
 - VcComponentScrolledEventArgs 846
 - VcComponentScrollingEventArgs 848
 - VcCurveClickingEventArgs 851, 852, 861
 - VcCurveModifyingEventArgs 854, 855
 - VcCurvePointDeletingEventArgs 856, 858
 - VcCurvePointInsertingEventArgs 859, 860
 - VcDataModifiedEventArgs 863
 - VcDataRecordCreatedEventArgs 863
 - VcDataRecordCreatingEventArgs 865
 - VcDataRecordDeletedEventArgs 866
 - VcDataRecordDeletingEventArgs 866
 - VcDataRecordModifiedEventArgs 867
 - VcDataRecordModifyingEventArgs 868
 - VcDataRecordNotFoundEventArgs 869
 - VcDateLineClickingEventArgs 870
 - VcDateLineModifyingEventArgs 869
 - VcDateShowingEventArgs 871
 - VcDiagramClickingEventArgs 876, 878
 - VcDiagramHorizontalScrolledEventArgs 872
 - VcDiagramHorizontalScrollingEventArgs 874

- VcDragCompletingEventArgs 879
- VcDragEventArgs 879
- VcDragStartingEventArgs 880
- VcErrorOccurringEventArgs 881
- VcFieldSelectingEventArgs 881
- VcGroupClickingEventArgs 883, 884, 888
- VcGroupDeletingEventArgs 882, 920
- VcGroupModifiedEventArgs 885
- VcGroupModifyingEventArgs 886
- VcGroupsMarkedEventArgs 889
- VcHelpRequestedEventArgs 890
- VcHistogramClickingEventArgs 892, 893, 894
- VcHistogramCurveNameShowingInMenuEventArgs 891
- VcHistogramsHeightChangedEventArgs 895
- VcHistogramsHeightChangingEventArgs 896
- VcHistogramsHeightChangingEventArgs 897
- VcInPlaceEditorShowingEventArgs 897
- VcInteractionEndedEventArgs 899
- VcInteractionModeChangedEventArgs 901
- VcInteractionModeChangingEventArgs 901
- VcInteractionObjectChangedEventArgs 902
- VcInteractionStartedEventArgs 903
- VcLegendViewClosedEventArgs 905
- VcLinkCreatedEventArgs 905
- VcLinkCreatingEventArgs 906
- VcLinkDeletedEventArgs 907
- VcLinkDeletingEventArgs 908
- VcLinksClickingEventArgs 909, 910, 911
- VcNodeClickingEventArgs 915, 916, 921
- VcNodeCreatedEventArgs 912
- VcNodeCreatingEventArgs 913
- VcNodeDeletedEventArgs 913
- VcNodeDeletingEventArgs 914
- VcNodeModifiedEventArgs 917
- VcNodeModifiedExEventArgs 918
- VcNodeModifyingEventArgs 919
- VcNodesMarkedEventArgs 922
- VcNodesMarkingEventArgs 889, 923
- VcNumericScaleClickingEventArgs 924, 925, 927
- VcNumericScaleRescalingEventArgs 926
- VcObjectDrawingEventArgs 929
- VcObjectDrawnEventArgs 930
- VcResourceSchedulingProgressingEventArgs 931
- VcSashButtonClickedEventArgs 935
- VcStatusLineTextShowingEventArgs 935
- VcTableCaptionClickingEventArgs 936, 937, 938
- VcTableColumnWidthChangedEventArgs 939
- VcTableColumnWidthChangingEventArgs 940
- VcTableColumnWidthOptimizingEventArgs 941
- VcTableWidthChangingEventArgs 942
- VcTableWidthChangingExEventArgs 943
- VcTextEntrySupplyingEventArgs 943
- VcTimeScaleClickingEventArgs 958, 959, 960
- VcTimeScaleSectionRescalingEventArgs 962
- VcTimeScaleSectionRescalingExEventArgs 963

VcTimeScaleSectionStartModifyingEventArgs 964

VcToolTipTextSupplyingEventArgs 965

VcViewComponentsSizeModifiedEventArgs 967

VcWorldViewClosedEventArgs 969

VcZoomFactorModifiedEventArgs 969

Event security check 231

Events 108

KeyDown

VcGantt 836

KeyPress

VcGantt 836

KeyUp

VcGantt 837

VcBoxCreated

VcGantt 838

VcBoxCreating

VcGantt 838

VcBoxLeftClicking

VcGantt 839

VcBoxLeftDoubleClicking

VcGantt 840

VcBoxModified

VcGantt 841

VcBoxModifying

VcGantt 842

VcBoxRightClicking

VcGantt 843

VcCalendarGridRightClicking

VcGantt 844

VcComponentScrolled

VcGantt 845

VcComponentScrolling

VcGantt 848

VcCurveLeftClicking

VcGantt 851

VcCurveLeftDoubleClicking

VcGantt 852

VcCurveModified

VcGantt 853

VcCurveModifying

VcGantt 853

VcCurveModifyingEx

VcGantt 855

VcCurvePointDeleting

VcGantt 856

VcCurvePointDeletingEx

VcGantt 857

VcCurvePointInserting

VcGantt 858

VcCurvePointInsertingEx

VcGantt 860

VcCurveRightClicking

VcGantt 861

VcDataModified

VcGantt 862

VcDataRecordCreated

VcGantt 863

VcDataRecordCreating

VcGantt 864

VcDataRecordDeleted

VcGantt 865

VcDataRecordDeleting

VcGantt 866

VcDataRecordModified

VcGantt 867

VcDataRecordModifying

VcGantt 868

VcDataRecordNotFound

VcGantt 869

VcDateLineModifying

VcGantt 869

- VcDateLineRightClicking
 - VcGantt 870
- VcDateShowing
 - VcGantt 871
- VcDiagramHorizontalScrolled
 - VcGantt 872
- VcDiagramHorizontalScrolling
 - VcGantt 873
- VcDiagramLeftClicking
 - VcGantt 875
- VcDiagramLeftDoubleClicking
 - VcGantt 876
- VcDiagramRightClicking
 - VcGantt 877
- VcDragCompleting
 - VcGantt 878
- VcDragOver
 - VcGantt 879
- VcDragStarting
 - VcGantt 880
- VcErrorOccurring
 - VcGantt 880
- VcFieldSelecting
 - VcGantt 881
- VcGroupDeleting
 - VcGantt 882
- VcGroupLeftClicking
 - VcGantt 883
- VcGroupLeftDoubleClicking
 - VcGantt 884
- VcGroupModified
 - VcGantt 885
- VcGroupModifying
 - VcGantt 885
- VcGroupRightClicking
 - VcGantt 887
- VcGroupsMarked
 - VcGantt 888
- VcGroupsMarking
 - VcGantt 889
- VcHelpRequested
 - VcGantt 890
- VcHistogramCurveNameShowingInMenu
 - VcGantt 891
- VcHistogramLeftClicking
 - VcGantt 892
- VcHistogramLeftDoubleClicking
 - VcGantt 893
- VcHistogramRightClicking
 - VcGantt 894
- VcHistogramsHeightChanged
 - VcGantt 895
- VcHistogramsHeightChanging
 - VcGantt 895
- VcHistogramsHeightChangingEx
 - VcGantt 896
- VcInPlaceEditorShowing
 - VcGantt 897
- VcInteractionEnded
 - VcGantt 899
- VcInteractionModeChanged
 - VcGantt 900
- VcInteractionModeChanging
 - VcGantt 901
- VcInteractionObjectChanged
 - VcGantt 902
- VcInteractionStarted
 - VcGantt 903
- VcLegendViewClosed
 - VcGantt 904
- VcLinkCreated
 - VcGantt 905
- VcLinkCreating

- VcGantt 906
- VcLinkDeleted
 - VcGantt 907
- VcLinkDeleting
 - VcGantt 907
- VcLinksLeftClicking
 - VcGantt 908
- VcLinksLeftDoubleClicking
 - VcGantt 909
- VcLinksRightClicking
 - VcGantt 910
- VcNodeCreated
 - VcGantt 911
- VcNodeCreating
 - VcGantt 912
- VcNodeDeleted
 - VcGantt 913
- VcNodeDeleting
 - VcGantt 914
- VcNodeLeftClicking
 - VcGantt 915
- VcNodeLeftDoubleClicking
 - VcGantt 916
- VcNodeModified
 - VcGantt 917
- VcNodeModifiedEx
 - VcGantt 917
- VcNodeModifying
 - VcGantt 919
- VcNodeResizeStarting
 - VcGantt 920
- VcNodeRightClicking
 - VcGantt 921
- VcNodesMarked
 - VcGantt 922
- VcNodesMarking
 - VcGantt 923
- VcNumericScaleLeftClicking
 - VcGantt 924
- VcNumericScaleLeftDoubleClicking
 - VcGantt 925
- VcNumericScaleRescaling
 - VcGantt 926
- VcNumericScaleRightClicking
 - VcGantt 927
- VcObjectDrawing
 - VcGantt 928
- VcObjectDrawn
 - VcGantt 930
- VcResourceSchedulingProgressing
 - VcGantt 931
- VcResourceSchedulingWarning
 - VcGantt 932
- VcSashButtonClicked
 - VcGantt 934
- VcStatusLineTextShowing
 - VcGantt 935
- VcTableCaptionLeftClicking
 - VcGantt 936
- VcTableCaptionLeftDoubleClicking
 - VcGantt 937
- VcTableCaptionRightClicking
 - VcGantt 938
- VcTableColumnWidthChanged
 - VcGantt 939
- VcTableColumnWidthChanging
 - VcGantt 940
- VcTableColumnWidthOptimizing
 - VcGantt 941
- VcTableWidthChanging
 - VcGantt 941
- VcTableWidthChangingEx
 - VcGantt 942
- VcTextEntrySupplying

VcGantt 943
 VcTimeScaleEndModified
 VcGantt 957
 VcTimeScaleLeftClicking
 VcGantt 957
 VcTimeScaleLeftDoubleClicking
 VcGantt 958
 VcTimeScaleModified
 VcGantt 959
 VcTimeScaleRightClicking
 VcGantt 959
 VcTimeScaleSectionRescaled
 VcGantt 961
 VcTimeScaleSectionRescaledEx
 VcGantt 961
 VcTimeScaleSectionRescaling
 VcGantt 961
 VcTimeScaleSectionRescalingEx
 VcGantt 962
 VcTimeScaleSectionStartModifying
 VcGantt 963
 VcTimeScaleStartModified
 VcGantt 965
 VcToolTipTextSupplying
 VcGantt 965
 VcViewComponentsSizeModified
 VcGantt 967
 VcWorldViewClosed
 VcGantt 968
 VcZoomFactorModified
 VcGantt 969
EventsSecurityCheck
 Property of
 VcGantt 726
Export graphic 418
ExportGraphicsToFileEx
 Method of

VcGantt 792
ExtendedDataTablesEnabled
 Property of
 VcGantt 726
ExtendedEditingBehavior
 Property of
 VcGantt 727

F

field contents
 modify 399
FieldsSeparatedByLines
 Property of
 VcBoxFormat 485
 VcTableFormat 1381
FieldText
 Property of
 VcBox 464
FilePath
 Property of
 VcGantt 727
FillReference1BackgroundColor
 Property of
 VcCurve 556
FillReference1Name
 Property of
 VcCurve 556
FillReference1Pattern
 Property of
 VcCurve 557
FillReference1PatternColor
 Property of
 VcCurve 561
FillReference2Color
 Property of
 VcCurve 561
FillReference2Name

- Property of
 - VcCurve 562
- FillReference2Pattern**
 - Property of
 - VcCurve 562
- FillReference2PatternColor**
 - Property of
 - VcCurve 566
- Filter 109**
 - by index 693
 - comparison value 279
 - see also
 - VcFilter 683
 - using 46
- FilterByIndex**
 - Method of
 - VcFilterCollection 693
- FilterByName**
 - Method of
 - VcFilterCollection 693
- FilterCollection**
 - Property of
 - VcGantt 728
 - see also
 - VcFilterCollection 690
- FilterName**
 - Property of
 - VcCurve 567
 - VcFilterSubCondition 698
 - VcLayer 1071
 - VcLinkAppearance 1160
 - VcTableFormat 1382
- Filters**
 - administration 276
 - editing 278
- FilterSubCondition**
 - see also
 - VcFilterSubCondition 696
- FirstBox**
 - Method of
 - VcBoxCollection 482
- FirstCalendar**
 - Method of
 - VcCalendarCollection 518
- FirstCalendarGrid**
 - Method of
 - VcCalendarGridCollection 542
- FirstCalendarProfile**
 - Method of
 - VcCalendarProfileCollection 551
- FirstCurve**
 - Method of
 - VcCurveCollection 592
- FirstDataDefinitionField**
 - Method of
 - VcDataDefinitionTable 603
- FirstDataRecord**
 - Method of
 - VcDataRecordCollection 616
- FirstDataTable**
 - Method of
 - VcDataTableCollection 628
- FirstDataTableField**
 - Method of
 - VcDataTableFieldCollection 641
- FirstDateLine**
 - Method of
 - VcDateLineCollection 660
- FirstDateLineGrid**
 - Method of
 - VcDateLineGridCollection 680
- FirstFilter**
 - Method of
 - VcFilterCollection 693

FirstFormat

Method of

VcBoxFormatCollection 492

VcLineFormatCollection 1138

VcTableFormatCollection 1388

FirstGroup

Method of

VcGroupCollection 983

FirstGroupLevelLayout

Method of

VcGroupLevelLayoutCollection
1011

FirstHistogram

Method of

VcHistogramCollection 1034

FirstInterval

Method of

VcIntervalCollection 1062

FirstLayer

Method of

VcLayerCollection 1110

FirstLink

Method of

VcLinkCollection 1177

FirstLinkAppearance

Method of

VcLinkAppearanceCollection 1171

FirstMap

Method of

VcMapCollection 1189

FirstMapEntry

Method of

VcMap 1185

FirstNode

Method of

VcNodeCollection 1217

FirstNumericScale

Method of

VcNumericScaleCollection 1245

FirstTable

Method of

VcTableCollection 1378

FirstTimeScale

Method of

VcTimeScaleCollection 1414

FirstUpdateBehavior

Method of

VcUpdateBehaviorCollection 1423

FitChartIntoView

Method of

VcGantt 795

FitHistogramsIntoView

Method of

VcGantt 795

FitRangeIntoView

Method of

VcGantt 796

VcHistogram 1029

FitToPage

Property of

VcPrinter 1259

FoldingMarksType

Property of

VcPrinter 1260

Font

Property of

VcDateLine 646

VcNumericScale 1230

VcRibbon 1347

VcTimeScale 1409

FontAntiAliasingEnabled

Property of

VcGantt 728

FontBody

Property of
 VcMapEntry 1196

FontColor
 Property of
 VcDateLine 646
 VcNumericScale 1231
 VcRibbon 1347
 VcTimeScale 1409

FontName
 Property of
 VcMapEntry 1196

Fonts 447
 anti-aliasing 728

FontSize
 Property of
 VcMapEntry 1197

Form
 adjusting 25

Format
 Property of
 VcLayer 1072

FormatByIndex
 Method of
 VcBoxFormatCollection 493
 VcLineFormatCollection 1139
 VcTableFormatCollection 1388

FormatByName
 Method of
 VcBoxFormatCollection 493
 VcLineFormatCollection 1139
 VcTableFormatCollection 1389

FormatField
 Property of
 VcBoxFormat 486
 VcLayerFormat 1114
 VcLineFormat 1133
 VcTableFormat 1382

FormatFieldCount
 Property of
 VcBoxFormat 486
 VcLayerFormat 1115
 VcLineFormat 1134
 VcTableFormat 1383

FormatName
 Property of
 VcBox 464
 VcBoxFormatField 498
 VcDateLineGrid 666
 VcLayerFormatField 1119
 VcLineFormatField 1145
 VcTableFormatField 1393

FreeFloatDataFieldIndex
 Property of
 VcScheduler 1361

Full diagram 417, 420

FullUsageOfPlanningUnitsEnabled
 Property of
 VcResourceScheduler2 1292

G

German Version 19

GetActualExtent
 Method of
 VcBox 474

GetActualScaleValues
 Method of
 VcHistogram 1030

GetAValueFromARGB
 Method of
 VcGantt 797

GetBValueFromARGB
 Method of
 VcGantt 797

GetCurrentComponentStart

- Method of
 - VcGantt 798
- GetCurrentViewDates**
 - Method of
 - VcGantt 799
- GetCurrentYValues**
 - Method of
 - VcHistogram 1030
- GetDate**
 - Method of
 - VcGantt 800
- GetDateAsString**
 - Method of
 - VcGantt 800
- GetEndOfPreviousWorktime**
 - Method of
 - VcCalendar 510
- GetEnumerator**
 - Method of
 - VcBoxCollection 482
 - VcBoxFormat 488
 - VcBoxFormatCollection 494
 - VcCalendarCollection 518
 - VcCalendarGridCollection 542
 - VcCurveCollection 592
 - VcDataDefinitionTable 604
 - VcDataRecordCollection 617
 - VcDataTableCollection 629
 - VcDataTableFieldCollection 642
 - VcDateLineCollection 661
 - VcDateLineGridCollection 680
 - VcFilter 688
 - VcFilterCollection 694
 - VcFilterSubCondition 700
 - VcGroupCollection 983
 - VcGroupLevelLayoutCollection 1011
 - VcHistogramCollection 1035
 - VcLayerCollection 1111
 - VcLayerFormat 1115
 - VcLinkAppearanceCollection 1172
 - VcLinkCollection 1177
 - VcMapCollection 1190
 - VcNodeCollection 1217
 - VcNumericScaleCollection 1246
 - VcTableCollection 1378
 - VcTableFormat 1386
 - VcTableFormatCollection 1389
 - VcTimeScaleCollection 1415
 - VcUpdateBehaviorCollection 1424
- Property of
 - VcMap 1181
- GetFirstOverload**
 - Method of
 - VcCurve 579
- GetFirstOverloadEx**
 - Method of
 - VcCurve 581
- GetGValueFromARGB**
 - Method of
 - VcGantt 801
- GetLinkByID**
 - Method of
 - VcGantt 802
- GetLinkByNodeIDs**
 - Method of
 - VcGantt 802
- GetMapEntry**
 - Method of
 - VcMap 1185
- GetNewUniqueID**
 - Method of
 - VcDataRecordCollection 618
- GetNextIntervalBorder**

- Method of
 - VcCalendar 510
- GetNextOverload**
 - Method of
 - VcCurve 582
- GetNextOverloadEx**
 - Method of
 - VcCurve 583
- GetNodeByID**
 - Method of
 - VcGantt 803
- GetPositionInView**
 - Method of
 - VcNode 1211
- GetPreviousIntervalBorder**
 - Method of
 - VcCalendar 511
- GetRValueFromARGB**
 - Method of
 - VcGantt 803
- GetStartOfInterval**
 - Method of
 - VcCalendar 511
- GetStartOfNextWorktime**
 - Method of
 - VcCalendar 512
- GetTopLeftPixel**
 - Method of
 - VcBox 475
- GetValues**
 - Method of
 - VcCurve 584
- GetValuesEx**
 - Method of
 - VcCurve 585
- GetViewComponentSize**
 - Method of
 - VcGantt 804
- GetXOffset**
 - Method of
 - VcBox 475
- Graphic**
 - Export 69, 418
- Graphical element**
 - current scroll date 798, 821
- Graphics Format 111**
- GraphicsFileName**
 - Property of
 - VcBoundingBox 455
 - VcLayer 1072
 - VcMapEntry 1197
 - VcTableFormatField 1393
- GraphicsFileNameDataFieldIndex**
 - Property of
 - VcLayer 1074
 - VcTableFormatField 1394
- GraphicsFileNameMapName**
 - Property of
 - VcLayer 1075
 - VcTableFormatField 1394
- GraphicsHeight**
 - Property of
 - VcBoxFormatField 499
 - VcTableFormatField 1395
- Grid 425**
- Group**
 - collapsed 988
 - ID 974
 - name 994
 - see also
 - VcGroup 971
 - show date line grids 990
 - visible 1008
- Group level layout**

- calendar grid name 989, 990, 992
- calendar grids for sub groups 989
- date line grids for sub groups 991
- Group level Layout**
 - map name for bodies collapsed 989
 - map name for invisible groups 993
- Group levels**
 - show group nodes 992
 - show groups 992
- Group node**
 - visible 291
- Group title row**
 - background color 998
 - fill pattern 999
- GroupByName**
 - Method of
 - VcGroupCollection 984
- GroupCollection**
 - Property of
 - VcGantt 729
 - see also
 - VcGroupCollection 982
- GroupDataFieldIndex**
 - Property of
 - VcGroupLevelLayout 992
- Groupi level layout**
 - grouping level 993
- Grouping 115, 292, 295, 296**
 - all nodes of all groups in one line/in separate lines/expanded/collapsed 118
 - calendar 293
 - calendar grid 989, 991, 1022, 1221
 - collapsing/expanding allowed 295, 993
 - date line 991
 - date lines for child groups 991
 - date lines for nodes 1221
 - initially collapsed 295
 - interactive regrouping of nodes 117
 - making overlapping activities in a group visible 439
 - Separation line color 1004
 - separation line color map name 1004
 - separation lines 294
 - sorting order 292, 293, 1007, 1227
- Grouping level**
 - line thickness 1006, 1226
 - line type 1227
 - name of nodes' calendar grid 1221
 - separation line color 1004, 1225
- Grouping levels**
 - data field index for color maps of nodes 1222, 1223
 - data field index for maps of nodes 1224
 - name of color maps of nodes 1222, 1224
 - name of maps of nodes 1224
 - pattern color of nodes 1223
 - row pattern of nodes 1223
 - show calendar grids of nodes 1221
 - show separation lines 1004, 1225, 1226
 - show separation lines at top 1005
 - specify row background color of nodes 1222
- GroupingDataFieldIndex**
 - Property of
 - VcGantt 729
- GroupingLevel**
 - Property of
 - VcGroup 973
- GroupingModificationsAllowed**
 - Property of
 - VcGantt 730
- GroupInvisible**

Property of
 VcGroup 974

GroupLevelLayout
 see also
 VcGroupLevelLayout 986

GroupLevelLayoutByIndex
 Method of
 VcGroupLevelLayoutCollection 1012

GroupLevelLayoutByName
 Method of
 VcGroupLevelLayoutCollection 1012

GroupLevelLayoutCollection
 Property of
 VcGantt 730
 see also
 VcGroupLevelLayoutCollection 1009

GroupNodes
 Method of
 VcGantt 805

GroupNodesVisible
 Property of
 VcGroupLevelLayout 992

GroupOptimizationOnInteractionsEnabled
 Property of
 VcGantt 731

Groups
 automatically collapse 987, 1014
 automatically expand 988, 1015
 automatically restore 997, 1018
 editing 827
 editing data 396
 move vertically in the diagram 994
 move vertically in the table 994
 optimization on interactions 234

page break 997

GroupsInvisible

Property of
 VcGroupLevelLayout 992

GroupsInvisibleCollapsedMapName

Property of
 VcGroupLevelLayout 993

GroupsInvisibleDataFieldIndex

Property of
 VcGroupLevelLayout 993

GroupSortingDataFieldIndex

Property of
 VcGantt 731

GroupSortingOrder

Property of
 VcGantt 732

H

Height

Property of
 VcLayer 1077
 VcLegendView 1126
 VcRect 1274
 VcWorldView 1432

Height ratio

diagram area/histogram 249

HeightActualValue

Property of
 VcLegendView 1127
 VcWorldView 1433

HeightDataFieldIndex

Property of
 VcLayer 1077

HeightMapName

Property of
 VcLayer 1078

Hidden

- Property of
 - VcDataDefinitionField 597
 - VcDataTableField 633
- Hierarchical Order 121**
- Hierarchy 289, 290, 291**
 - all nodes of one group of this level in one line 1017
 - moving nodes interactively 122
 - moving summary bars interactively 123
 - page break after groups 1017
- Hierarchy level**
 - specify page break for certain levels 1016
- Hierarchy level layout**
 - Map name 1016
- Hierarchy level layout**
 - Bodies collapsed 1015
- Hierarchy levels**
 - line thickness 1019
 - line type 1020
 - separation line color 1018
 - show node separation lines 1017
 - show separation lines 1019
 - show summary bars 1020
 - sorting index 1016
- HierarchyDataFieldIndex**
 - Property of
 - VcGantt 732
 - VcHierarchyLevelLayout 1016
- HierarchyLevelLayout**
 - Property of
 - VcGantt 733
 - see also
 - VcHierarchyLevelLayout 1014
- Histogram 124, 248**
 - actual values of numeric scale 1030
 - all histograms in a window 795
 - assign calendar 1022
 - background color 1025
 - creating 50
 - curve collection 1022
 - curves 360
 - high-low curve values 1030
 - matching numeric scale 1029
 - maximum value of the numeric scale 1023
 - minimum value of the numeric scale 1024
 - modification of diagram/histogram height ratio 895, 896
 - name 1023
 - order 1030
 - Property of
 - VcCurve 567
 - VcNumericScale 1231
 - ratio of the histogram height to the total diagram height 723
 - scale collection 1024
 - see also
 - VcHistogram 1021
 - separation line color 734
 - visible 1028**
- HistogramByIndex**
 - Method of
 - VcHistogramCollection 1035
- HistogramByName**
 - Method of
 - VcHistogramCollection 1035
- HistogramCollection**
 - Property of
 - VcGantt 733
 - see also
 - VcHistogramCollection 1032
- Histogramm**
 - fill pattern 1025

HistogramSeparationLineColor

Property of
VcGantt 734

HorAlignment

Property of
VcDateLineGrid 666

HorizontalMovementWhileDraggingAllowed

Property of
VcGantt 734

HorizontalOffset

Property of
VcLayer 1079

HTML page 215**ID**

Property of
VcDataRecord 609
VcGroup 974
VcLink 1156
VcNode 1206

Identifiable

Property of
VcCalendarGrid 525
VcDateLine 647

IdentifyField

Method of
VcGantt 806

IdentifyFormatField

Method of
VcBox 475
VcTable 1376

IdentifyInterval

Method of
VcCalendarGrid 536

IdentifyLayerAt

Method of
VcGantt 807

IdentifyObject

Method of
VcDataRecord 610
VcGantt 810

IdentifyObjectAt

Method of
VcGantt 811

ImportConfiguration

Method of
VcGantt 813

InbuiltMouseCursorWhileDraggingEnabled

Property of
VcGantt 734

IncomingLinks

Property of
VcNode 1206

IndentColumn

Property of
VcTableFormat 1383

IndentWidth

Property of
VcTableFormat 1384

Index

Property of
VcBoxFormatField 499
VcDataDefinitionField 597
VcDataTableField 634
VcFilterSubCondition 698
VcLayerFormatField 1119
VcLineFormatField 1145
VcTableFormatField 1395

InfoWindow

Property of
VcGantt 735

- see also
 - VcInfoWindow 1037
- InflowWindow 735**
- InitializeForWebService**
 - Method of
 - VcGantt 813
- InitialRowCount**
 - Property of
 - VcGantt 735
- In-place editing**
 - groups in diagram 228
 - groups in table 228
 - nodes in diagram 227
 - nodes in table 227
- InPlaceEditingOnGroupsInDiagramEnabled**
 - Property of
 - VcGantt 735
- InPlaceEditingOnGroupsInTableEnabled**
 - Property of
 - VcGantt 736
- InPlaceEditingOnNodesInDiagramEnabled**
 - Property of
 - VcGantt 736
- InPlaceEditingOnNodesInTableEnabled**
 - Property of
 - VcGantt 737
- InsertLinkRecord**
 - Method of
 - VcGantt 813
- InsertNodeRecord**
 - Method of
 - VcGantt 814
- Installation 13**
- InteractionMode**
 - Property of
 - VcGantt 738
- Interactions**
 - activities 41
 - Show snap lines 242
 - Show snap markings 242
 - Specify snap targets 241
 - table and diagram area 39
- Interactive moving of layers/nodes**
 - earliest start time 1081
 - latest end time 1081
- Internet 69, 219, 376**
- Interval**
 - Add 1061
 - annotation of the time ribbon 1057
 - background color 1046
 - by index 1063
 - calendar profile 1046
 - copy 1062
 - day of end month 1046
 - day of start month 1047
 - duration 1047
 - end date and time 1047
 - end month 1048
 - end time 1048
 - end weekday 1048
 - first weekday 1056
 - line type 1050
 - name 1051
 - number 1061
 - order 1058
 - pattern 1051
 - pattern color 1055
 - remove 1064
 - retrieving an interval by its name 1063
 - see also

- VcInterval 1044
- start date and time 1055
- start month 1056
- start time 1056
- time unit 1057
- type 1057
- usage of graphical attributes 535, 1058
- IntervalByIndex**
 - Method of
 - VcIntervalCollection 1063
- IntervalByName**
 - Method of
 - VcIntervalCollection 1063
- IntervalCollection**
 - Property of
 - VcCalendar 506
 - VcCalendarProfile 545
 - see also
 - VcIntervalCollection 1060
- Intervall**
 - erstes Intervall 1062
 - nächststes Intervall 1063
- Intervall collection**
 - update 1064
- Intervalle**
 - line thickness 1050
- Intervals**
 - line color 1049
 - Specify 336, 340, 342, 343, 345
- IsEditable**
 - Property of
 - VcUpdateBehavior 1417
 - VcUpdateBehaviorContext 1428
- IsValid**
 - Method of
 - VcFilter 688

- VcFilterSubCondition 700
- IsWorktime**
 - Method of
 - VcCalendar 512

K

- KeepingNodesTogetherDataFieldIndex**
 - Property of
 - VcGantt 738
- KeyDown**
 - Event of
 - VcGantt 836
- KeyPress**
 - Event of
 - VcGantt 836
- KeyUp**
 - Event of
 - VcGantt 837

L

- LabelPosition**
 - Property of
 - VcDateLine 647
- LabelSizeDependence**
 - Property of
 - VcLayer 1079
- Language 176**
- LateEndDateDataFieldIndex**
 - Property of
 - VcScheduler 1361
- LateStartDateDataFieldIndex**
 - Property of
 - VcScheduler 1362
- Layer 160**
 - 3D effect 267
 - by index 1111

- duration 270
- end date field 270
- height 267
- layer shape 267
- moving 383, 388
- order 1106
- see also
 - VcLayer 1065
- Start date as snap target 1102
- start date field 270
- using 43
- visible in legend 265
- LayerByIndex**
 - Method of
 - VcLayerCollection 1111
- LayerByName**
 - Method of
 - VcLayerCollection 1111
- LayerCollection**
 - Property of
 - VcGantt 739
 - see also
 - VcLayerCollection 1108
- LayerFormat**
 - see also
 - VcLayerFormat 1114
- LayerFormatField**
 - see also
 - VcLayerFormatField 1117
- LayerName**
 - Property of
 - VcCurve 568
- LayersWithNonWorkInterval**
 - Property of
 - VcGantt 739
- LeavingControlWhileDraggingAllowed**
 - Property of
 - VcGantt 740
- Left**
 - Property of
 - VcLegendView 1127
 - VcRect 1275
 - VcWorldView 1433
- LeftActualValue**
 - Property of
 - VcLegendView 1128
 - VcWorldView 1434
- LeftMargin**
 - Property of
 - VcLayerFormatField 1119
 - VcTableFormatField 1395
- LeftTable**
 - Property of
 - VcGantt 740
- LeftTableDiagramWidthRatio**
 - Property of
 - VcGantt 741
- LeftTableDiagramWidthRatioEx**
 - Property of
 - VcGantt 741
- Legend**
 - Arrangement 373, 374
 - extended attributes 373
 - Font 374
 - Title 373
- Legend view 418**
- Legend View 163**
- LegendElementsArrangement**
 - Property of
 - VcBoundingBox 456
- LegendElementsBottomMargin**
 - Property of
 - VcBoundingBox 456

LegendElementsMaximumColumnCount

Property of
VcBoundingBox 456

LegendElementsMaximumRowCount

Property of
VcBoundingBox 457

LegendElementsTopMargin

Property of
VcBoundingBox 457

LegendFont

Property of
VcBoundingBox 457

LegendText

Property of
VcLayer 1080
VcMapEntry 1199

LegendTitle

Property of
VcBoundingBox 458

LegendTitleFont

Property of
VcBoundingBox 458

LegendTitleVisible

Property of
VcBoundingBox 459

LegendView

Property of
VcGantt 742
see also
VcLegendView 1125

Level

Property of
VcGroupLevelLayout 993

LevelMaximumForPagebreaks

Property of
VcHierarchyLevelLayout 1016

Licencing 436**Licensing 16, 234**

Request License Information 377

Lilne format field

fill pattern 1147

line color map 1084**Line formats**

administration 282, 284

Line grid 354**Line Grids**

administrate 300

LineColor

Property of
VcBox 465
VcCalendarGrid 525
VcCurve 568
VcDateLine 647
VcDateLineGrid 667
VcInterval 1049
VcLayer 1080
VcLinkAppearance 1161
VcNumericScale 1232
VcSection 1367

LineColorDataFieldIndex

Property of
VcCalendarGrid 526
VcDateLineGrid 667
VcLayer 1080

LineColorMapName

Property of
VcCalendarGrid 526
VcDateLineGrid 667
VcLayer 1081

LineFormat

see also
VcLineFormat 1133

LineFormatCollection

Property of

VcGantt 742

see also

VcLineFormatCollection 1136

LineFormatField

see also

VcLineFormatField 1142

LineThickness

Property of

VcBox 465

VcCalendarGrid 526

VcCurve 569

VcDateLine 648

VcDateLineGrid 668

VcInterval 1049

VcLinkAppearance 1161

LineType

Property of

VcBox 466

VcCalendarGrid 527

VcCurve 570

VcDateLine 649

VcDateLineGrid 669

VcInterval 1050

VcLinkAppearance 1163

Link 166

appearances 165

editing data 393

ID 1156

predecessor node 253

see also

VcLink 1154

show 319

successor node 254

type 254

Link appearance

order 1168

Link appearance collection

add 1170

add by specification 1171

copy 1171

delete 1174

Link appearance object

by index 1172

by name 1173

enumerator object 1172

iteration, initial value 1171

iteration, subsequent value 1173

layer of predecessor 1165

layer of successor 1166

number in collection 1169

predecessor port symbol 1165

routing type 1165

successor port symbol 1167

visible 1167

LinkAppearance

see also

VcLinkAppearance 1160

LinkAppearanceByIndex

Method of

VcLinkAppearanceCollection 1172

LinkAppearanceByName

Method of

VcLinkAppearanceCollection 1173

LinkAppearanceCollection

Property of

VcGantt 743

see also

VcLinkAppearanceCollection 1169

LinkCollection

Property of

VcGantt 743

see also

VcLinkCollection 1176

LinkDataTableName

Property of
VcResourceScheduler2 1293

LinkDurationDataFieldIndex

Property of
VcScheduler 1362

LinkDurationFieldIndex

Property of
VcResourceScheduler2 1295

LinkPredecessorDataFieldIndex

Property of
VcGantt 743

LinkPredecessorOperationIDFieldIndex

Property of
VcResourceScheduler2 1295

LinkPredecessorTaskIDFieldIndex

Property of
VcResourceScheduler2 1296

Links

Administrative Link Appearances 319
rounded slants 232, 233

LinksDataTableName

Property of
VcGantt 745

LinkSuccessorDataFieldIndex

Property of
VcGantt 746

LinkSuccessorOperationIDFieldIndex

Property of
VcResourceScheduler2 1297

LinkSuccessorTaskIDFieldIndex

Property of
VcResourceScheduler2 1297

LinkTypeDataFieldIndex

Property of
VcGantt 747

Live Update 170**Load**

Method of
VcGantt 815

M**MajorTicks**

Property of
VcNumericScale 1232
VcRibbon 1348

MajorTicksEx

Property of
VcNumericScale 1233

MakeARGB

Method of
VcGantt 816

Map 178

by index 1190
see also
VcMap 1180

MapByIndex

Method of
VcMapCollection 1190

MapByName

Method of
VcMapCollection 1190

MapCollection

Property of
VcGantt 748
see also
VcMapCollection 1187

MapEntry

see also
VcMapEntry 1194

Maps

Specifying value ranges by using
filters 1180

Margins 411

MarginsShownInInches

Property of
VcPrinter 1262

Marked

Property of
VcBox 468
VcCurve 571
VcGroup 975
VcNode 1207

MarkedNodesFilter

Property of
VcFilterCollection 691

MarkingColor

Property of
VcWorldView 1434

MaxHorizontalPagesCount

Property of
VcPrinter 1262

MaximumEndDataFieldIndex

Property of
VcLayer 1081

MaximumTextLineCount

Property of
VcBoxFormatField 500
VcTableFormatField 1396

MaxVerticalPagesCount

Property of
VcPrinter 1263

Methods

Add
VcBoxCollection 479
VcBoxFormatCollection 491
VcCalendarCollection 516
VcCalendarGridCollection 539
VcCalendarProfileCollection 549
VcCurveCollection 589

VcDataRecordCollection 614

VcDataTableCollection 626

VcDataTableFieldCollection 639

VcDateLineCollection 657

VcDateLineGridCollection 677

VcFilterCollection 691

VcGroupLevelLayoutCollection
1010

VcIntervalCollection 1061

VcLayerCollection 1109

VcLineFormatCollection 1137

VcLinkAppearanceCollection 1170

VcMapCollection 1188

VcUpdateBehaviorCollection 1421

AddBySpecification

VcBoxCollection 479

VcBoxFormatCollection 491

VcCalendarCollection 516

VcCalendarGridCollection 539

VcCalendarProfileCollection 549

VcCurveCollection 590

VcDateLineCollection 657

VcDateLineGridCollection 677

VcFilterCollection 692

VcGroupLevelLayoutCollection
1010

VcIntervalCollection 1061

VcLayerCollection 1109

VcLineFormatCollection 1137

VcLinkAppearanceCollection 1171

VcMapCollection 1188

VcUpdateBehaviorCollection 1422

AddDuration

VcCalendar 508

AddSubCondition

VcFilter 686

AnchorToNode

- VcBox 473
- BorderBox
 - VcBorderArea 452
- BoxByIndex
 - VcBoxCollection 480
- BoxByName
 - VcBoxCollection 481
- CalcDuration
 - VcCalendar 509
- CalculateCurrentWidth
 - VcLayer 1106
- CalculateLineCount
 - VcLayerFormatField 1124
- CalendarByIndex
 - VcCalendarCollection 517
- CalendarByName
 - VcCalendarCollection 517
- CalendarGridByIndex
 - VcCalendarGridCollection 540
- CalendarGridByName
 - VcCalendarGridCollection 541
- CalendarProfileByIndex
 - VcCalendarProfileCollection 550
- CalendarProfileByName
 - VcCalendarProfileCollection 550
- Clear
 - VcCalendar 509
 - VcCurve 578
- ConvertDistance
 - VcGantt 788
- Copy
 - VcBoxCollection 481
 - VcBoxFormatCollection 492
 - VcCalendarCollection 517
 - VcCalendarGridCollection 541
 - VcCalendarProfileCollection 550
 - VcCurveCollection 590
 - VcDataTableCollection 626
 - VcDataTableFieldCollection 639
 - VcDateLineCollection 658
 - VcDateLineGridCollection 678
 - VcFilterCollection 692
 - VcGroupLevelLayoutCollection 1011
 - VcIntervalCollection 1062
 - VcLayerCollection 1110
 - VcLineFormatCollection 1138
 - VcLinkAppearanceCollection 1171
 - VcMapCollection 1189
 - VcUpdateBehaviorCollection 1422
- CopyFormatField
 - VcBoxFormat 488
 - VcLayerFormat 1115
 - VcLineFormat 1135
- CopySubCondition
 - VcFilter 687
- CreateDataDefinitionField
 - VcDataDefinitionTable 601
- CreateEntry
 - VcMap 1183
- CreateHistogram
 - VcHistogramCollection 1033
- CurveByIndex
 - VcCurveCollection 591
- CurveByName
 - VcCurveCollection 591
- DataDefinitionFieldByIndex
 - VcDataDefinitionTable 602
- DataDefinitionFieldByName
 - VcDataDefinitionTable 602
- DataRecord
 - VcGroup 979
 - VcLink 1157
 - VcNode 1210

- DataRecordById
 - VcDataRecordCollection 616
- DataTableByIndex
 - VcDataTableCollection 627
- DataTableByName
 - VcDataTableCollection 628
- DataTableFieldByIndex
 - VcDataTableFieldCollection 640
- DataTableFieldByName
 - VcDataTableFieldCollection 641
- DateLineByIndex
 - VcDateLineCollection 659
- DateLineByName
 - VcDateLineCollection 659
- DateLineGridByIndex
 - VcDateLineGridCollection 679
- DateLineGridByName
 - VcDateLineGridCollection 679
- Delete
 - VcDataRecord 609
 - VcGroup 979
 - VcHistogramCollection 1034
 - VcLink 1157
 - VcNode 1211
- DeleteEntry
 - VcMap 1184
- DeleteLinkRecord
 - VcGantt 789
- DeleteNodeRecord
 - VcGantt 789
- DeletePoint
 - VcCurve 579
- DetectDataTableFieldName
 - VcGantt 790
- DetectDataTableName
 - VcGantt 790
- DetectFieldIndex
 - VcGantt 791
- DetermineIDOfFirstOperationByTaskID
 - VcResourceScheduler2 1341
- DetermineIDOfLastOperationByTaskID
 - VcResourceScheduler2 1342
- DumpConfiguration
 - VcGantt 791
- EndLoading
 - VcGantt 792
- Evaluate
 - VcFilter 687
- ExportGraphicsToFileEx
 - VcGantt 792
- FilterByIndex
 - VcFilterCollection 693
- FilterByName
 - VcFilterCollection 693
- FirstBox
 - VcBoxCollection 482
- FirstCalendar
 - VcCalendarCollection 518
- FirstCalendarGrid
 - VcCalendarGridCollection 542
- FirstCalendarProfile
 - VcCalendarProfileCollection 551
- FirstCurve
 - VcCurveCollection 592
- FirstDataDefinitionField
 - VcDataDefinitionTable 603
- FirstDataRecord
 - VcDataRecordCollection 616
- FirstDataTable
 - VcDataTableCollection 628
- FirstDataTableField
 - VcDataTableFieldCollection 641

- FirstDateLine
 - VcDateLineCollection 660
- FirstDateLineGrid
 - VcDateLineGridCollection 680
- FirstFilter
 - VcFilterCollection 693
- FirstFormat
 - VcBoxFormatCollection 492
 - VcLineFormatCollection 1138
 - VcTableFormatCollection 1388
- FirstGroup
 - VcGroupCollection 983
- FirstGroupLevelLayout
 - VcGroupLevelLayoutCollection 1011
- FirstHistogram
 - VcHistogramCollection 1034
- FirstInterval
 - VcIntervalCollection 1062
- FirstLayer
 - VcLayerCollection 1110
- FirstLink
 - VcLinkCollection 1177
- FirstLinkAppearance
 - VcLinkAppearanceCollection 1171
- FirstMap
 - VcMapCollection 1189
- FirstMapEntry
 - VcMap 1185
- FirstNode
 - VcNodeCollection 1217
- FirstNumericScale
 - VcNumericScaleCollection 1245
- FirstTable
 - VcTableCollection 1378
- FirstTimeScale
 - VcTimeScaleCollection 1414
- FirstUpdateBehavior
 - VcUpdateBehaviorCollection 1423
- FitChartIntoView
 - VcGantt 795
- FitHistogramsIntoView
 - VcGantt 795
- FitRangeIntoView
 - VcGantt 796
 - VcHistogram 1029
- FormatByIndex
 - VcBoxFormatCollection 493
 - VcLineFormatCollection 1139
 - VcTableFormatCollection 1388
- FormatByName
 - VcBoxFormatCollection 493
 - VcLineFormatCollection 1139
 - VcTableFormatCollection 1389
- GetActualExtent
 - VcBox 474
- GetActualScaleValues
 - VcHistogram 1030
- GetAValueFromARGB
 - VcGantt 797
- GetBValueFromARGB
 - VcGantt 797
- GetCurrentComponentStart
 - VcGantt 798
- GetCurrentViewDates
 - VcGantt 799
- GetCurrentYValues
 - VcHistogram 1030
- GetDate
 - VcGantt 800
- GetDateAsString
 - VcGantt 800
- GetEndOfPreviousWorktime
 - VcCalendar 510

- GetEnumerator
 - VcBoxCollection 482
 - VcBoxFormat 488
 - VcBoxFormatCollection 494
 - VcCalendarCollection 518
 - VcCalendarGridCollection 542
 - VcCurveCollection 592
 - VcDataDefinitionTable 604
 - VcDataRecordCollection 617
 - VcDataTableCollection 629
 - VcDataTableFieldCollection 642
 - VcDateLineCollection 661
 - VcDateLineGridCollection 680
 - VcFilter 688
 - VcFilterCollection 694
 - VcFilterSubCondition 700
 - VcGroupCollection 983
 - VcGroupLevelLayoutCollection 1011
 - VcHistogramCollection 1035
 - VcLayerCollection 1111
 - VcLayerFormat 1115
 - VcLinkAppearanceCollection 1172
 - VcLinkCollection 1177
 - VcMapCollection 1190
 - VcNodeCollection 1217
 - VcNumericScaleCollection 1246
 - VcTableCollection 1378
 - VcTableFormat 1386
 - VcTableFormatCollection 1389
 - VcTimeScaleCollection 1415
 - VcUpdateBehaviorCollection 1424
- GetFirstOverload
 - VcCurve 579
- GetFirstOverloadEx
 - VcCurve 581
- GetGValueFromARGB
 - VcGantt 801
- GetLinkById
 - VcGantt 802
- GetLinkByNodeIDs
 - VcGantt 802
- GetMapEntry
 - VcMap 1185
- GetNewUniqueID
 - VcDataRecordCollection 618
- GetNextIntervalBorder
 - VcCalendar 510
- GetNextOverload
 - VcCurve 582
- GetNextOverloadEx
 - VcCurve 583
- GetNodeById
 - VcGantt 803
- GetPositionInView
 - VcNode 1211
- GetPreviousIntervalBorder
 - VcCalendar 511
- GetRValueFromARGB
 - VcGantt 803
- GetStartOfInterval
 - VcCalendar 511
- GetStartOfNextWorktime
 - VcCalendar 512
- GetTopLeftPixel
 - VcBox 475
- GetValues
 - VcCurve 584
- GetValuesEx
 - VcCurve 585
- GetViewComponentSize
 - VcGantt 804
- GetXyOffset
 - VcBox 475

- GroupByName
 - VcGroupCollection 984
- GroupLevelLayoutByIndex
 - VcGroupLevelLayoutCollection 1012
- GroupLevelLayoutByName
 - VcGroupLevelLayoutCollection 1012
- GroupNodes
 - VcGantt 805
- HistogramByIndex
 - VcHistogramCollection 1035
- HistogramByName
 - VcHistogramCollection 1035
- IdentifyField
 - VcGantt 806
- IdentifyFormatField
 - VcBox 475
 - VcTable 1376
- IdentifyInterval
 - VcCalendarGrid 536
- IdentifyLayerAt
 - VcGantt 807
- IdentifyObject
 - VcDataRecord 610
 - VcGantt 810
- IdentifyObjectAt
 - VcGantt 811
- ImportConfiguration
 - VcGantt 813
- InitializeForWebService
 - VcGantt 813
- InsertLinkRecord
 - VcGantt 813
- InsertNodeRecord
 - VcGantt 814
- IntervalByIndex
 - VcIntervalCollection 1063
- IntervalByName
 - VcIntervalCollection 1063
- IsValid
 - VcFilter 688
 - VcFilterSubCondition 700
- IsWorktime
 - VcCalendar 512
- LayerByIndex
 - VcLayerCollection 1111
- LayerByName
 - VcLayerCollection 1111
- LinkAppearanceByIndex
 - VcLinkAppearanceCollection 1172
- LinkAppearanceByName
 - VcLinkAppearanceCollection 1173
- Load
 - VcGantt 815
- MakeARGB
 - VcGantt 816
- MapByIndex
 - VcMapCollection 1190
- MapByName
 - VcMapCollection 1190
- NextBox
 - VcBoxCollection 483
- NextCalendar
 - VcCalendarCollection 519
- NextCalendarGrid
 - VcCalendarGridCollection 543
- NextCalendarProfile
 - VcCalendarProfileCollection 551
- NextCurve
 - VcCurveCollection 593
- NextDataDefinitionField
 - VcDataDefinitionTable 604
- NextDataRecord

- VcDataRecordCollection 618
- NextDataTable
 - VcDataTableCollection 629
- NextDataTableField
 - VcDataTableFieldCollection 642
- NextDateLine
 - VcDateLineCollection 661
- NextDateLineGrid
 - VcDateLineGridCollection 681
- NextFilter
 - VcFilterCollection 694
- NextFormat
 - VcBoxFormatCollection 494
 - VcLineFormatCollection 1140
 - VcTableFormatCollection 1390
- NextGroup
 - VcGroupCollection 984
- NextGroupLevelLayout
 - VcGroupLevelLayoutCollection 1012
- NextHistogram
 - VcHistogramCollection 1036
- NextInterval
 - VcIntervalCollection 1063
- NextLayer
 - VcLayerCollection 1112
- NextLink
 - VcLinkCollection 1177
- NextLinkAppearance
 - VcLinkAppearanceCollection 1173
- NextMap
 - VcMapCollection 1191
- NextMapEntry
 - VcMap 1185
- NextNode
 - VcNodeCollection 1217
- NextNumericScale
 - VcNumericScaleCollection 1246
- NextTable
 - VcTableCollection 1378
- NextTimeScale
 - VcTimeScaleCollection 1415
- NextUpdateBehavior
 - VcUpdateBehaviorCollection 1424
- NodeRowInView
 - VcNode 1212
- NumericScaleByIndex
 - VcNumericScaleCollection 1247
- NumericScaleByName
 - VcNumericScaleCollection 1247
- OptimizeColumnWidth
 - VcTable 1376
- OptimizeTimeScaleStartEnd
 - VcGantt 816
- OutlineIndent
 - VcNode 1212
- OutlineOutdent
 - VcNode 1213
- PrintEx
 - VcGantt 817
- PrintToFile
 - VcGantt 818
- Process
 - VcResourceScheduler2 1342
- PutInOrderAfter
 - VcCalendarProfile 547
 - VcDateLine 654
 - VcHistogram 1030
 - VcInterval 1058
 - VcLayer 1106
 - VcLinkAppearance 1168
 - VcUpdateBehavior 1419
- RecalculateAllStructureCodes
 - VcGantt 819

- RelatedDataRecord
 - VcDataRecord 611
 - VcGroup 980
 - VcLink 1158
 - VcNode 1214
- Remove
 - VcBoxCollection 483
 - VcBoxFormatCollection 495
 - VcCalendarCollection 519
 - VcCalendarGridCollection 543
 - VcCalendarProfileCollection 551
 - VcCurveCollection 594
 - VcDataRecordCollection 619
 - VcDateLineCollection 662
 - VcDateLineGridCollection 681
 - VcFilterCollection 695
 - VcGroupLevelLayoutCollection 1013
 - VcIntervalCollection 1064
 - VcLayerCollection 1112
 - VcLineFormatCollection 1140
 - VcLinkAppearanceCollection 1174
 - VcMapCollection 1192
 - VcUpdateBehaviorCollection 1425
- RemoveFormatField
 - VcBoxFormat 489
 - VcLayerFormat 1116
 - VcLineFormat 1135
- RemoveSubCondition
 - VcFilter 689
- ReOptimizeNodes
 - VcGroup 980
- Reset
 - VcGantt 819
- SaveAsEx
 - VcGantt 820
- ScheduleProject
 - VcGantt 820
 - VcScheduler 1364
- ScrollComponentStartTo
 - VcGantt 821
- ScrollToDate
 - VcGantt 822
- ScrollToGroupLine
 - VcGantt 822
- ScrollToNode
 - VcGantt 823
- ScrollToNodeLine
 - VcGantt 824
- ScrollToValue
 - VcHistogram 1031
- SelectCalendarProfiles
 - VcCalendarProfileCollection 552
- SelectGroups
 - VcGroupCollection 985
- SelectLinks
 - VcLinkCollection 1178
- SelectMaps
 - VcMapCollection 1192
- SelectNodes
 - VcNodeCollection 1218
- SetImageResource
 - VcGantt 825
- SetPositionInView
 - VcNode 1214
- SetValues
 - VcCurve 586
- SetXYOffset
 - VcBox 476
- SetXYOffsetByTopLeftPixel
 - VcBox 476
- ShowAboutDialog
 - VcGantt 826
- ShowEditGroupDialog

- VcGantt 827
- ShowExportGraphicsDialog
 - VcGantt 827
- ShowLinkEditDialog
 - VcGantt 828
- ShowNodeEditDialog
 - VcGantt 829
- ShowPageSetupDialog
 - VcGantt 830
- ShowPrintDialog
 - VcGantt 830
- ShowPrinterSetupDialog
 - VcGantt 830
- ShowPrintPreviewDialog
 - VcGantt 831
- SortGroups
 - VcGantt 831
- SortNodes
 - VcGantt 831
- SuspendUpdate
 - VcGantt 832
- TableByIndex
 - VcTableCollection 1379
- TableByName
 - VcTableCollection 1379
- TimeScaleByIndex
 - VcTimeScaleCollection 1416
- TimeScaleByName
 - VcTimeScaleCollection 1416
- Update
 - VcBoxCollection 484
 - VcCalendar 513
 - VcCalendarCollection 520
 - VcCalendarGridCollection 544
 - VcCalendarProfileCollection 552
 - VcDataRecord 611
 - VcDataRecordCollection 620
 - VcDataTableCollection 630
 - VcDateLineCollection 662
 - VcDateLineGridCollection 682
 - VcGroup 980
 - VcGroupLevelLayoutCollection 1013
 - VcIntervalCollection 1064
 - VcLayerCollection 1113
 - VcLegendView 1132
 - VcLink 1158
 - VcLinkAppearanceCollection 1174
 - VcMapCollection 1193
 - VcNode 1215
 - UpdateBehaviorByIndex
 - VcUpdateBehaviorCollection 1425
 - UpdateBehaviorByName
 - VcUpdateBehaviorCollection 1426
 - UpdateLinkRecord
 - VcGantt 834
 - UpdateNodeRecord
 - VcGantt 834
 - UpdateRowNumberFields
 - VcGantt 835
 - Zoom
 - VcGantt 835
- Millimeter**
 - Property of
 - VcMapEntry 1199
- MinimumRowHeight**
 - Property of
 - VcGantt 748
- MinimumStartDataFieldIndex**
 - Property of
 - VcLayer 1081
- MinimumTextLineCount**
 - Property of
 - VcBoxFormatField 500

VcTableFormatField 1396

MinimumWidth

Property of

VcBoxFormatField 501

VcLayerFormatField 1119

MinorTicks

Property of

VcNumericScale 1234

VcRibbon 1348

MinorTicksEx

Property of

VcNumericScale 1234

Mode

create link 416

create node 415, 417

panning mode 415

pointer mode 415

Property of

VcWorldView 1435

ModificationsAllowed

Property of

VcGroupLevelLayout 993

Modifying the scaling and the frontiers of sections 405

MouseProcessingEnabled

Property of

VcGantt 749

Movable

Property of

VcDateLine 650

VcLayer 1082

Move layers

with shift key 241

Moveable

Property of

VcBox 468

MoveMode

Property of

VcGantt 749

Moving

nodes or layers 383

MovingGroupsVerticallyViaDiagramAllowed

Property of

VcGroupLevelLayout 994

MovingGroupsVerticallyViaTableAllowed

Property of

VcGroupLevelLayout 994

MovingLayersAsNodeWithShiftKeyAllowed

Property of

VcGantt 750

MultipleBoxMarkingAllowed

Property of

VcGantt 750

MultiplePrimaryKeysAllowed

Property of

VcDataTable 624

MultiState

Property of

VcTableFormatField 1396

MultiState fields 183

N

Name

Property of

VcBox 469

VcBoxFormat 487

VcCalendar 506

VcCalendarGrid 528

VcCalendarProfile 545

VcCurve 572

VcDataDefinitionField 598

- VcDataTable 624
- VcDataTableField 634
- VcDateLine 650
- VcFilter 684
- VcGroup 975
- VcGroupLevelLayout 994
- VcHistogram 1023
- VcInterval 1051
- VcLayer 1082
- VcLineFormat 1134
- VcLinkAppearance 1164
- VcMap 1181
- VcNumericScale 1235
- VcTable 1374
- VcTableFormat 1384
- VcTimeScale 1409
- VcUpdateBehavior 1418
- Navigation**
 - Keyboard 381
- NextBox**
 - Method of
 - VcBoxCollection 483
- NextCalendar**
 - Method of
 - VcCalendarCollection 519
- NextCalendarGrid**
 - Method of
 - VcCalendarGridCollection 543
- NextCalendarProfile**
 - Method of
 - VcCalendarProfileCollection 551
- NextCurve**
 - Method of
 - VcCurveCollection 593
- NextDataDefinitionField**
 - Method of
 - VcDataDefinitionTable 604
- NextDataRecord**
 - Method of
 - VcDataRecordCollection 618
- NextDataTable**
 - Method of
 - VcDataTableCollection 629
- NextDataTableField**
 - Method of
 - VcDataTableFieldCollection 642
- NextDateLine**
 - Method of
 - VcDateLineCollection 661
- NextDateLineGrid**
 - Method of
 - VcDateLineGridCollection 681
- NextFilter**
 - Method of
 - VcFilterCollection 694
- NextFormat**
 - Method of
 - VcBoxFormatCollection 494
 - VcLineFormatCollection 1140
 - VcTableFormatCollection 1390
- NextGroup**
 - Method of
 - VcGroupCollection 984
- NextGroupLevelLayout**
 - Method of
 - VcGroupLevelLayoutCollection 1012
- NextHistogram**
 - Method of
 - VcHistogramCollection 1036
- NextInterval**
 - Method of
 - VcIntervalCollection 1063
- NextLayer**

- Method of
 - VcLayerCollection 1112
- NextLink**
 - Method of
 - VcLinkCollection 1177
- NextLinkAppearance**
 - Method of
 - VcLinkAppearanceCollection 1173
- NextMap**
 - Method of
 - VcMapCollection 1191
- NextMapEntry**
 - Method of
 - VcMap 1185
- NextNode**
 - Method of
 - VcNodeCollection 1217
- NextNumericScale**
 - Method of
 - VcNumericScaleCollection 1246
- NextTable**
 - Method of
 - VcTableCollection 1378
- NextTimeScale**
 - Method of
 - VcTimeScaleCollection 1415
- NextUpdateBehavior**
 - Method of
 - VcUpdateBehaviorCollection 1424
- Node 185**
 - allow new nodes 240
 - assign calendars to nodes 240
 - creating 185
 - delete, cut, copy, paste 390
 - deleting 185, 420
 - edit new node 239
 - ID 1206
 - information window 735
 - interactive regrouping 117
 - marking type in diagram 239
 - marking type in table 238
 - move node when marked 241
 - moving 383, 385
 - moving all selected nodes 240, 241
 - new nodes via double click 240
 - see also
 - VcNode 1204
- node order**
 - change in diagram 242
 - change in table 242
- Node rows**
 - initial number 247
 - minimal height 247
- NodeCalendarNameDataFieldIndex**
 - Property of
 - VcGantt 751
- NodeCollection**
 - Property of
 - VcGantt 751
 - VcGroup 976
 - see also
 - VcNodeCollection 1216
- NodeCreationAllowed**
 - Property of
 - VcGantt 752
- NodeCreationAtDroppingEnabled**
 - Property of
 - VcGantt 752
- NodeCreationViaDoubleClick**
 - Property of
 - VcGantt 753
- NodeCreationWithDialog**
 - Property of
 - VcGantt 753

NodeDurationDataFieldIndex

Property of
VcGantt 753

NodeEndDateDataFieldIndex

Property of
VcGantt 754

NodeID

Property of
VcBox 469

NodeLevelLayout

Property of
VcGantt 754
see also
VcNodeLevelLayout 1220

NodeRowInView

Method of
VcNode 1212

NodeRowNumberDataFieldIndex

Property of
VcGantt 754

Nodes

all nodes of one group of this level in
one line 995
data record 979, 1157, 1210
delete, cut, copy, paste 394
groups in one line 289, 295
interactive generation 441
move all layers with shift key 750
move vertically 785, 786
moving 387
node layout optimized 995, 1017
optimized arrangement 731
related data record 980, 1158, 1214
Selecting by rubber rectangle 760,
769

NodesAndGroupsBelowCollapsed

Property of

VcGroup 976

NodesDataTableName

Property of
VcGantt 755

NodeSeparationLinesVisible

Property of
VcHierarchyLevelLayout 1016

NodesInHeader

Property of
VcGroup 977

NodesInHeaders

Property of
VcGroupLevelLayout 995
VcHierarchyLevelLayout 1017

NodeSortingDataFieldIndex

Property of
VcGantt 756

NodeSortingOrder

Property of
VcGantt 757

NodesOverlaid

Property of
VcGroup 977
VcGroupLevelLayout 995
VcHierarchyLevelLayout 1017

NodeStartDateDataFieldIndex

Property of
VcGantt 757

NodesUseCalendars

Property of
VcGantt 757

NodeToolTipTextDataFieldIndex

Property of
VcGantt 758

NominalScaleMaximum

Property of
VcHistogram 1023

NominalScaleMinimum

Property of
VcHistogram 1024

Non work interval

Date line grid 1084

Non Working Intervals

mark 37

NonWorkIntervalBackgroundColor

Property of
VcLayer 1083

NonWorkIntervalBackgroundColorDataFieldIndex

Property of
VcLayer 1083

NonWorkIntervalBackgroundColorMapName

Property of
VcLayer 1083

NonWorkIntervalLineColor

Property of
VcLayer 1084

NonWorkIntervalLineColorDataFieldIndex

Property of
VcLayer 1084

NonWorkIntervalLineColorMapName

Property of
VcLayer 1084

NonWorkIntervalLineThickness

Property of
VcLayer 1085

NonWorkIntervalLineType

Property of
VcLayer 1085

NonWorkIntervalPattern

Property of
VcLayer 1086

NonWorkIntervalPatternColor

Property of
VcLayer 1089

NonWorkIntervalPatternColorDataFieldIndex

Property of
VcLayer 1089

NonWorkIntervalPatternColorMapName

Property of
VcLayer 1090

NonWorkIntervalPatternDataFieldIndex

Property of
VcLayer 1090

NonWorkIntervalPatternMapName

Property of
VcLayer 1090

NonWorkIntervalsCollapsed

Property of
VcSection 1368

NonWorkIntervalShape

Property of
VcLayer 1091

NoOfColumns

Property of
VcTable 1374

Number

Property of
VcMapEntry 1200

Numeric scale

3D effect 1239
background color of pattern 1235
by index 1247
font body 1230
font color 1231
histogram associated 1231
major ticks 1232, 1233
maximum value 1023

minimum value 1024

minor ticks 1234

name 1235

Pattern 1236

pattern color 1236

rescale 230

title 1240

unit 1241

unit label 1241

unit width 1242

NumericScale

see also

VcNumericScale 1229

NumericScaleByIndex

Method of

VcNumericScaleCollection 1247

NumericScaleByName

Method of

VcNumericScaleCollection 1247

NumericScaleCollection

Property of

VcGantt 758

VcHistogram 1024

see also

VcNumericScaleCollection 1244

NumericScaleRescalingAllowed

Property of

VcGantt 759

O

ObjectDraw events

enabled 1091

ObjectDrawEventsEnabled

Property of

VcLayer 1091

Objects

VcBorderArea 452

VcBoundingBox 454

VcBox 462

VcBoxCollection 478

VcBoxFormat 485

VcBoxFormatCollection 490

VcBoxFormatField 497

VcCalendar 505

VcCalendarCollection 514

VcCalendarGrid 521

VcCalendarGridCollection 538

VcCalendarProfile 545

VcCalendarProfileCollection 548

VcCurve 554

VcCurveCollection 588

VcDataDefinitionField 595

VcDataDefinitionTable 600

VcDataRecord 606

VcDataRecordCollection 613

VcDataTable 622

VcDataTableCollection 625

VcDataTableField 631

VcDataTableFieldCollection 638

VcDateLine 644

VcDateLineCollection 656

VcDateLineGrid 664

VcDateLineGridCollection 676

VcFilter 683

VcFilterCollection 690

VcFilterSubCondition 696

VcGantt 701

VcGroup 971

VcGroupCollection 982

VcGroupLevelLayout 986

VcGroupLevelLayoutCollection 1009

VcHierarchyLevelLayout 1014

VcHistogram 1021

VcHistogramCollection 1032

- VcInfoWindow 1037
- VcInterval 1044
- VcIntervalCollection 1060
- VcLayer 1065
- VcLayerCollection 1108
- VcLayerFormat 1114
- VcLayerFormatField 1117
- VcLegendView 1125
- VcLineFormat 1133
- VcLineFormatCollection 1136
- VcLineFormatField 1142
- VcLink 1154
- VcLinkAppearance 1160
- VcLinkAppearanceCollection 1169
- VcLinkCollection 1176
- VcMap 1180
- VcMapCollection 1187
- VcMapEntry 1194
- VcNode 1204
- VcNodeCollection 1216
- VcNodeLevelLayout 1220
- VcNumericScale 1229
- VcNumericScaleCollection 1244
- VcPrinter 1249
- VcRect 1274
- VcResourceScheduler2 1278
- VcRibbon 1344
- VcScheduler 1358
- VcSection 1366
- VcTable 1372
- VcTableCollection 1377
- VcTableFormat 1380
- VcTableFormatCollection 1387
- VcTableFormatField 1391
- VcTimeScale 1407
- VcTimeScaleCollection 1413
- VcUpdateBehavior 1417
- VcUpdateBehaviorCollection 1420
- VcUpdateBehaviorContext 1427
- VcWorldView 1431
- ObserveDST**
 - Property of
 - VcDateLineGrid 670
 - VcRibbon 1349
- OLE-DragDrop**
 - Enabled in diagram 759
 - Enabled in table 759
- OLEDragViaDiagram**
 - Property of
 - VcGantt 759
- OLEDragViaTable**
 - Property of
 - VcGantt 759
- OperationDataTableName**
 - Property of
 - VcResourceScheduler2 1298
- OperationLoadPerItemFieldIndex**
 - Property of
 - VcResourceScheduler2 1300
- OperationMaximumInterruptionTimeFieldIndex**
 - Property of
 - VcResourceScheduler2 1300
- OperationMinimumSupplementTimeFieldIndex**
 - Property of
 - VcResourceScheduler2 1301
- OperationOverlapQuantityFieldIndex**
 - Property of
 - VcResourceScheduler2 1302
- OperationPostLoadFieldIndex**
 - Property of
 - VcResourceScheduler2 1304
- OperationPostOffsetFieldIndex**

- Property of
 - VcResourceScheduler2 1304
- OperationPreparationLoadFieldIndex**
 - Property of
 - VcResourceScheduler2 1305
- OperationPreparationOffsetFieldIndex**
 - Property of
 - VcResourceScheduler2 1306
- OperationResultEndDateFieldIndex**
 - Property of
 - VcResourceScheduler2 1306
- OperationResultPostEndDateFieldIndex**
 - Property of
 - VcResourceScheduler2 1307
- OperationResultPreparationStartDateFieldIndex**
 - Property of
 - VcResourceScheduler2 1308
- OperationResultProcessingTimeFieldIndex**
 - Property of
 - VcResourceScheduler2 1308
- OperationResultSelectedTimingResourceIDFieldIndex**
 - Property of
 - VcResourceScheduler2 1309
- OperationResultStartDateFieldIndex**
 - Property of
 - VcResourceScheduler2 1309
- OperationResultStatusFieldIndex**
 - Property of
 - VcResourceScheduler2 1310
- OperationRouteFieldIndex**
 - Property of
 - VcResourceScheduler2 1311
- OperationSequenceNumberFieldIndex**
 - Property of
 - VcResourceScheduler2 1311
- OperationStartLockDateFieldIndex**
 - Property of
 - VcResourceScheduler2 1312
- OperationTaskIDFieldIndex**
 - Property of
 - VcResourceScheduler2 1313
- OperationWorkInProgressFieldIndex**
 - Property of
 - VcResourceScheduler2 1313
- Operator**
 - Property of
 - VcFilterSubCondition 699
- OptimizeColumnWidth**
 - Method of
 - VcTable 1376
- OptimizedNodesSortDataFieldIndex**
 - Property of
 - VcGroupLevelLayout 995
- OptimizedNodesSortOrder**
 - Property of
 - VcGroupLevelLayout 996
- OptimizeTimeScaleStartEnd**
 - Method of
 - VcGantt 816
- Orientation**
 - Property of
 - VcPrinter 1263
- Origin**
 - Property of
 - VcBox 470
- OutgoingLinks**
 - Property of
 - VcNode 1208
- OutlineIndent**
 - Method of
 - VcNode 1212

OutlineOutdent

Method of
VcNode 1213

Output

fitting to page count 408
print diagram 408, 409
start date for the time range to be
printed 409
zoom factor 408
zoom with horizontal fitting 408

OutputFormatForCenterDate

Property of
VcInfoWindow 1037

OutputFormatForDuration

Property of
VcInfoWindow 1039

OutputFormatForEndDate

Property of
VcInfoWindow 1040

OutputFormatForStartDate

Property of
VcInfoWindow 1041

OverlaidNodesSortDataFieldIndex

Property of
VcGroupLevelLayout 996

OverlaidNodesSortOrder

Property of
VcGroupLevelLayout 996

OverlapLayerEnabled

Property of
VcGantt 760

OverlapLayerName

Property of
VcGantt 760

Overload data

Calendar for intervals 572

OverloadResultsCalendarName

Property of
VcCurve 572

P**Page numbers 411****Page setup 417****PagebreakMode**

Property of
VcGroupLevelLayout 997
VcHierarchyLevelLayout 1017

PageDescription

Property of
VcPrinter 1264

PageDescriptionString

Property of
VcPrinter 1264

PageFrame

Property of
VcPrinter 1265

PageNumberMode

Property of
VcPrinter 1265

PageNumbers

Property of
VcPrinter 1266

PagePaddingEnabled

Property of
VcPrinter 1266

Panning mode 415**PanningModeAllowed**

Property of
VcGantt 760

PaperSize

Property of
VcPrinter 1267

PartialLoadThreshold

Property of

VcGantt 761

Pattern

Property of

VcCalendarGrid 528

VcInterval 1051

VcLayer 1091

VcMapEntry 1200

PatternBackgroundColor

Property of

VcBoxFormatField 501

PatternBackgroundColorAsARGB

Property of

VcLineFormatField 1145

VcNumericScale 1235

VcRibbon 1349

VcTableFormatField 1397

PatternBackgroundColorDataFieldIndex

Property of

VcLineFormatField 1146

VcTableFormatField 1397

PatternBackgroundColorMapName

Property of

VcLineFormatField 1146

VcTableFormatField 1397

PatternColor

Property of

VcCalendarGrid 531

VcInterval 1055

VcLayer 1094

PatternColorAsARGB

Property of

VcBoxFormatField 502

VcLineFormatField 1146

VcNumericScale 1235

VcRibbon 1350

VcTableFormatField 1398

PatternColorDataFieldIndex

Property of

VcCalendarGrid 532

VcLayer 1095

VcTableFormatField 1398

PatternColorMapName

Property of

VcCalendarGrid 532

VcLayer 1095

VcLineFormatField 1147

VcTableFormatField 1398

PatternDataFieldIndex

Property of

VcCalendarGrid 533

VcLayer 1096

PatternEx

Property of

VcLineFormatField 1147

VcNumericScale 1236

VcRibbon 1350

VcTableFormatField 1399

PatternExDataFieldIndex

Property of

VcLineFormatField 1150

VcTableFormatField 1402

PatternExMapName

Property of

VcLineFormatField 1151

VcTableFormatField 1402

PatternMapName

Property of

VcCalendarGrid 533

VcLayer 1097

PDF Files

Export 221

performance 443

Period

- Property of
 - VcDateLineGrid 670
- PhantomDrawingWhileDraggingEnabled**
 - Property of
 - VcGantt 762
- PhantomLayerHeight**
 - Property of
 - VcGantt 763
- PlanningEndDate**
 - Property of
 - VcResourceScheduler2 1314
- PlanningStartDate**
 - Property of
 - VcResourceScheduler2 1315
- PlanningStrategy**
 - Property of
 - VcResourceScheduler2 1316
- Platforms x86 and x64 187**
- Pointer mode 415**
- PointsEquidistant**
 - Property of
 - VcCurve 573
- Position**
 - Property of
 - VcRibbon 1354
 - VcTable 1374
- PredecessorLayerName**
 - Property of
 - VcLinkAppearance 1164
- PredecessorNode**
 - Property of
 - VcLink 1156
- PredecessorPortSymbol**
 - Property of
 - VcLinkAppearance 1165
- Primary key**
 - composite 624
- PrimaryKey**
 - Property of
 - VcDataTableField 635
- Print Preview 412**
- PrintDate**
 - Property of
 - VcPrinter 1267
- Printer**
 - Property of
 - VcGantt 763
 - see also
 - VcPrinter 1249
- PrinterName**
 - Property of
 - VcPrinter 1268
- PrintEx**
 - Method of
 - VcGantt 817
- Printing 68, 418**
 - absolute height of the bottom margin in cm 1273
 - absolute height of the bottom margin in inches 1250
 - absolute height of the top margin in cm 1253
 - absolute height of the top margin in inches 1253
 - absolute width of the lefthand margin in cm 1251
 - absolute width of the lefthand margin in inches 1251
 - absolute width of the righthand margin in cm 1252
 - absolute width of the righthand margin in inches 1252
 - actual zoom factor 1256
 - all controls at once 1255

- automatic re-optimization of groups 1269
- current printer 1258
- date format in Page Layout dialog 1258
- diagram 1259
- end date of time range to be printed 1271
- folding marks 1261
- into file 818
- number of table columns 1270
- print date 411
- print preview 417
- printer setup 417
- scaling mode 1269
- start date of time range to be printed 1272
- Table columns 409
- table width 1271
- table width as on screen 408

PrintPreviewWithFirstPage

- Property of
 - VcPrinter 1268

PrintToFile

- Method of
 - VcGantt 818

Priority

- boxes 311
- Property of
 - VcBox 470
 - VcCalendarGrid 533
 - VcDateLine 651
 - VcDateLineGrid 671
 - VcLayerFormatField 1120

Process

- Method of
 - VcResourceScheduler2 1342

Project end 224**Project start 224****Properties**

- AbsoluteBottomMarginInInches
 - VcPrinter 1250
- AbsoluteLeftMarginInCM
 - VcPrinter 1251
- AbsoluteLeftMarginInInches
 - VcPrinter 1251
- AbsoluteRightMarginInCM
 - VcPrinter 1252
- AbsoluteRightMarginInInches
 - VcPrinter 1252
- AbsoluteTopMarginInCM
 - VcPrinter 1253
- AbsoluteTopMarginInInches
 - VcPrinter 1253
- Active
 - VcCalendarCollection 514
 - VcHistogramCollection 1032
 - VcNumericScaleCollection 1244
 - VcTableCollection 1377
 - VcTimeScaleCollection 1413
 - VcUpdateBehaviorCollection 1420
- ActiveNodeFilter
 - VcGantt 710
- ActualEndDateDataFieldIndex
 - VcScheduler 1359
- ActualStartDateDataFieldIndex
 - VcScheduler 1359
- Addend
 - VcCurve 555
- AdjustToReferenceDate
 - VcDateLineGrid 665
- Alignment
 - VcBoundingBox 454
 - VcBoxFormatField 497
 - VcLayerFormatField 1118

- VcLineFormatField 1143
- VcPrinter 1254
- VcTableFormatField 1392
- AllBorderBoxesShownOnCombinedControls
 - VcPrinter 1254
- AllData
 - VcDataRecord 606
 - VcLink 1154
 - VcNode 1205
- AllLayersMovingTogether
 - VcGantt 710
- AllLayersMovingTogetherAlways
 - VcGantt 711
- AlwaysCurrentDate
 - VcDateLine 645
- AnchoringInteractionsAllowed
 - VcBox 463
- AnchoringLineVisible
 - VcBox 463
- AnnotationAtBottom
 - VcDateLineGrid 665
- AnnotationAtCenter
 - VcDateLineGrid 665
- AnnotationAtTop
 - VcDateLineGrid 666
- Arrangement
 - VcGantt 711
- ArrowKeyMode
 - VcGantt 712
- ArrowKeyStepSizeMultiplier
 - VcGantt 713
- AssignmentDataTableName
 - VcResourceScheduler2 1281
- AssignmentIsResultFieldIndex
 - VcResourceScheduler2 1283
- AssignmentIsVisibleFieldIndex
 - VcResourceScheduler2 1283
- AssignmentLoadOrConsumptionPerItemFieldIndex
 - VcResourceScheduler2 1284
- AssignmentMaximumLoadFieldIndex
 - VcResourceScheduler2 1285
- AssignmentMinimumLoadFieldIndex
 - VcResourceScheduler2 1285
- AssignmentMinimumMaximumLoadType
 - VcResourceScheduler2 1286
- AssignmentOperationIDFieldIndex
 - VcResourceScheduler2 1287
- AssignmentResourceIDFieldIndex
 - VcResourceScheduler2 1287
- AssignmentResourceSelectionStrategyFieldIndex
 - VcResourceScheduler2 1288
- AutoCollapseGroups
 - VcGroupLevelLayout 987
 - VcHierarchyLevelLayout 1014
- AutoExpandTargetGroup
 - VcGroupLevelLayout 988
 - VcHierarchyLevelLayout 1015
- AutomaticSchedulingEnabled
 - VcScheduler 1359
- BackgroundColor
 - VcCalendarGrid 522
 - VcInterval 1046
 - VcLayer 1067
 - VcTimeScale 1407
- BackgroundColorDataFieldIndex
 - VcCalendarGrid 523
 - VcLayer 1067
- BackgroundColorMapName
 - VcCalendarGrid 523
 - VcLayer 1069

- BaseCalendarUsageForSupplementTimes
 - VcResourceScheduler2 1289
- BaseTimeUnit
 - VcResourceScheduler2 1290
- BaseTimeUnitsPerStep
 - VcResourceScheduler2 1290
- BodiesCollapsed
 - VcGroupLevelLayout 988
 - VcHierarchyLevelLayout 1015
- BodiesCollapsedDataFieldIndex
 - VcGroupLevelLayout 988
 - VcHierarchyLevelLayout 1015
- BodiesCollapsedMapName
 - VcGroupLevelLayout 989
 - VcHierarchyLevelLayout 1016
- BodyCollapsed
 - VcGroup 972
- Border
 - VcLegendView 1125
 - VcWorldView 1431
- BorderArea
 - VcGantt 714
- BorderColor
 - VcLegendView 1126
 - VcWorldView 1432
- Bottom
 - VcRect 1274
- BottomMargin
 - VcLayerFormatField 1118
 - VcTableFormatField 1392
- BoxCollection
 - VcGantt 715
- BoxCreationAllowed
 - VcGantt 715
- BoxFormatCollection
 - VcGantt 715
- CalendarCollection
 - VcGantt 716
- CalendarGrid
 - VcSection 1366
- CalendarGridCollection
 - VcGantt 716
- CalendarGridName
 - VcGroupLevelLayout 989
 - VcNodeLevelLayout 1221
- CalendarGridsVisible
 - VcGroupLevelLayout 989
 - VcHistogram 1022
 - VcNodeLevelLayout 1221
 - VcTimeScale 1408
- CalendarGridsWithChildGroups
 - VcGroupLevelLayout 989
- CalendarName
 - VcCalendarGrid 523
 - VcHistogram 1022
 - VcRibbon 1345
- CalendarNameDataFieldIndex
 - VcCalendarGrid 524
 - VcGroupLevelLayout 990
- CalendarNameMapName
 - VcCalendarGrid 524
- CalendarProfileCollection
 - VcCalendar 506
 - VcGantt 717
- CalendarProfileName
 - VcInterval 1046
- CollapseColumn
 - VcTableFormat 1381
- Color
 - VcMapEntry 1194
- ColumnName
 - VcTable 1372
- ColumnWidth

- VcTable 1373
- CombiningControlsEnabled
 - VcPrinter 1255
- ComparisonValueAsString
 - VcFilterSubCondition 696
- CompletionDataFieldIndex
 - VcLayer 1070
- ConnectionOperator
 - VcFilterSubCondition 697
- ConsiderFilterEntries
 - VcMap 1180
- ConsiderLinkRelationTypesOnNodeDragging
 - VcGantt 717
- ConstantText
 - VcLayerFormatField 1118
 - VcLineFormatField 1143
 - VcTableFormatField 1393
- ContextMenuForBoxesEnabled
 - VcGantt 717
- Count
 - VcBoxCollection 478
 - VcBoxFormatCollection 490
 - VcCalendarCollection 515
 - VcCalendarGridCollection 538
 - VcCalendarProfileCollection 549
 - VcCurveCollection 588
 - VcDataDefinitionTable 600
 - VcDataRecordCollection 613
 - VcDataTableCollection 625
 - VcDataTableFieldCollection 638
 - VcDateLineCollection 656
 - VcDateLineGridCollection 676
 - VcFilterCollection 690
 - VcGroupCollection 982
 - VcGroupLevelLayoutCollection 1009
 - VcHistogramCollection 1033
 - VcIntervalCollection 1061
 - VcLayerCollection 1108
 - VcLineFormatCollection 1136
 - VcLinkAppearanceCollection 1169
 - VcLinkCollection 1176
 - VcMap 1181
 - VcMapCollection 1187
 - VcNodeCollection 1216
 - VcNumericScaleCollection 1245
 - VcTableCollection 1378
 - VcTableFormatCollection 1387
 - VcTimeScaleCollection 1414
 - VcUpdateBehaviorCollection 1421
- CtrlCXVProcessingEnabled
 - VcGantt 718
- CurrentHorizontalPagesCount
 - VcPrinter 1255
- CurrentVerticalPagesCount
 - VcPrinter 1256
- CurrentZoomFactor
 - VcPrinter 1256
- CurveCollection
 - VcHistogram 1022
- CuttingMarks
 - VcPrinter 1256
- DataDefinition
 - VcGantt 718
- DataDefinitionTable
 - VcFilter 684
- DataField
 - VcDataRecord 607
 - VcGroup 972
 - VcLink 1155
 - VcNode 1205
- DataFieldIndex
 - VcFilterSubCondition 697

- DataFieldValue
 - VcMapEntry 1195
- DataRecordCollection
 - VcDataTable 622
- DataRecordEventsEnabled
 - VcResourceScheduler2 1291
- DataTableCollection
 - VcGantt 719
- DataTableFieldCollection
 - VcDataTable 623
- DataTableName
 - VcDataRecord 608
 - VcDataTableField 631
- Date
 - VcDateLine 645
- DateDataFieldIndex
 - VcDateLine 646
- DateFormat
 - VcDataDefinitionField 595
 - VcDataTableField 632
 - VcPrinter 1257
- DateGridsVisible
 - VcTimeScale 1408
- DateLineCollection
 - VcGantt 719
- DateLineGrid
 - VcSection 1367
- DateLineGridName
 - VcGroupLevelLayout 990
- DateLineGridsVisible
 - VcGroupLevelLayout 990
- DateLineGridsWithChildGroups
 - VcGroupLevelLayout 991
- DateLineName
 - VcGroupLevelLayout 991
 - VcNodeLevelLayout 1221
- DateLinesVisible
 - VcGroupLevelLayout 991
 - VcNodeLevelLayout 1221
- DateLinesWithChildGroups
 - VcGroupLevelLayout 991
- DateOutputFormat
 - VcGantt 720
 - VcLineFormatField 1143
 - VcRibbon 1345
- DatesWithHourAndMinute
 - VcFilter 684
- DayInEndMonth
 - VcInterval 1046
- DayInStartMonth
 - VcInterval 1047
- DefaultOperationMaximumInterruptionTime
 - VcResourceScheduler2 1291
- DefaultPrinterName
 - VcPrinter 1258
- DefaultResourceCalendarName
 - VcResourceScheduler2 1292
- DelayTime
 - VcUpdateBehaviorContext 1427
- Description
 - VcDataTable 623
- DiagramAlternatingRowBackgroundColor
 - VcGantt 721
- DiagramBackgroundColor
 - VcGantt 722
- DiagramEnabled
 - VcPrinter 1258, 1259
- DiagramHistogramHeightRatio
 - VcGantt 722
- DiagramHistogramHeightRatioEx
 - VcGantt 723
- DiagramVisible

- VcGantt 723
- DialogFont
 - VcGantt 723
- DirectDataWritingModeEnabled
 - VcGantt 724
- DocumentName
 - VcPrinter 1259
- DoubleOutputFormat
 - VcGantt 724
 - VcNumericScale 1230
- Duration
 - VcInterval 1047
- DurationDataFieldIndex
 - VcLayer 1070
 - VcScheduler 1360
- EarlyEndDateDataFieldIndex
 - VcScheduler 1360
- EarlyStartDateDataFieldIndex
 - VcScheduler 1360
- Editable
 - VcDataDefinitionField 596
 - VcDataTableField 633
- Enabled
 - VcGantt 725
- EndDataFieldIndex
 - VcLayer 1071
- EndDateForAutomaticScheduling
 - VcGantt 725
 - VcScheduler 1360
- EndDateNotLaterThanDataFieldIndex
 - VcScheduler 1361
- EndTime
 - VcInterval 1047
- EndMonth
 - VcInterval 1048
- EndSnapTarget
 - VcCalendarGrid 524
- VcLayer 1071
- EndTime
 - VcInterval 1048
- EndWeekday
 - VcInterval 1048
- EventsSecurityCheck
 - VcGantt 726
- ExtendedDataTablesEnabled
 - VcGantt 726
- ExtendedEditingBehavior
 - VcGantt 727
- FieldsSeparatedByLines
 - VcBoxFormat 485
 - VcTableFormat 1381
- FieldText
 - VcBox 464
- FilePath
 - VcGantt 727
- FillReference1BackgroundColor
 - VcCurve 556
- FillReference1Name
 - VcCurve 556
- FillReference1Pattern
 - VcCurve 557
- FillReference1PatternColor
 - VcCurve 561
- FillReference2Color
 - VcCurve 561
- FillReference2Name
 - VcCurve 562
- FillReference2Pattern
 - VcCurve 562
- FillReference2PatternColor
 - VcCurve 566
- FilterCollection
 - VcGantt 728
- FilterName

- VcCurve 567
- VcFilterSubCondition 698
- VcLayer 1071
- VcLinkAppearance 1160
- VcTableFormat 1382
- FitToPage
 - VcPrinter 1259
- FoldingMarksType
 - VcPrinter 1260
- Font
 - VcDateLine 646
 - VcNumericScale 1230
 - VcRibbon 1347
 - VcTimeScale 1409
- FontAntiAliasingEnabled
 - VcGantt 728
- FontBody
 - VcMapEntry 1196
- FontColor
 - VcDateLine 646
 - VcNumericScale 1231
 - VcRibbon 1347
 - VcTimeScale 1409
- FontName
 - VcMapEntry 1196
- FontSize
 - VcMapEntry 1197
- Format
 - VcLayer 1072
- FormatField
 - VcBoxFormat 486
 - VcLayerFormat 1114
 - VcLineFormat 1133
 - VcTableFormat 1382
- FormatFieldCount
 - VcBoxFormat 486
 - VcLayerFormat 1115
- VcLineFormat 1134
- VcTableFormat 1383
- FormatName
 - VcBox 464
 - VcBoxFormatField 498
 - VcDateLineGrid 666
 - VcLayerFormatField 1119
 - VcLineFormatField 1145
 - VcTableFormatField 1393
- FreeFloatDataFieldIndex
 - VcScheduler 1361
- FullUsageOfPlanningUnitsEnabled
 - VcResourceScheduler2 1292
- GetEnumerator
 - VcMap 1181
- GraphicsFileName
 - VcBoundingBox 455
 - VcLayer 1072
 - VcMapEntry 1197
 - VcTableFormatField 1393
- GraphicsFileNameDataFieldIndex
 - VcLayer 1074
 - VcTableFormatField 1394
- GraphicsFileNameMapName
 - VcLayer 1075
 - VcTableFormatField 1394
- GraphicsHeight
 - VcBoxFormatField 499
 - VcTableFormatField 1395
- GroupCollection
 - VcGantt 729
- GroupDataFieldIndex
 - VcGroupLevelLayout 992
- GroupingDataFieldIndex
 - VcGantt 729
- GroupingLevel
 - VcGroup 973

- GroupingModificationsAllowed
 - VcGantt 730
- GroupInvisible
 - VcGroup 974
- GroupLevelLayoutCollection
 - VcGantt 730
- GroupNodesVisible
 - VcGroupLevelLayout 992
- GroupOptimizationOnInteractionsEnabled
 - VcGantt 731
- GroupsInvisible
 - VcGroupLevelLayout 992
- GroupsInvisibleCollapsedMapName
 - VcGroupLevelLayout 993
- GroupsInvisibleDataFieldIndex
 - VcGroupLevelLayout 993
- GroupSortingDataFieldIndex
 - VcGantt 731
- GroupSortingOrder
 - VcGantt 732
- Height
 - VcLayer 1077
 - VcLegendView 1126
 - VcRect 1274
 - VcWorldView 1432
- HeightActualValue
 - VcLegendView 1127
 - VcWorldView 1433
- HeightDataFieldIndex
 - VcLayer 1077
- HeightMapName
 - VcLayer 1078
- Hidden
 - VcDataDefinitionField 597
 - VcDataTableField 633
- HierarchyDataFieldIndex
 - VcGantt 732
 - VcHierarchyLevelLayout 1016
- HierarchyLevelLayout
 - VcGantt 733
- Histogram
 - VcCurve 567
 - VcNumericScale 1231
- HistogramCollection
 - VcGantt 733
- HistogramSeparationLineColor
 - VcGantt 734
- HorAlignment
 - VcDateLineGrid 666
- HorizontalMovementWhileDraggingAllowed
 - VcGantt 734
- HorizontalOffset
 - VcLayer 1079
- ID
 - VcDataRecord 609
 - VcGroup 974
 - VcLink 1156
 - VcNode 1206
- Identifiable
 - VcCalendarGrid 525
 - VcDateLine 647
- InbuiltMouseCursorWhileDraggingEnabled
 - VcGantt 734
- IncomingLinks
 - VcNode 1206
- IndentColumn
 - VcTableFormat 1383
- IndentWidth
 - VcTableFormat 1384
- Index
 - VcBoxFormatField 499

- VcDataDefinitionField 597
- VcDataTableField 634
- VcFilterSubCondition 698
- VcLayerFormatField 1119
- VcLineFormatField 1145
- VcTableFormatField 1395
- InfoWindow
 - VcGantt 735
- InitialRowCount
 - VcGantt 735
- InPlaceEditingOnGroupsInDiagramEnabled
 - VcGantt 735
- InPlaceEditingOnGroupsInTableEnabled
 - VcGantt 736
- InPlaceEditingOnNodesInDiagramEnabled
 - VcGantt 736
- InPlaceEditingOnNodesInTableEnabled
 - VcGantt 737
- InteractionMode
 - VcGantt 738
- IntervalCollection
 - VcCalendar 506
 - VcCalendarProfile 545
- IsEditable
 - VcUpdateBehavior 1417
 - VcUpdateBehaviorContext 1428
- KeepingNodesTogetherDataFieldIndex
 - VcGantt 738
- LabelPosition
 - VcDateLine 647
- LabelSizeDependence
 - VcLayer 1079
- LateEndDateDataFieldIndex
 - VcScheduler 1361
- LateStartDateDataFieldIndex
 - VcScheduler 1362
- LayerCollection
 - VcGantt 739
- LayerName
 - VcCurve 568
- LayersWithNonWorkInterval
 - VcGantt 739
- LeavingControlWhileDraggingAllowed
 - VcGantt 740
- Left
 - VcLegendView 1127
 - VcRect 1275
 - VcWorldView 1433
- LeftActualValue
 - VcLegendView 1128
 - VcWorldView 1434
- LeftMargin
 - VcLayerFormatField 1119
 - VcTableFormatField 1395
- LeftTable
 - VcGantt 740
- LeftTableDiagramWidthRatio
 - VcGantt 741
- LeftTableDiagramWidthRatioEx
 - VcGantt 741
- LegendElementsArrangement
 - VcBoundingBox 456
- LegendElementsBottomMargin
 - VcBoundingBox 456
- LegendElementsMaximumColumnCount
 - VcBoundingBox 456
- LegendElementsMaximumRowCount
 - VcBoundingBox 457
- LegendElementsTopMargin

- VcBoundingBox 457
- LegendFont
 - VcBoundingBox 457
- LegendText
 - VcLayer 1080
 - VcMapEntry 1199
- LegendTitle
 - VcBoundingBox 458
- LegendTitleFont
 - VcBoundingBox 458
- LegendTitleVisible
 - VcBoundingBox 459
- LegendView
 - VcGantt 742
- Level
 - VcGroupLevelLayout 993
- LevelMaximumForPagebreaks
 - VcHierarchyLevelLayout 1016
- LineColor
 - VcBox 465
 - VcCalendarGrid 525
 - VcCurve 568
 - VcDateLine 647
 - VcDateLineGrid 667
 - VcInterval 1049
 - VcLayer 1080
 - VcLinkAppearance 1161
 - VcNumericScale 1232
 - VcSection 1367
- LineColorDataFieldIndex
 - VcCalendarGrid 526
 - VcDateLineGrid 667
 - VcLayer 1080
- LineColorMapName
 - VcCalendarGrid 526
 - VcDateLineGrid 667
 - VcLayer 1081
- LineFormatCollection
 - VcGantt 742
- LineThickness
 - VcBox 465
 - VcCalendarGrid 526
 - VcCurve 569
 - VcDateLine 648
 - VcDateLineGrid 668
 - VcInterval 1049
 - VcLinkAppearance 1161
- LineType
 - VcBox 466
 - VcCalendarGrid 527
 - VcCurve 570
 - VcDateLine 649
 - VcDateLineGrid 669
 - VcInterval 1050
 - VcLinkAppearance 1163
- LinkAppearanceCollection
 - VcGantt 743
- LinkCollection
 - VcGantt 743
- LinkDataTableName
 - VcResourceScheduler2 1293
- LinkDurationDataFieldIndex
 - VcScheduler 1362
- LinkDurationFieldIndex
 - VcResourceScheduler2 1295
- LinkPredecessorDataFieldIndex
 - VcGantt 743
- LinkPredecessorOperationIDFieldIndex
 - VcResourceScheduler2 1295
- LinkPredecessorTaskIDFieldIndex
 - VcResourceScheduler2 1296
- LinksDataTableName
 - VcGantt 745

- LinkSuccessorDataFieldIndex
 - VcGantt 746
- LinkSuccessorOperationIDFieldIndex
 - VcResourceScheduler2 1297
- LinkSuccessorTaskIDFieldIndex
 - VcResourceScheduler2 1297
- LinkTypeDataFieldIndex
 - VcGantt 747
- MajorTicks
 - VcNumericScale 1232
 - VcRibbon 1348
- MajorTicksEx
 - VcNumericScale 1233
- MapCollection
 - VcGantt 748
- MarginsShownInInches
 - VcPrinter 1262
- Marked
 - VcBox 468
 - VcCurve 571
 - VcGroup 975
 - VcNode 1207
- MarkedNodesFilter
 - VcFilterCollection 691
- MarkingColor
 - VcWorldView 1434
- MaxHorizontalPagesCount
 - VcPrinter 1262
- MaximumEndDataFieldIndex
 - VcLayer 1081
- MaximumTextLineCount
 - VcBoxFormatField 500
 - VcTableFormatField 1396
- MaxVerticalPagesCount
 - VcPrinter 1263
- Millimeter
 - VcMapEntry 1199
- MinimumRowHeight
 - VcGantt 748
- MinimumStartDataFieldIndex
 - VcLayer 1081
- MinimumTextLineCount
 - VcBoxFormatField 500
 - VcTableFormatField 1396
- MinimumWidth
 - VcBoxFormatField 501
 - VcLayerFormatField 1119
- MinorTicks
 - VcNumericScale 1234
 - VcRibbon 1348
- MinorTicksEx
 - VcNumericScale 1234
- Mode
 - VcWorldView 1435
- ModificationsAllowed
 - VcGroupLevelLayout 993
- MouseProcessingEnabled
 - VcGantt 749
- Movable
 - VcDateLine 650
 - VcLayer 1082
- Moveable
 - VcBox 468
- MoveMode
 - VcGantt 749
- MovingGroupsVerticallyViaDiagramAllowed
 - VcGroupLevelLayout 994
- MovingGroupsVerticallyViaTableAllowed
 - VcGroupLevelLayout 994
- MovingLayersAsNodeWithShiftKeyAllowed
 - VcGantt 750
- MultipleBoxMarkingAllowed

- VcGantt 750
- MultiplePrimaryKeysAllowed
 - VcDataTable 624
- MultiState
 - VcTableFormatField 1396
- Name
 - VcBox 469
 - VcBoxFormat 487
 - VcCalendar 506
 - VcCalendarGrid 528
 - VcCalendarProfile 545
 - VcCurve 572
 - VcDataDefinitionField 598
 - VcDataTable 624
 - VcDataTableField 634
 - VcDateLine 650
 - VcFilter 684
 - VcGroup 975
 - VcGroupLevelLayout 994
 - VcHistogram 1023
 - VcInterval 1051
 - VcLayer 1082
 - VcLineFormat 1134
 - VcLinkAppearance 1164
 - VcMap 1181
 - VcNumericScale 1235
 - VcTable 1374
 - VcTableFormat 1384
 - VcTimeScale 1409
 - VcUpdateBehavior 1418
- NodeCalendarNameDataFieldIndex
 - VcGantt 751
- NodeCollection
 - VcGantt 751
 - VcGroup 976
- NodeCreationAllowed
 - VcGantt 752
- NodeCreationAtDroppingEnabled
 - VcGantt 752
- NodeCreationViaDoubleClick
 - VcGantt 753
- NodeCreationWithDialog
 - VcGantt 753
- NodeDurationDataFieldIndex
 - VcGantt 753
- NodeEndDateDataFieldIndex
 - VcGantt 754
- NodeID
 - VcBox 469
- NodeLevelLayout
 - VcGantt 754
- NodeRowNumberDataFieldIndex
 - VcGantt 754
- NodesAndGroupsBelowCollapsed
 - VcGroup 976
- NodesDataTableName
 - VcGantt 755
- NodeSeparationLinesVisible
 - VcHierarchyLevelLayout 1016
- NodesInHeader
 - VcGroup 977
- NodesInHeaders
 - VcGroupLevelLayout 995
 - VcHierarchyLevelLayout 1017
- NodeSortingDataFieldIndex
 - VcGantt 756
- NodeSortingOrder
 - VcGantt 757
- NodesOverlaid
 - VcGroup 977
 - VcGroupLevelLayout 995
 - VcHierarchyLevelLayout 1017
- NodeStartDateDataFieldIndex
 - VcGantt 757

- NodesUseCalendars
 - VcGantt 757
- NodeToolTipTextDataFieldIndex
 - VcGantt 758
- NominalScaleMaximum
 - VcHistogram 1023
- NominalScaleMinimum
 - VcHistogram 1024
- NonWorkIntervalBackgroundColor
 - VcLayer 1083
- NonWorkIntervalBackgroundColorDataFieldIndex
 - VcLayer 1083
- NonWorkIntervalBackgroundColorMapName
 - VcLayer 1083
- NonWorkIntervalLineColor
 - VcLayer 1084
- NonWorkIntervalLineColorDataFieldIndex
 - VcLayer 1084
- NonWorkIntervalLineColorMapName
 - VcLayer 1084
- NonWorkIntervalLineThickness
 - VcLayer 1085
- NonWorkIntervalLineType
 - VcLayer 1085
- NonWorkIntervalPattern
 - VcLayer 1086
- NonWorkIntervalPatternColor
 - VcLayer 1089
- NonWorkIntervalPatternColorDataFieldIndex
 - VcLayer 1089
- NonWorkIntervalPatternColorMapName
 - VcLayer 1090
- NonWorkIntervalPatternDataFieldIndex
 - VcLayer 1090
- NonWorkIntervalPatternMapName
 - VcLayer 1090
- NoOfColumns
 - VcTable 1374
- Number
 - VcMapEntry 1200
- NumericScaleCollection
 - VcGantt 758
 - VcHistogram 1024
- NumericScaleRescalingAllowed
 - VcGantt 759
- ObjectDrawEventsEnabled
 - VcLayer 1091
- ObserveDST
 - VcDateLineGrid 670
 - VcRibbon 1349
- OLEDragViaDiagram
 - VcGantt 759
- OLEDragViaTable
 - VcGantt 759
- OperationDataTableName
 - VcResourceScheduler2 1298
- OperationLoadPerItemFieldIndex
 - VcResourceScheduler2 1300
- OperationMaximumInterruptionTimeFieldIndex
 - VcResourceScheduler2 1300
- OperationMinimumSupplementTimeFieldIndex
 - VcResourceScheduler2 1301
- OperationOverlapQuantityFieldIndex
 - VcResourceScheduler2 1302

- OperationPostLoadFieldIndex
 - VcResourceScheduler2 1304
- OperationPostOffsetFieldIndex
 - VcResourceScheduler2 1304
- OperationPreparationLoadFieldIndex
 - VcResourceScheduler2 1305
- OperationPreparationOffsetFieldIndex
 - VcResourceScheduler2 1306
- OperationResultEndDateFieldIndex
 - VcResourceScheduler2 1306
- OperationResultPostEndDateFieldIndex
 - VcResourceScheduler2 1307
- OperationResultPreparationStartDateFieldIndex
 - VcResourceScheduler2 1308
- OperationResultProcessingTimeFieldIndex
 - VcResourceScheduler2 1308
- OperationResultSelectedTimingResourceIDFieldIndex
 - VcResourceScheduler2 1309
- OperationResultStartDateFieldIndex
 - VcResourceScheduler2 1309
- OperationResultStatusFieldIndex
 - VcResourceScheduler2 1310
- OperationRouteFieldIndex
 - VcResourceScheduler2 1311
- OperationSequenceNumberFieldIndex
 - VcResourceScheduler2 1311
- OperationStartLockDateFieldIndex
 - VcResourceScheduler2 1312
- OperationTaskIDFieldIndex
 - VcResourceScheduler2 1313
- OperationWorkInProgressFieldIndex
 - VcResourceScheduler2 1313
- Operator
 - VcFilterSubCondition 699
- OptimizedNodesSortDataFieldIndex
 - VcGroupLevelLayout 995
- OptimizedNodesSortOrder
 - VcGroupLevelLayout 996
- Orientation
 - VcPrinter 1263
- Origin
 - VcBox 470
- OutgoingLinks
 - VcNode 1208
- OutputFormatForCenterDate
 - VcInfoWindow 1037
- OutputFormatForDuration
 - VcInfoWindow 1039
- OutputFormatForEndDate
 - VcInfoWindow 1040
- OutputFormatForStartDate
 - VcInfoWindow 1041
- OverlaidNodesSortDataFieldIndex
 - VcGroupLevelLayout 996
- OverlaidNodesSortOrder
 - VcGroupLevelLayout 996
- OverlapLayerEnabled
 - VcGantt 760
- OverlapLayerName
 - VcGantt 760
- OverloadResultsCalendarName
 - VcCurve 572
- PagebreakMode
 - VcGroupLevelLayout 997
 - VcHierarchyLevelLayout 1017
- PageDescription
 - VcPrinter 1264
- PageDescriptionString
 - VcPrinter 1264
- PageFrame

- VcPrinter 1265
- PageNumberMode
 - VcPrinter 1265
- PageNumbers
 - VcPrinter 1266
- PagePaddingEnabled
 - VcPrinter 1266
- PanningModeAllowed
 - VcGantt 760
- PaperSize
 - VcPrinter 1267
- PartialLoadThreshold
 - VcGantt 761
- Pattern
 - VcCalendarGrid 528
 - VcInterval 1051
 - VcLayer 1091
 - VcMapEntry 1200
- PatternBackgroundColor
 - VcBoxFormatField 501
- PatternBackgroundColorAsARGB
 - VcLineFormatField 1145
 - VcNumericScale 1235
 - VcRibbon 1349
 - VcTableFormatField 1397
- PatternBackgroundColorDataFieldIndex
 - VcLineFormatField 1146
 - VcTableFormatField 1397
- PatternBackgroundColorMapName
 - VcLineFormatField 1146
 - VcTableFormatField 1397
- PatternColor
 - VcCalendarGrid 531
 - VcInterval 1055
 - VcLayer 1094
- PatternColorAsARGB
 - VcBoxFormatField 502
 - VcLineFormatField 1146
 - VcNumericScale 1235
 - VcRibbon 1350
 - VcTableFormatField 1398
- PatternColorDataFieldIndex
 - VcCalendarGrid 532
 - VcLayer 1095
 - VcTableFormatField 1398
- PatternColorMapName
 - VcCalendarGrid 532
 - VcLayer 1095
 - VcLineFormatField 1147
 - VcTableFormatField 1398
- PatternDataFieldIndex
 - VcCalendarGrid 533
 - VcLayer 1096
- PatternEx
 - VcLineFormatField 1147
 - VcNumericScale 1236
 - VcRibbon 1350
 - VcTableFormatField 1399
- PatternExDataFieldIndex
 - VcLineFormatField 1150
 - VcTableFormatField 1402
- PatternExMapName
 - VcLineFormatField 1151
 - VcTableFormatField 1402
- PatternMapName
 - VcCalendarGrid 533
 - VcLayer 1097
- Period
 - VcDateLineGrid 670
- PhantomDrawingWhileDraggingEnabled
 - VcGantt 762
- PhantomLayerHeight

- VcGantt 763
- PlanningEndDate
 - VcResourceScheduler2 1314
- PlanningStartDate
 - VcResourceScheduler2 1315
- PlanningStrategy
 - VcResourceScheduler2 1316
- PointsEquidistant
 - VcCurve 573
- Position
 - VcRibbon 1354
 - VcTable 1374
- PredecessorLayerName
 - VcLinkAppearance 1164
- PredecessorNode
 - VcLink 1156
- PredecessorPortSymbol
 - VcLinkAppearance 1165
- PrimaryKey
 - VcDataTableField 635
- PrintDate
 - VcPrinter 1267
- Printer
 - VcGantt 763
- PrinterName
 - VcPrinter 1268
- PrintPreviewWithFirstPage
 - VcPrinter 1268
- Priority
 - VcBox 470
 - VcCalendarGrid 533
 - VcDateLine 651
 - VcDateLineGrid 671
 - VcLayerFormatField 1120
- ReferenceDate
 - VcDateLineGrid 672
 - VcInfoWindow 1043
- VcRibbon 1354
- ReferencePoint
 - VcBox 471
- RelationshipFieldIndex
 - VcDataTableField 635
- ReOptimizeNodesInGroupsEnabled
 - VcPrinter 1269
- Resizing
 - VcBox 471
- ResourceCalendarNameFieldIndex
 - VcResourceScheduler2 1317
- ResourceCapacityType
 - VcResourceScheduler2 1317
- ResourceCapacityTypeFieldIndex
 - VcResourceScheduler2 1318
- ResourceConstraintTypeFieldIndex
 - VcResourceScheduler2 1319
- ResourceDataTableName
 - VcResourceScheduler2 1320
- ResourceEfficiencyFieldIndex
 - VcResourceScheduler2 1322
- ResourceGroupDataTableName
 - VcResourceScheduler2 1323
- ResourceGroupIDFieldIndex
 - VcResourceScheduler2 1324
- ResourceNameFieldIndex
 - VcResourceScheduler2 1324
- ResourceResultLoadCurveNamePrefix
 - VcResourceScheduler2 1325
- ResourceResultStockCurveNamePrefix
 - VcResourceScheduler2 1326
- ResourceScheduler2
 - VcGantt 763
- ResourceSelectionStrategy
 - VcResourceScheduler2 1327

- ResourceType
 - VcResourceScheduler2 1328
- RestoreAutoCollapsedGroups
 - VcGroupLevelLayout 997
 - VcHierarchyLevelLayout 1018
- RestoreAutoExpandedGroups
 - VcGroupLevelLayout 997
 - VcHierarchyLevelLayout 1018
- ResultProcessingStepCount
 - VcResourceScheduler2 1329
- Ribbon
 - VcSection 1369
 - VcTimeScale 1410
- Right
 - VcRect 1276
- RightMargin
 - VcLayerFormatField 1120
 - VcTableFormatField 1403
- RightTable
 - VcGantt 764
- RightTableDiagramWidthRatio
 - VcGantt 764
- RightTableDiagramWidthRatioEx
 - VcGantt 765
- RoundedLinkSlantsEnabled
 - VcGantt 765
- RoutingType
 - VcLinkAppearance 1165
- RowBackColorAsARGB
 - VcGroupLevelLayout 998
 - VcHistogram 1025
- RowBackColorDataFieldIndex
 - VcGroupLevelLayout 998
- RowBackColorMapName
 - VcGroupLevelLayout 998
- RowBackgroundColorAsARGB
 - VcNodeLevelLayout 1222
- RowBackgroundColorDataFieldIndex
 - VcNodeLevelLayout 1222
- RowBackgroundColorMapName
 - VcNodeLevelLayout 1222
- RowHeightReductionEnabled
 - VcGantt 766
- RowMargins
 - VcGantt 766
- RowPattern
 - VcGroupLevelLayout 999
 - VcHistogram 1025
 - VcNodeLevelLayout 1223
- RowPatternColorAsARGB
 - VcGroupLevelLayout 1002
 - VcHistogram 1028
 - VcNodeLevelLayout 1223
- RowPatternColorDataFieldIndex
 - VcGroupLevelLayout 1002
 - VcNodeLevelLayout 1223
- RowPatternColorMapName
 - VcGroupLevelLayout 1002
 - VcNodeLevelLayout 1224
- RowPatternDataFieldIndex
 - VcGroupLevelLayout 1003
 - VcNodeLevelLayout 1224
- RowPatternMapName
 - VcGroupLevelLayout 1003
 - VcNodeLevelLayout 1224
- Sash3DStyleEnabled
 - VcGantt 767
- SashThickness
 - VcGantt 767
- ScalingMode
 - VcPrinter 1269
- ScheduledProjectEndDate
 - VcScheduler 1362
- ScheduledProjectStartDate

- VcScheduler 1363
- Scheduler
 - VcGantt 767
- ScheduleSuccessorsOnlyEnabled
 - VcScheduler 1363
- ScrollBarMode
 - VcLegendView 1128
 - VcWorldView 1435
- ScrollEventsEnabled
 - VcGantt 768
- SecondsPerWorkday
 - VcCalendar 507
- Section
 - VcTimeScale 1410
- SelectedNodesMovingTogether
 - VcGantt 768
- SelectedRowBackgroundColor
 - VcGantt 769
- SelectionViaRubberRectAllowed
 - VcGantt 769
- SeparationLineColor
 - VcGroupLevelLayout 1003
 - VcHierarchyLevelLayout 1018
 - VcNodeLevelLayout 1225
 - VcTableFormat 1385
- SeparationLineColorDataFieldIndex
 - VcGroupLevelLayout 1004
- SeparationLineColorMapName
 - VcGroupLevelLayout 1004
- SeparationLineInterval
 - VcNodeLevelLayout 1225
- SeparationLinesVisible
 - VcGroupLevelLayout 1004
 - VcHierarchyLevelLayout 1019
 - VcNodeLevelLayout 1225
- SeparationLinesVisibleAtTop
 - VcGroupLevelLayout 1005
- VcNodeLevelLayout 1226
- SeparationLineThickness
 - VcGroupLevelLayout 1005
 - VcHierarchyLevelLayout 1019
 - VcNodeLevelLayout 1226
- SeparationLineType
 - VcGroupLevelLayout 1006
 - VcHierarchyLevelLayout 1020
 - VcNodeLevelLayout 1227
- Shape
 - VcLayer 1099
- ShowSnapLines
 - VcGantt 769
- ShowSnapMarkings
 - VcGantt 770
- Sizeable
 - VcLayer 1101
- SnapTarget
 - VcCalendarGrid 534
 - VcDateLine 651
 - VcDateLineGrid 672
- SnapTargetMode
 - VcNode 1209
- SnapTargetNodesSelectionMode
 - VcGantt 770
- SortDataFieldIndex
 - VcGroupLevelLayout 1007
 - VcNodeLevelLayout 1227
- SortOrder
 - VcGroupLevelLayout 1007
 - VcNodeLevelLayout 1227
- Source
 - VcCurve 573
- Specification
 - VcBox 472
 - VcBoxFormat 487
 - VcCalendar 507

- VcCalendarGrid 534
- VcCalendarProfile 546
- VcCurve 574
- VcDateLine 652
- VcFilter 685
- VcGroupLevelLayout 1008
- VcInterval 1055
- VcLayer 1102
- VcLineFormat 1134
- VcMap 1182
- VcUpdateBehavior 1418
- StackReferenceName
 - VcCurve 574
- StartDataFieldIndex
 - VcLayer 1102
- StartDate
 - VcSection 1370
- StartDateForAutomaticScheduling
 - VcGantt 770
 - VcScheduler 1363
- StartDateNotEarlierThanDataFieldIndex
 - VcScheduler 1363
- StartDateTime
 - VcInterval 1055
- StartMonth
 - VcInterval 1056
- StartSnapTarget
 - VcCalendarGrid 535
 - VcLayer 1102
- StartTime
 - VcInterval 1056
- StartWeekday
 - VcInterval 1056
- StringsCaseSensitive
 - VcFilter 685
- SubCondition
 - VcFilter 686
- SubConditionCount
 - VcFilter 686
- SubGroups
 - VcGroup 977
- SubRowMargins
 - VcGantt 771
- SuccessorLayerName
 - VcLinkAppearance 1166
- SuccessorNode
 - VcLink 1157
- SuccessorPortSymbol
 - VcLinkAppearance 1167
- SummaryBarsVisible
 - VcGantt 771
 - VcGroupLevelLayout 1008
 - VcHierarchyLevelLayout 1020
- SuperGroup
 - VcGroup 978
 - VcNode 1209
- TableCollection
 - VcGantt 772
- TableColumnRanges
 - VcPrinter 1270
- TableColumnWidthOptimizationAllowed
 - VcGantt 772
- TableFormatCollection
 - VcTable 1375
- TableTimeScaleOnAllPages
 - VcPrinter 1270
- TableWidthAdoptionFromViewOnScreen
 - VcPrinter 1271
- TaskDataTableName
 - VcResourceScheduler2 1330
- TaskDueDateFieldIndex

- VcResourceScheduler2 1331
- TaskPlanningStrategyFieldIndex
 - VcResourceScheduler2 1331
- TaskPriorityFieldIndex
 - VcResourceScheduler2 1332
- TaskQuantityFieldIndex
 - VcResourceScheduler2 1333
- TaskReleaseDateFieldIndex
 - VcResourceScheduler2 1334
- TaskResultEndDateFieldIndex
 - VcResourceScheduler2 1335
- TaskResultPostEndDateFieldIndex
 - VcResourceScheduler2 1335
- TaskResultPreparationStartDateFieldIndex
 - VcResourceScheduler2 1336
- TaskResultProcessingStepFieldIndex
 - VcResourceScheduler2 1336
- TaskResultProcessingTimeFieldIndex
 - VcResourceScheduler2 1337
- TaskResultRouteFieldIndex
 - VcResourceScheduler2 1338
- TaskResultStartDateFieldIndex
 - VcResourceScheduler2 1338
- Text
 - VcBoundingBox 459
 - VcDateLine 652
 - VcInterval 1057
- TextAlignment
 - VcRibbon 1354
- TextAndGraphicsCombined
 - VcTableFormatField 1403
- TextDataFieldIndex
 - VcLayerFormatField 1120
 - VcLineFormatField 1151
 - VcTableFormatField 1403
- TextEntrySupplyingEventEnabled
- VcGantt 773
- TextFont
 - VcBoundingBox 460
 - VcBoxFormatField 503
 - VcLayerFormatField 1121
 - VcTableFormatField 1403
- TextFontColor
 - VcBoxFormatField 503
 - VcLayerFormatField 1121
 - VcLineFormatField 1151
 - VcTableFormatField 1404
- TextFontColorDataFieldIndex
 - VcLayerFormatField 1121
 - VcLineFormatField 1152
 - VcTableFormatField 1404
- TextFontColorMapName
 - VcLayerFormatField 1121
 - VcLineFormatField 1152
 - VcTableFormatField 1404
- TextFontDataFieldIndex
 - VcLayerFormatField 1122
 - VcLineFormatField 1152
 - VcTableFormatField 1405
- TextFontMapName
 - VcLayerFormatField 1122
 - VcLineFormatField 1153
 - VcTableFormatField 1405
- TextLineCount
 - VcLayerFormatField 1122
 - VcLineFormatField 1153
- TextLineCountDataFieldIndex
 - VcLayerFormatField 1123
- TextLineCountMapName
 - VcLayerFormatField 1123
- ThreeDEffect
 - VcLayer 1103
 - VcNumericScale 1239

- VcTableFormat 1385
- VcTimeScale 1411
- TickColor
 - VcNumericScale 1240
 - VcRibbon 1355
- TickPosition
 - VcRibbon 1355
- TimeColumnEndDate
 - VcPrinter 1271
- TimeColumnStartDate
 - VcPrinter 1272
- TimeScaleAdjustment
 - VcPrinter 1272
- TimeScaleCollection
 - VcGantt 773
- TimeScaleDialogEnabled
 - VcGantt 774
- TimeScaleEnd
 - VcGantt 774
- TimeScaleRescalingAllowed
 - VcGantt 775
- TimeScaleStart
 - VcGantt 775
- TimeUnit
 - VcCurve 575
 - VcGantt 776
 - VcInterval 1057
 - VcSection 1370
- TimeUnitsPerStep
 - VcGantt 777
- Title
 - VcNumericScale 1240
- ToleranceTimeOnASAPDueDates
 - VcResourceScheduler2 1339
- ToleranceTimeOnJITReleaseDates
 - VcResourceScheduler2 1339
- ToleranceTimeOnStartLockDates
 - VcResourceScheduler2 1340
- ToolTipChangeDuration
 - VcGantt 777
- ToolTipDuration
 - VcGantt 777
- ToolTipPointerDuration
 - VcGantt 778
- ToolTipShowAfterClick
 - VcGantt 778
- ToolTipTextSupplyingEventEnabled
 - VcGantt 779
- Top
 - VcLegendView 1129
 - VcRect 1276
 - VcWorldView 1436
- TopActualValue
 - VcLegendView 1129
 - VcWorldView 1436
- TopMargin
 - VcLayerFormatField 1123
 - VcTableFormatField 1405
- TotalFloatDataFieldIndex
 - VcScheduler 1364
- TrackingSpaceBackgroundColor
 - VcGantt 779
- TrackingSpacePattern
 - VcGantt 780
- TrackingSpacePatternColor
 - VcGantt 783
- TruncatedTextSuppressed
 - VcLayerFormatField 1124
- TurningAnnotationEnabled
 - VcDateLine 653
 - VcDateLineGrid 672
- Type
 - VcBoundingBox 461
 - VcBoxFormatField 504

- VcCalendar 507
- VcCalendarProfile 546
- VcCurve 575
- VcDataDefinitionField 598
- VcDataTableField 636
- VcInterval 1057
- VcMap 1183
- VcRibbon 1355
- VcTableFormatField 1405
- VcUpdateBehaviorContext 1428
- Unit
 - VcDateLineGrid 673
 - VcNumericScale 1241
- UnitEx
 - VcNumericScale 1241
- UnitLabel
 - VcNumericScale 1241
- UnitSeparation
 - VcRibbon 1356
- UnitsPerStep
 - VcCurve 576
- UnitWidth
 - VcNumericScale 1242
 - VcSection 1371
- UnitWidthEx
 - VcSection 1371
- UpdateBehaviorCollection
 - VcGantt 783
- UpdateBehaviorName
 - VcBox 473
 - VcCurve 577
 - VcDateLine 653
 - VcNode 1210
 - VcNumericScale 1242
 - VcTable 1375
 - VcTimeScale 1411
 - VcWorldView 1437
- UpdateMode
 - VcUpdateBehaviorContext 1429
- UsedAsOverlapLayer
 - VcLayer 1103
- UseGraphicalAttributes
 - VcInterval 1058
- UseGraphicalAttributesOfIntervals
 - VcCalendarGrid 535
- UseHigherDiagramHistogramHeightRatioPrecision
 - VcGantt 784
- UseHigherTableDiagramWidthRatioPrecision
 - VcGantt 784
- UseReferenceDate
 - VcDateLineGrid 673
 - VcInfoWindow 1043
 - VcRibbon 1357
- UseSnapTargetsInInteractions
 - VcGantt 785
- UseTwinLineSashPhantom
 - VcGantt 785
- ValencyDataFieldIndex
 - VcCurve 577
- VcCalendarGrid
 - VcPrinter 1273
- VerticalNodeMovementAllowed
 - VcGantt 785
- VerticalNodeMovementViaTableAllowed
 - VcGantt 786
- VerticalOffset
 - VcLayer 1104
- VerticalOffsetDataFieldIndex
 - VcLayer 1104
- VerticalOffsetMapName
 - VcLayer 1104
- ViewComponentsBackgroundColor

- VcGantt 786
- ViewComponentsBorderColor
 - VcGantt 786
- Visible
 - VcBox 473
 - VcCalendarGrid 535
 - VcCurve 577
 - VcDateLine 653
 - VcDateLineGrid 674
 - VcGroup 978
 - VcGroupLevelLayout 1008
 - VcHistogram 1028
 - VcInfoWindow 1043
 - VcLayer 1105
 - VcLegendView 1130
 - VcLinkAppearance 1167
 - VcTable 1375
 - VcWorldView 1437
- VisibleDataFieldIndex
 - VcCalendarGrid 536
 - VcDateLine 654
 - VcDateLineGrid 674
- VisibleInLegend
 - VcLayer 1105
- VisibleMapName
 - VcCalendarGrid 536
 - VcDateLine 654
 - VcDateLineGrid 674
- WaitCursorEnabled
 - VcGantt 787
- Width
 - VcLegendView 1130
 - VcRect 1277
 - VcWorldView 1438
- WidthActualValue
 - VcLegendView 1130
 - VcWorldView 1438

- WindowMode
 - VcLegendView 1131
- WorkInProcessType
 - VcResourceScheduler2 1340
- WorldView
 - VcGantt 787
- WritingDebugFilesEnabled
 - VcResourceScheduler2 1341
- ZoomFactor
 - VcGantt 787
- ZoomFactorAsDouble
 - VcPrinter 1273
- ZoomingPerMouseWheelAllowed
 - VcGantt 788

Property page

- Border Area 235
- General 224
- Layout 247
- Link 253
- Node 237
- Objects 251
- Schedule 255

Property Page

- Additional Views 243

PutInOrderAfter

- Method of
 - VcCalendarProfile 547
 - VcDateLine 654
 - VcHistogram 1030
 - VcInterval 1058
 - VcLayer 1106
 - VcLinkAppearance 1168
 - VcUpdateBehavior 1419

R**RecalculateAllStructureCodes**

- Method of

VcGantt 819

Rect

see also

VcRect 1274

Reference curve 128

ReferenceDate

Property of

VcDateLineGrid 672

VcInfoWindow 1043

VcRibbon 1354

ReferencePoint

Property of

VcBox 471

RelatedDataRecord

Method of

VcDataRecord 611

VcGroup 980

VcLink 1158

VcNode 1214

Relation

type 254

RelationshipFieldIndex

Property of

VcDataTableField 635

Remove

Method of

VcBoxCollection 483

VcBoxFormatCollection 495

VcCalendarCollection 519

VcCalendarGridCollection 543

VcCalendarProfileCollection 551

VcCurveCollection 594

VcDataRecordCollection 619

VcDateLineCollection 662

VcDateLineGridCollection 681

VcFilterCollection 695

VcGroupLevelLayoutCollection
1013

VcIntervalCollection 1064

VcLayerCollection 1112

VcLineFormatCollection 1140

VcLinkAppearanceCollection 1174

VcMapCollection 1192

VcUpdateBehaviorCollection 1425

RemoveFormatField

Method of

VcBoxFormat 489

VcLayerFormat 1116

VcLineFormat 1135

RemoveSubCondition

Method of

VcFilter 689

ReOptimizeNodes

Method of

VcGroup 980

ReOptimizeNodesInGroupsEnabled

Property of

VcPrinter 1269

Reset

Method of

VcGantt 819

Resizing

Property of

VcBox 471

Resource scheduler

lock date 1312

resource group 1324

resource name 1325

scheduling progress 931

task end date 1335

task planning strategy 1332

task priority 1333

task processing time 1337

task route 1338
 task start date 1338
 warnings 932

Resource Scheduler 190

resource scheduling

calendar name 1317

Resource scheduling

allowed due date variation 1339
 allowed release date variation 1339
 assignment, visible 1284
 assignment, associated operation 1287
 assignment, associated resource 1287
 assignment, data set generation 1283
 assignment, quantity multiplier 1284
 assignment, table name 1282
 assignmentData table 1294
 calendar for minimum supplement time 1289
 calendar, default name 1292
 capacity of the production system 1317
 capacity, full usage of 1292
 completion 1314
 debug files 1341
 end date of scheduling period 1314
 events 1291
 job completion 1340
 link predecessor operation 1295, 1297
 link, predecessor task 1296
 link, successor task 1297
 link, temporal distance 1295
 locked start date allowance 1340
 operation finish date 1307
 operation follow-up time resource-independent 1304

operation lead time resource-independent 1306
 operation maximum interruption time 1301
 operation minimum supplement time 1301
 operation post time 1304
 operation preparation time 1305
 operation processing time 1308
 operation start date 1309
 operation status 1310
 operation table, quantity multiplier 1300
 operation, default value for maximum interruption time 1291
 operation, end date of post time 1307, 1335
 operation, ID of timing resource 1309
 operation, overlap quantity 1303
 operation, start date of preparation time 1308, 1336
 operation, table name 1299
 operations table, associated task 1313
 operations table, routes 1311
 operations table, sequence of operations 1311
 planning strategy 1316
 process 1342
 resource capacity absolute or relative 1286
 resource constraint 1319
 resource selection 1327
 resource selection strategy 1289
 resource type 1329
 resource type single 1319
 resource, maximum work load limit 1286
 resource, minimum work load limit 1285

- resource, name of group table 1323
- resource, table name 1321
- setting to the Gantt object 763
- start date of scheduling period 1315
- stock curve 1326
- task due date 1331
- task quantity 1334
- task release date 1334
- task sequence 1337
- task, table name 1330
- tasks, number of 1329
- time unit, basic 1290
- time units per step 1290
- workload curve 1325
- resource scheduling**
 - efficiency 1322
- ResourceCalendarNameFieldIndex**
 - Property of
 - VcResourceScheduler2 1317
- ResourceCapacityType**
 - Property of
 - VcResourceScheduler2 1317
- ResourceCapacityTypeFieldIndex**
 - Property of
 - VcResourceScheduler2 1318
- ResourceConstraintTypeFieldIndex**
 - Property of
 - VcResourceScheduler2 1319
- ResourceDataTableName**
 - Property of
 - VcResourceScheduler2 1320
- ResourceEfficiencyFieldIndex**
 - Property of
 - VcResourceScheduler2 1322
- ResourceGroupDataTableName**
 - Property of
 - VcResourceScheduler2 1323
- ResourceGroupIDFieldIndex**
 - Property of
 - VcResourceScheduler2 1324
- ResourceNameFieldIndex**
 - Property of
 - VcResourceScheduler2 1324
- ResourceResultLoadCurveNamePrefix**
 - Property of
 - VcResourceScheduler2 1325
- ResourceResultStockCurveNamePrefix**
 - Property of
 - VcResourceScheduler2 1326
- ResourceScheduler2**
 - Property of
 - VcGantt 763
 - see also
 - VcResourceScheduler2 1278
- ResourceSelectionStrategy**
 - Property of
 - VcResourceScheduler2 1327
- ResourceType**
 - Property of
 - VcResourceScheduler2 1328
- RestoreAutoCollapsedGroups**
 - Property of
 - VcGroupLevelLayout 997
 - VcHierarchyLevelLayout 1018
- RestoreAutoExpandedGroups**
 - Property of
 - VcGroupLevelLayout 997
 - VcHierarchyLevelLayout 1018
- ResultProcessingStepCount**
 - Property of
 - VcResourceScheduler2 1329
- Return status 108**

Ribbon 209, 350

- calendar 1345
- date format 1346
- font 1347
- font color 1347
- major ticks 1348
- minor ticks 1349
- Pattern 1350
- position 1354
- Property of
 - VcSection 1369
 - VcTimeScale 1410
- reference date 1354, 1357
- see also
 - VcRibbon 1344
- text alignment 1354
- tick color 1355
- tick position 1355
- type 1355
- unit separation 1356

Ribbons

- background color of pattern 1349
- pattern color 1350

Right

- Property of
 - VcRect 1276

RightMargin

- Property of
 - VcLayerFormatField 1120
 - VcTableFormatField 1403

RightTable

- Property of
 - VcGantt 764

RightTableDiagramWidthRatio

- Property of
 - VcGantt 764

RightTableDiagramWidthRatioEx

Property of

VcGantt 765

RoundedLinkSlantsEnabled

Property of

VcGantt 765

RoutingType

Property of

VcLinkAppearance 1165

Row background color

alternating 250

Row height

reduction 231, 232

RowBackColorAsARGB

Property of

VcGroupLevelLayout 998

VcHistogram 1025

RowBackColorDataFieldIndex

Property of

VcGroupLevelLayout 998

RowBackColorMapName

Property of

VcGroupLevelLayout 998

RowBackgroundColorAsARGB

Property of

VcNodeLevelLayout 1222

RowBackgroundColorDataFieldIndex

Property of

VcNodeLevelLayout 1222

RowBackgroundColorMapName

Property of

VcNodeLevelLayout 1222

RowHeightReductionEnabled

Property of

VcGantt 766

RowMargins

Property of

VcGantt 766

RowPattern

- Property of
 - VcGroupLevelLayout 999
 - VcHistogram 1025
 - VcNodeLevelLayout 1223

RowPatternColorAsARGB

- Property of
 - VcGroupLevelLayout 1002
 - VcHistogram 1028
 - VcNodeLevelLayout 1223

RowPatternColorDataFieldIndex

- Property of
 - VcGroupLevelLayout 1002
 - VcNodeLevelLayout 1223

RowPatternColorMapName

- Property of
 - VcGroupLevelLayout 1002
 - VcNodeLevelLayout 1224

RowPatternDataFieldIndex

- Property of
 - VcGroupLevelLayout 1003
 - VcNodeLevelLayout 1224

RowPatternMapName

- Property of
 - VcGroupLevelLayout 1003
 - VcNodeLevelLayout 1224

S

Sash

- 3D style switched on/off 767
- Double phantom line while moving sash switched on/off 785
- Thickness 767

Sash3DStyleEnabled

- Property of
 - VcGantt 767

SashThickness

- Property of
 - VcGantt 767

SaveAsEx

- Method of
 - VcGantt 820

ScalingMode

- Property of
 - VcPrinter 1269

ScheduledProjectEndDate

- Property of
 - VcScheduler 1362

ScheduledProjectStartDate

- Property of
 - VcScheduler 1363

ScheduleProject

- Method of
 - VcGantt 820
 - VcScheduler 1364

Scheduler

- Property of
 - VcGantt 767
- see also
 - VcScheduler 1358

ScheduleSuccessorsOnlyEnabled

- Property of
 - VcScheduler 1363

Scheduling 194, 767

- actual end date 1359
- actual start date 1359
- Autoschedule 256
- duration 1360
- earliest possible end date 1360
- earliest possible start date 1360
- free float 1361
- latest possible end date 1361
- latest possible start date 1362
- link duration 1362

- schedule input 255
- schedule result 255
- scheduled end date 1361
- scheduled start date 1363
- scheduling only of nodes with predecessors 1363
- total float 1364

Scheduling: 1362, 1363

Screen section

- move 234

Scroll events

- enable/disable 231

ScrollBarMode

- Property of
 - VcLegendView 1128
 - VcWorldView 1435

ScrollComponentStartTo

- Method of
 - VcGantt 821

ScrollEventsEnabled

- Property of
 - VcGantt 768

Scrolling

- to a value in histogram 1031

ScrollToDate

- Method of
 - VcGantt 822

ScrollToGroupLine

- Method of
 - VcGantt 822

ScrollToNode

- Method of
 - VcGantt 823

ScrollToNodeLine

- Method of
 - VcGantt 824

ScrollToValue

- Method of
 - VcHistogram 1031

SecondsPerWorkday

- Property of
 - VcCalendar 507

Section 208, 348

- collapsing workfree periods 350
- Property of
 - VcTimeScale 1410
- see also
 - VcSection 1366
- unit width 350

Security guidelines

- run time 197

SelectCalendarProfiles

- Method of
 - VcCalendarProfileCollection 552

SelectedNodesMovingTogether

- Property of
 - VcGantt 768

SelectedRowBackgroundColor

- Property of
 - VcGantt 769

SelectGroups

- Method of
 - VcGroupCollection 985

Selection

- Rubber rect 234

SelectionViaRubberRectAllowed

- Property of
 - VcGantt 769

SelectLinks

- Method of
 - VcLinkCollection 1178

SelectMaps

- Method of
 - VcMapCollection 1192

SelectNodes

Method of

VcNodeCollection 1218

Separation lines

interval 1225

SeparationLineColor

Property of

VcGroupLevelLayout 1003

VcHierarchyLevelLayout 1018

VcNodeLevelLayout 1225

VcTableFormat 1385

SeparationLineColorDataFieldIndex

Property of

VcGroupLevelLayout 1004

SeparationLineColorMapName

Property of

VcGroupLevelLayout 1004

SeparationLineInterval

Property of

VcNodeLevelLayout 1225

SeparationLinesVisible

Property of

VcGroupLevelLayout 1004

VcHierarchyLevelLayout 1019

VcNodeLevelLayout 1225

SeparationLinesVisibleAtTop

Property of

VcGroupLevelLayout 1005

VcNodeLevelLayout 1226

SeparationLineThickness

Property of

VcGroupLevelLayout 1005

VcHierarchyLevelLayout 1019

VcNodeLevelLayout 1226

SeparationLineType

Property of

VcGroupLevelLayout 1006

VcHierarchyLevelLayout 1020

VcNodeLevelLayout 1227

SetImageResource

Method of

VcGantt 825

SetPositionInView

Method of

VcNode 1214

SetValues

Method of

VcCurve 586

SetXYOffset

Method of

VcBox 476

SetXYOffsetByTopLeftPixel

Method of

VcBox 476

Shape

Property of

VcLayer 1099

Shift calendar

annotation of the time ribbon 1057

Shipping 17**Show group 974****ShowAboutDialog**

Method of

VcGantt 826

ShowEditGroupDialog

Method of

VcGantt 827

ShowExportGraphicsDialog

Method of

VcGantt 827

ShowLinkEditDialog

Method of

VcGantt 828

ShowNodeEditDialog

- Method of
 - VcGantt 829
- ShowPageSetupDialog**
 - Method of
 - VcGantt 830
- ShowPrintDialog**
 - Method of
 - VcGantt 830
- ShowPrinterSetupDialog**
 - Method of
 - VcGantt 830
- ShowPrintPreviewDialog**
 - Method of
 - VcGantt 831
- ShowSnapLines**
 - Property of
 - VcGantt 769
- ShowSnapMarkings**
 - Property of
 - VcGantt 770
- Sizeable**
 - Property of
 - VcLayer 1101
- Smallest time interval 225**
- Snap Target ar used/not used 785**
- Snap targets**
 - manual selection 770
 - Select mode for node 1209
 - Target lines 770
- SnapTarget**
 - Property of
 - VcCalendarGrid 534
 - VcDateLine 651
 - VcDateLineGrid 672
- SnapTargetMode**
 - Property of
 - VcNode 1209
- SnapTargetNodesSelectionMode**
 - Property of
 - VcGantt 770
- SortDataFieldIndex**
 - Property of
 - VcGroupLevelLayout 1007
 - VcNodeLevelLayout 1227
- SortGroups**
 - Method of
 - VcGantt 831
- Sorting 199, 297**
 - enquire the sorting properties 754
- SortNodes**
 - Method of
 - VcGantt 831
- SortOrder**
 - Property of
 - VcGroupLevelLayout 1007
 - VcNodeLevelLayout 1227
- Source**
 - Property of
 - VcCurve 573
- Specification**
 - Property of
 - VcBox 472
 - VcBoxFormat 487
 - VcCalendar 507
 - VcCalendarGrid 534
 - VcCalendarProfile 546
 - VcCurve 574
 - VcDateLine 652
 - VcFilter 685
 - VcGroupLevelLayout 1008
 - VcInterval 1055
 - VcLayer 1102
 - VcLineFormat 1134
 - VcMap 1182

- VcUpdateBehavior 1418
- StackReferenceName**
 - Property of
 - VcCurve 574
- StartDataFieldIndex**
 - Property of
 - VcLayer 1102
- StartDate**
 - Property of
 - VcSection 1370
- StartDateForAutomaticScheduling**
 - Property of
 - VcGantt 770
 - VcScheduler 1363
- StartDateNotEarlierThanDataFieldIndex**
 - Property of
 - VcScheduler 1363
- StartDateTime**
 - Property of
 - VcInterval 1055
- StartMonth**
 - Property of
 - VcInterval 1056
- StartSnapTarget**
 - Property of
 - VcCalendarGrid 535
 - VcLayer 1102
- StartTime**
 - Property of
 - VcInterval 1056
- StartWeekday**
 - Property of
 - VcInterval 1056
- StringsCaseSensitive**
 - Property of
 - VcFilter 685
- SubCondition**
 - Property of
 - VcFilter 686
- SubConditionCount**
 - Property of
 - VcFilter 686
- Subdiagram 417, 420**
- SubGroups**
 - Property of
 - VcGroup 977
- SubRowMargins**
 - Property of
 - VcGantt 771
- SuccessorLayerName**
 - Property of
 - VcLinkAppearance 1166
- SuccessorNode**
 - Property of
 - VcLink 1157
- SuccessorPortSymbol**
 - Property of
 - VcLinkAppearance 1167
- Summary bar 120**
- Summary bars**
 - displaying 295
- SummaryBarsVisible**
 - Property of
 - VcGantt 771
 - VcGroupLevelLayout 1008
 - VcHierarchyLevelLayout 1020
- SuperGroup**
 - Property of
 - VcGroup 978
 - VcNode 1209
- Support 21**
- Suppress empty pages 410**
- SuspendUpdate**

Method of
VcGantt 832

T

Tabel format

by index 1388

Tabelle

Iteration, subsequent value 1379
Iteration-Initial 1378

Tabellenformat

enumeration object 1386

Table 205

active 1377
by index 1379
columns 325
editing 325
editing formats 327
modify column width 400
modifying table/diagram ratio 401
name 1374
number 1378
Number of Columns 1374
optimizing column width 229
ratio of the left table width to the
diagram width 741
ratio of the right table width to the
diagram width 765
see also
VcTable 1372
specifying 323
table formats 325

Table editing

extended 228

Table format

3D effect 1385
display of +/- 1381
field by index 1382

filter 1382
indentation column 1383
indentation width of text 1384
name 1384
number of table columns 1383
separation line color 1385
separation lines visible 1381

Table format field

fill pattern 1399
pattern color 1147, 1398

table rows

insert 402

TableByIndex

Method of
VcTableCollection 1379

TableByName

Method of
VcTableCollection 1379

TableCollection

Property of
VcGantt 772
see also
VcTableCollection 1377

TableColumnRanges

Property of
VcPrinter 1270

TableColumnWidthOptimizationAllowed

Property of
VcGantt 772

TableFormat

see also
VcTableFormat 1380

TableFormatCollection

Property of
VcTable 1375
see also

- VcTableFormatCollection 1387
- TableFormatField**
 - see also
 - VcTableFormatField 1391
- TableTimeScaleOnAllPages**
 - Property of
 - VcPrinter 1270
- TableWidthAdoptionFromViewOnScreen**
 - Property of
 - VcPrinter 1271
- TaskDataTableName**
 - Property of
 - VcResourceScheduler2 1330
- TaskDueDateFieldIndex**
 - Property of
 - VcResourceScheduler2 1331
- TaskPlanningStrategyFieldIndex**
 - Property of
 - VcResourceScheduler2 1331
- TaskPriorityFieldIndex**
 - Property of
 - VcResourceScheduler2 1332
- TaskQuantityFieldIndex**
 - Property of
 - VcResourceScheduler2 1333
- TaskReleaseDateFieldIndex**
 - Property of
 - VcResourceScheduler2 1334
- TaskResultEndDateFieldIndex**
 - Property of
 - VcResourceScheduler2 1335
- TaskResultPostEndDateFieldIndex**
 - Property of
 - VcResourceScheduler2 1335
- TaskResultPreparationStartDateFieldIndex**
 - Property of
 - VcResourceScheduler2 1336
- TaskResultProcessingStepFieldIndex**
 - Property of
 - VcResourceScheduler2 1336
- TaskResultProcessingTimeFieldIndex**
 - Property of
 - VcResourceScheduler2 1337
- TaskResultRouteFieldIndex**
 - Property of
 - VcResourceScheduler2 1338
- TaskResultStartDateFieldIndex**
 - Property of
 - VcResourceScheduler2 1338
- Text**
 - Property of
 - VcBoundingBox 459
 - VcDateLine 652
 - VcInterval 1057
- TextAlignment**
 - Property of
 - VcRibbon 1354
- TextAndGraphicsCombined**
 - Property of
 - VcTableFormatField 1403
- TextDataFieldIndex**
 - Property of
 - VcLayerFormatField 1120
 - VcLineFormatField 1151
 - VcTableFormatField 1403
- TextEntrySupplyingEventEnabled**
 - Property of
 - VcGantt 773
- TextFont**
 - Property of
 - VcBoundingBox 460
 - VcBoxFormatField 503

- VcLayerFormatField 1121
- VcTableFormatField 1403
- TextFontColor**
 - Property of
 - VcBoxFormatField 503
 - VcLayerFormatField 1121
 - VcLineFormatField 1151
 - VcTableFormatField 1404
- TextFontColorDataFieldIndex**
 - Property of
 - VcLayerFormatField 1121
 - VcLineFormatField 1152
 - VcTableFormatField 1404
- TextFontColorMapName**
 - Property of
 - VcLayerFormatField 1121
 - VcLineFormatField 1152
 - VcTableFormatField 1404
- TextFontDataFieldIndex**
 - Property of
 - VcLayerFormatField 1122
 - VcLineFormatField 1152
 - VcTableFormatField 1405
- TextFontMapName**
 - Property of
 - VcLayerFormatField 1122
 - VcLineFormatField 1153
 - VcTableFormatField 1405
- TextLineCount**
 - Property of
 - VcLayerFormatField 1122
 - VcLineFormatField 1153
- TextLineCountDataFieldIndex**
 - Property of
 - VcLayerFormatField 1123
- TextLineCountMapName**
 - Property of
- VcLayerFormatField 1123
- ThreeDEffect**
 - Property of
 - VcLayer 1103
 - VcNumericScale 1239
 - VcTableFormat 1385
 - VcTimeScale 1411
- TickColor**
 - Property of
 - VcNumericScale 1240
 - VcRibbon 1355
- TickPosition**
 - Property of
 - VcRibbon 1355
- Time interval**
 - smallest 225
- Time scale 207, 403**
 - adjust 409, 1272
 - allow timescale rescale 230
 - by index 1416
 - dialog 403
 - editing 425
 - end 404
 - ribbons 209, 350
 - sections 208
 - show time scale dialog 230
 - start 403
 - start and end 208
- Time scheduling**
 - automatic 1359
- Time unit 225**
- Time unit width 209**
- TimeColumnEndDate**
 - Property of
 - VcPrinter 1271
- TimeColumnStartDate**
 - Property of

- VcPrinter 1272
- Time-critical operations**
 - Wait cursor 234
- Timescale**
 - editing 403
 - limiting its width 437
- TimeScale**
 - see also
 - VcTimeScale 1407
- TimeScaleAdjustment**
 - Property of
 - VcPrinter 1272
- TimeScaleByIndex**
 - Method of
 - VcTimeScaleCollection 1416
- TimeScaleByName**
 - Method of
 - VcTimeScaleCollection 1416
- TimeScaleCollection**
 - Property of
 - VcGantt 773
 - see also
 - VcTimeScaleCollection 1413
- TimeScaleDialogEnabled**
 - Property of
 - VcGantt 774
- TimeScaleEnd**
 - Property of
 - VcGantt 774
- TimeScaleRescalingAllowed**
 - Property of
 - VcGantt 775
- TimeScaleStart**
 - Property of
 - VcGantt 775
- TimeUnit**
 - Property of
 - VcCurve 575
 - VcGantt 776
 - VcInterval 1057
 - VcSection 1370
- TimeUnitsPerStep**
 - Property of
 - VcGantt 777
- Title**
 - Property of
 - VcNumericScale 1240
- ToleranceTimeOnASAPDueDates**
 - Property of
 - VcResourceScheduler2 1339
- ToleranceTimeOnJITReleaseDates**
 - Property of
 - VcResourceScheduler2 1339
- ToleranceTimeOnStartLockDates**
 - Property of
 - VcResourceScheduler2 1340
- Tool tip**
 - disappearance on click 778
 - duration of appearance 778
 - duration of change 777
 - time elapsed till appearance 778
- ToolTipChangeDuration**
 - Property of
 - VcGantt 777
- ToolTipDuration**
 - Property of
 - VcGantt 777
- ToolTipPointerDuration**
 - Property of
 - VcGantt 778
- Tooltips 231**
 - data field for text 238
 - during runtime 213
- ToolTipShowAfterClick**

Property of
VcGantt 778

ToolTipTextSupplyingEventEnabled
Property of
VcGantt 779

Top
Property of
VcLegendView 1129
VcRect 1276
VcWorldView 1436

TopActualValue
Property of
VcLegendView 1129
VcWorldView 1436

TopMargin
Property of
VcLayerFormatField 1123
VcTableFormatField 1405

TotalFloatDataFieldIndex
Property of
VcScheduler 1364

Tracking space
fill pattern 780
pattern color 783

TrackingSpaceBackgroundColor
Property of
VcGantt 779

TrackingSpacePattern
Property of
VcGantt 780

TrackingSpacePatternColor
Property of
VcGantt 783

Tree view style 324

TruncatedTextSuppressed
Property of
VcLayerFormatField 1124

TurningAnnotationEnabled
Property of
VcDateLine 653
VcDateLineGrid 672

Type
Property of
VcBoundingBox 461
VcBoxFormatField 504
VcCalendar 507
VcCalendarProfile 546
VcCurve 575
VcDataDefinitionField 598
VcDataTableField 636
VcInterval 1057
VcMap 1183
VcRibbon 1355
VcTableFormatField 1405
VcUpdateBehaviorContext 1428

U

Unicode 214

Unit
Property of
VcDateLineGrid 673
VcNumericScale 1241

UnitEx
Property of
VcNumericScale 1241

UnitLabel
Property of
VcNumericScale 1241

UnitSeparation
Property of
VcRibbon 1356

UnitsPerStep
Property of
VcCurve 576

UnitWidth

- Property of
 - VcNumericScale 1242
 - VcSection 1371

UnitWidthEx

- Property of
 - VcSection 1371

Update

- Method of
 - VcBoxCollection 484
 - VcCalendar 513
 - VcCalendarCollection 520
 - VcCalendarGridCollection 544
 - VcCalendarProfileCollection 552
 - VcDataRecord 611
 - VcDataRecordCollection 620
 - VcDataTableCollection 630
 - VcDateLineCollection 662
 - VcDateLineGridCollection 682
 - VcGroup 980
 - VcGroupLevelLayoutCollection 1013
 - VcIntervalCollection 1064
 - VcLayerCollection 1113
 - VcLegendView 1132
 - VcLink 1158
 - VcLinkAppearanceCollection 1174
 - VcMapCollection 1193
 - VcNode 1215

Update behavior

- editable 1417, 1419

UpdateBehavior

- see also
 - VcUpdateBehavior 1417

UpdateBehaviorByIndex

- Method of
 - VcUpdateBehaviorCollection 1425

UpdateBehaviorByName

- Method of
 - VcUpdateBehaviorCollection 1426

UpdateBehaviorCollection

- Property of
 - VcGantt 783
- see also
 - VcUpdateBehaviorCollection 1420

UpdateBehaviorContext

- see also
 - VcUpdateBehaviorContext 1427

UpdateBehaviorName

- Property of
 - VcBox 473
 - VcCurve 577
 - VcDateLine 653
 - VcNode 1210
 - VcNumericScale 1242
 - VcTable 1375
 - VcTimeScale 1411
 - VcWorldView 1437

UpdateLinkRecord

- Method of
 - VcGantt 834

UpdateMode

- Property of
 - VcUpdateBehaviorContext 1429

UpdateNodeRecord

- Method of
 - VcGantt 834

UpdateRowNumberFields

- Method of
 - VcGantt 835

UsedAsOverlapLayer

- Property of
 - VcLayer 1103

UseGraphicalAttributes

- Property of
 - VcInterval 1058
 - UseGraphicalAttributesOfIntervals**
 - Property of
 - VcCalendarGrid 535
 - UseHigherDiagramHistogramHeightRatioPrecision**
 - Property of
 - VcGantt 784
 - UseHigherTableDiagramWidthRatioPrecision**
 - Property of
 - VcGantt 784
 - User account**
 - control not working 446
 - UseReferenceDate**
 - Property of
 - VcDateLineGrid 673
 - VcInfoWindow 1043
 - VcRibbon 1357
 - UseSnapTargetsInInteractions**
 - Property of
 - VcGantt 785
 - UseTwinLineSashPhantom**
 - Property of
 - VcGantt 785
- ## V
- ValencyDataFieldIndex**
 - Property of
 - VcCurve 577
 - VARCHART XGantt**
 - automatic scaling 25
 - placing in a form 25
 - VcBorderArea 452**
 - BorderBox 452
 - VcBorderBox 454**
 - Alignment 454
 - GraphicsFileName 455
 - LegendElementsArrangement 456
 - LegendElementsBottomMargin 456
 - LegendElementsMaximumColumnCount 456
 - LegendElementsMaximumRowCount 457
 - LegendElementsTopMargin 457
 - LegendFont 457
 - LegendTitle 458
 - LegendTitleFont 458
 - LegendTitleVisible 459
 - Text 459
 - TextFont 460
 - Type 461
 - VcBox 462**
 - AnchoringInteractionsAllowed 463
 - AnchoringLineVisible 463
 - AnchorToNode 473
 - FieldText 464
 - FormatName 464
 - GetActualExtent 474
 - GetTopLeftPixel 475
 - GetXYOffset 475
 - IdentifyFormatField 475
 - LineColor 465
 - LineThickness 465
 - LineType 466
 - Marked 468
 - Moveable 468
 - Name 469
 - NodeID 469
 - Origin 470
 - Priority 470
 - ReferencePoint 471
 - Resizing 471

- SetXYOffset 476
- SetXYOffsetByTopLeftPixel 476
- Specification 472
- UpdateBehaviorName 473
- Visible 473
- VcBoxClickingEventArgs**
 - Event argument objekt of
 - VcBoxLeftClicking 839
 - VcBoxLeftDoubleClicking 840
 - VcBoxRightClicking 843
- VcBoxCollection 478**
 - Add 479
 - AddBySpecification 479
 - BoxByIndex 480
 - BoxByName 481
 - Copy 481
 - Count 478
 - FirstBox 482
 - GetEnumerator 482
 - NextBox 483
 - Remove 483
 - Update 484
- VcBoxCreated**
 - Event of
 - VcGantt 838
- VcBoxCreatedEventArgs**
 - Event argument objekt of
 - VcBoxCreated 838
- VcBoxCreating**
 - Event of
 - VcGantt 838
- VcBoxCreatingEventArgs**
 - Event argument objekt of
 - VcBoxCreating 839
- VcBoxFormat 485**
 - CopyFormatField 488
 - FieldsSeparatedByLines 485
 - FormatField 486
 - FormatFieldCount 486
 - GetEnumerator 488
 - Name 487
 - RemoveFormatField 489
 - Specification 487
- VcBoxFormatCollection 490**
 - Add 491
 - AddBySpecification 491
 - Copy 492
 - Count 490
 - FirstFormat 492
 - FormatByIndex 493
 - FormatByName 493
 - GetEnumerator 494
 - NextFormat 494
 - Remove 495
- VcBoxFormatField 497**
 - Alignment 497
 - FormatName 498
 - GraphicsHeight 499
 - Index 499
 - MaximumTextLineCount 500
 - MinimumTextLineCount 500
 - MinimumWidth 501
 - PatternBackgroundColor 501
 - PatternColorAsARGB 502
 - TextFont 503
 - TextFontColor 503
 - Type 504
- VcBoxLeftClicking**
 - Event of
 - VcGantt 839
- VcBoxLeftDoubleClicking**
 - Event of
 - VcGantt 840
- VcBoxModified**

- Event of
 - VcGantt 841
- VcBoxModifiedEventArgs**
 - Event argument objekt of
 - VcBoxModified 841
- VcBoxModifying**
 - Event of
 - VcGantt 842
- VcBoxModifyingEventArgs**
 - Event argument objekt of
 - VcBoxModifying 842
- VcBoxRightClicking**
 - Event of
 - VcGantt 843
- VcCalendar 505**
 - AddDuration 508
 - CalcDuration 509
 - CalendarProfileCollection 506
 - Clear 509
 - GetEndOfPreviousWorktime 510
 - GetNextIntervalBorder 510
 - GetPreviousIntervalBorder 511
 - GetStartOfInterval 511
 - GetStartOfNextWorktime 512
 - IntervalCollection 506
 - IsWorktime 512
 - Name 506
 - SecondsPerWorkday 507
 - Specification 507
 - Type 507
 - Update 513
- VcCalendarCollection 514**
 - Active 514
 - Add 516
 - AddBySpecification 516
 - CalendarByIndex 517
 - CalendarByName 517
 - Copy 517
 - Count 515
 - FirstCalendar 518
 - GetEnumerator 518
 - NextCalendar 519
 - Remove 519
 - Update 520
- VcCalendarGrid 521**
 - BackgroundColor 522
 - BackgroundColorDataFieldIndex 523
 - BackgroundColorMapName 523
 - CalendarName 523
 - CalendarNameDataFieldIndex 524
 - CalendarNameMapName 524
 - EndSnapTarget 524
 - Identifiable 525
 - IdentifyInterval 536
 - LineColor 525
 - LineColorDataFieldIndex 526
 - LineColorMapName 526
 - LineThickness 526
 - LineType 527
 - Name 528
 - Pattern 528
 - PatternColor 531
 - PatternColorDataFieldIndex 532
 - PatternColorMapName 532
 - PatternDataFieldIndex 533
 - PatternMapName 533
 - Priority 533
 - Property of
 - VcPrinter 1273
 - SnapTarget 534
 - Specification 534
 - StartSnapTarget 535
 - UseGraphicalAttributesOfIntervals 535

- Visible 535
- VisibleDataFieldIndex 536
- VisibleMapName 536
- VcCalendarGridClickingEventArgs**
 - Event argument objekt of
 - VcCalendarGridRightClicking 844
- VcCalendarGridCollection 538**
 - Add 539
 - AddBySpecification 539
 - CalendarGridByIndex 540
 - CalendarGridByName 541
 - Copy 541
 - Count 538
 - FirstCalendarGrid 542
 - GetEnumerator 542
 - NextCalendarGrid 543
 - Remove 543
 - Update 544
- VcCalendarGridRightClicking**
 - Event of
 - VcGantt 844
- VcCalendarProfile 545**
 - IntervalCollection 545
 - Name 545
 - PutInOrderAfter 547
 - Specification 546
 - Type 546
- VcCalendarProfileCollection 548**
 - Add 549
 - AddBySpecification 549
 - CalendarProfileByIndex 550
 - CalendarProfileByName 550
 - Copy 550
 - Count 549
 - FirstCalendarProfile 551
 - NextCalendarProfile 551
 - Remove 551
- SelectCalendarProfiles 552
- Update 552
- VcComponentScrolled**
 - Event of
 - VcGantt 845
- VcComponentScrolledEventArgs**
 - Event argument objekt of
 - VcComponentScrolled 846
- VcComponentScrolling**
 - Event of
 - VcGantt 848
- VcComponentScrollingEventArgs**
 - Event argument objekt of
 - VcComponentScrolling 848
- VcCurve 554**
 - Addend 555
 - Clear 578
 - DeletePoint 579
 - FillReference1BackgroundColor 556
 - FillReference1Name 556
 - FillReference1Pattern 557
 - FillReference1PatternColor 561
 - FillReference2Color 561
 - FillReference2Name 562
 - FillReference2Pattern 562
 - FillReference2PatternColor 566
 - FilterName 567
 - GetFirstOverload 579
 - GetFirstOverloadEx 581
 - GetNextOverload 582
 - GetNextOverloadEx 583
 - GetValues 584
 - GetValuesEx 585
 - Histogram 567
 - LayerName 568
 - LineColor 568
 - LineThickness 569

- LineType 570
- Marked 571
- Name 572
- OverloadResultsCalendarName 572
- PointsEquidistant 573
- SetValues 586
- Source 573
- Specification 574
- StackReferenceName 574
- TimeUnit 575
- Type 575
- UnitsPerStep 576
- UpdateBehaviorName 577
- ValencyDataFieldIndex 577
- Visible 577
- VcCurveClickingEventArgs**
 - Event argument objekt of
 - VcCurveLeftClicking 851
 - VcCurveLeftDoubleClicking 852
 - VcCurveRightClicking 861
- VcCurveCollection 588**
 - Add 589
 - AddBySpecification 590
 - Copy 590
 - Count 588
 - CurveByIndex 591
 - CurveByName 591
 - FirstCurve 592
 - GetEnumerator 592
 - NextCurve 593
 - Remove 594
- VcCurveLeftClicking**
 - Event of
 - VcGantt 851
- VcCurveLeftDoubleClicking**
 - Event of
 - VcGantt 852
- VcCurveModified**
 - Event of
 - VcGantt 853
- VcCurveModifying**
 - Event of
 - VcGantt 853
- VcCurveModifyingEventArgs**
 - Event argument objekt of
 - VcCurveModifying 854
 - VcCurveModifyingEx 855
- VcCurveModifyingEx**
 - Event of
 - VcGantt 855
- VcCurvePointDeleting**
 - Event of
 - VcGantt 856
- VcCurvePointDeletingEventArgs**
 - Event argument objekt of
 - VcCurvePointDeleting 856
 - VcCurvePointDeletingEx 858
- VcCurvePointDeletingEx**
 - Event of
 - VcGantt 857
- VcCurvePointInserting**
 - Event of
 - VcGantt 858
- VcCurvePointInsertingEventArgs**
 - Event argument objekt of
 - VcCurvePointInserting 859
 - VcCurvePointInsertingEx 860
- VcCurvePointInsertingEx**
 - Event of
 - VcGantt 860
- VcCurveRightClicking**
 - Event of
 - VcGantt 861
- VcDataDefinitionField 595**

- DateFormat 595
- Editable 596
- Hidden 597
- Index 597
- Name 598
- Type 598
- VcDataDefinitionTable 600**
 - Count 600
 - CreateDataDefinitionField 601
 - DataDefinitionFieldByIndex 602
 - DataDefinitionFieldByName 602
 - FirstDataDefinitionField 603
 - GetEnumerator 604
 - NextDataDefinitionField 604
- VcDataModified**
 - Event of
 - VcGantt 862
- VcDataModifiedEventArgs**
 - Event argument objekt of
 - VcDataModified 863
- VcDataRecord 606**
 - AllData 606
 - DataField 607
 - DataTableName 608
 - Delete 609
 - ID 609
 - IdentifyObject 610
 - RelatedDataRecord 611
 - Update 611
- VcDataRecordCollection 613**
 - Add 614
 - Count 613
 - DataRecordByID 616
 - FirstDataRecord 616
 - GetEnumerator 617
 - GetNewUniqueID 618
 - NextDataRecord 618
 - Remove 619
 - Update 620
- VcDataRecordCreated**
 - Event of
 - VcGantt 863
- VcDataRecordCreatedEventArgs**
 - Event argument objekt of
 - VcDataRecordCreated 863
- VcDataRecordCreating**
 - Event of
 - VcGantt 864
- VcDataRecordCreatingEventArgs**
 - Event argument objekt of
 - VcDataRecordCreating 865
- VcDataRecordDeleted**
 - Event of
 - VcGantt 865
- VcDataRecordDeletedEventArgs**
 - Event argument objekt of
 - VcDataRecordDeleted 866
- VcDataRecordDeleting**
 - Event of
 - VcGantt 866
- VcDataRecordDeletingEventArgs**
 - Event argument objekt of
 - VcDataRecordDeleting 866
- VcDataRecordModified**
 - Event of
 - VcGantt 867
- VcDataRecordModifiedEventArgs**
 - Event argument objekt of
 - VcDataRecordModified 867
- VcDataRecordModifying**
 - Event of
 - VcGantt 868
- VcDataRecordModifyingEventArgs**
 - Event argument objekt of

- VcDataRecordModifying 868
- VcDataRecordNotFound**
 - Event of
 - VcGantt 869
- VcDataRecordNotFoundEventArgs**
 - Event argument objekt of
 - VcDataRecordNotFound 869
- VcDataTable 622**
 - DataRecordCollection 622
 - DataTableFieldCollection 623
 - Description 623
 - MultiplePrimaryKeysAllowed 624
 - Name 624
- VcDataTableCollection 625**
 - Add 626
 - Copy 626
 - Count 625
 - DataTableByIndex 627
 - DataTableByName 628
 - FirstDataTable 628
 - GetEnumerator 629
 - NextDataTable 629
 - Update 630
- VcDataTableField 631**
 - DataTableName 631
 - DateFormat 632
 - Editable 633
 - Hidden 633
 - Index 634
 - Name 634
 - PrimaryKey 635
 - RelationshipFieldIndex 635
 - Type 636
- VcDataTableFieldCollection 638**
 - Add 639
 - Copy 639
 - Count 638
- DataTableFieldByIndex 640
- DataTableFieldByName 641
- FirstDataTableField 641
- GetEnumerator 642
- NextDataTableField 642
- VcDateLine 644**
 - AlwaysCurrentDate 645
 - Date 645
 - DateDataFieldIndex 646
 - Font 646
 - FontColor 646
 - Identifiable 647
 - LabelPosition 647
 - LineColor 647
 - LineThickness 648
 - LineType 649
 - Movable 650
 - Name 650
 - Priority 651
 - PutInOrderAfter 654
 - SnapTarget 651
 - Specification 652
 - Text 652
 - TurningAnnotationEnabled 653
 - UpdateBehaviorName 653
 - Visible 653
 - VisibleDataFieldIndex 654
 - VisibleMapName 654
- VcDateLineClickingEventArgs**
 - Event argument objekt of
 - VcDateLineRightClicking 870
- VcDateLineCollection 656**
 - Add 657
 - AddBySpecification 657
 - Copy 658
 - Count 656
 - DateLineByIndex 659

- DateLineByName 659
- FirstDateLine 660
- GetEnumerator 661
- NextDateLine 661
- Remove 662
- Update 662
- VcDateLineGrid 664**
 - AdjustToReferenceDate 665
 - AnnotationAtBottom 665
 - AnnotationAtCenter 665
 - AnnotationAtTop 666
 - FormatName 666
 - HorAlignment 666
 - LineColor 667
 - LineColorDataFieldIndex 667
 - LineColorMapName 667
 - LineThickness 668
 - LineType 669
 - ObserveDST 670
 - Period 670
 - Priority 671
 - ReferenceDate 672
 - SnapTarget 672
 - TurningAnnotationEnabled 672
 - Unit 673
 - UseReferenceDate 673
 - Visible 674
 - VisibleDataFieldIndex 674
 - VisibleMapName 674
- VcDateLineGridCollection 676**
 - Add 677
 - AddBySpecification 677
 - Copy 678
 - Count 676
 - DateLineGridByIndex 679
 - DateLineGridByName 679
 - FirstDateLineGrid 680
 - GetEnumerator 680
 - NextDateLineGrid 681
 - Remove 681
 - Update 682
- VcDateLineModifying**
 - Event of
 - VcGantt 869
- VcDateLineModifyingEventArgs**
 - Event argument objek of
 - VcDateLineModifying 869
- VcDateLineRightClicking**
 - Event of
 - VcGantt 870
- VcDateShowing**
 - Event of
 - VcGantt 871
- VcDateShowingEventArgs**
 - Event argument objek of
 - VcDateShowing 871
- VcDiagramClickingEventArgs**
 - Event argument objek of
 - VcDiagramLeftClicking 876
 - VcDiagramLeftDoubleClicking 876
 - VcDiagramRightClicking 878
- VcDiagramHorizontalScrolled**
 - Event of
 - VcGantt 872
- VcDiagramHorizontalScrolledEventArgs**
 - Event argument objek of
 - VcDiagramHorizontalScrolled 872
- VcDiagramHorizontalScrolling**
 - Event of
 - VcGantt 873
- VcDiagramHorizontalScrollingEventArgs**
 - Event argument objek of

- VcDiagramHorizontalScrolling 874
- VcDiagramLeftClicking**
 - Event of
 - VcGantt 875
- VcDiagramLeftDoubleClicking**
 - Event of
 - VcGantt 876
- VcDiagramRightClicking**
 - Event of
 - VcGantt 877
- VcDragCompleting**
 - Event of
 - VcGantt 878
- VcDragCompletingEventArgs**
 - Event argument objekt of
 - VcDragCompleting 879
- VcDragEventArgs**
 - Event argument objekt of
 - VcDragOver 879
- VcDragOver**
 - Event of
 - VcGantt 879
- VcDragStarting**
 - Event of
 - VcGantt 880
- VcDragStartingEventArgs**
 - Event argument objekt of
 - VcDragStarting 880
- VcErrorOccurring**
 - Event of
 - VcGantt 880
- VcErrorOccurringEventArgs**
 - Event argument objekt of
 - VcErrorOccurring 881
- VcFieldSelecting**
 - Event of
 - VcGantt 881
- VcFieldSelectingEventArgs**
 - Event argument objekt of
 - VcFieldSelecting 881
- VcFilter 683**
 - AddSubCondition 686
 - CopySubCondition 687
 - DataDefinitionTable 684
 - DatesWithHourAndMinute 684
 - Evaluate 687
 - GetEnumerator 688
 - IsValid 688
 - Name 684
 - RemoveSubCondition 689
 - Specification 685
 - StringsCaseSensitive 685
 - SubCondition 686
 - SubConditionCount 686
- VcFilterCollection 690**
 - Add 691
 - AddBySpecification 692
 - Copy 692
 - Count 690
 - FilterByIndex 693
 - FilterByName 693
 - FirstFilter 693
 - GetEnumerator 694
 - MarkedNodesFilter 691
 - NextFilter 694
 - Remove 695
- VcFilterSubCondition 696**
 - ComparisonValueAsString 696
 - ConnectionOperator 697
 - DataFieldIndex 697
 - FilterName 698
 - GetEnumerator 700
 - Index 698
 - IsValid 700

- Operator 699
- VcGantt 701**
 - ActiveNodeFilter 710
 - AllLayersMovingTogether 710
 - AllLayersMovingTogetherAlways 711
 - Arrangement 711
 - ArrowKeyMode 712
 - ArrowKeyStepSizeMultiplier 713
 - BorderArea 714
 - BoxCollection 715
 - BoxCreationAllowed 715
 - BoxFormatCollection 715
 - CalendarCollection 716
 - CalendarGridCollection 716
 - CalendarProfileCollection 717
 - ConsiderLinkRelationTypesOnNodeDragging 717
 - ContextMenuForBoxesEnabled 717
 - ConvertDistance 788
 - CtrlCXVProcessingEnabled 718
 - DataDefinition 718
 - DataTableCollection 719
 - DateLineCollection 719
 - DateOutputFormat 720
 - DeleteLinkRecord 789
 - DeleteNodeRecord 789
 - DetectDataTableFieldName 790
 - DetectDataTableName 790
 - DetectFieldIndex 791
 - DiagramAlternatingRowBackgroundColor 721
 - DiagramBackgroundColor 722
 - DiagramHistogramHeightRatio 722
 - DiagramHistogramHeightRatioEx 723
 - DiagramVisible 723
 - DialogFont 723
 - DirectDataWritingModeEnabled 724
 - DoubleOutputFormat 724
 - DumpConfiguration 791
 - Enabled 725
 - EndDateForAutomaticScheduling 725
 - EndLoading 792
 - EventsSecurityCheck 726
 - ExportGraphicsToFileEx 792
 - ExtendedDataTablesEnabled 726
 - ExtendedEditingBehavior 727
 - FilePath 727
 - FilterCollection 728
 - FitChartIntoView 795
 - FitHistogramsIntoView 795
 - FitRangeIntoView 796
 - FontAntiAliasingEnabled 728
 - GetAValueFromARGB 797
 - GetBValueFromARGB 797
 - GetCurrentComponentStart 798
 - GetCurrentViewDates 799
 - GetDate 800
 - GetDateAsString 800
 - GetGValueFromARGB 801
 - GetLinkByID 802
 - GetLinkByNodeIDs 802
 - GetNodeByID 803
 - GetRValueFromARGB 803
 - GetViewComponentSize 804
 - GroupCollection 729
 - GroupingDataFieldIndex 729
 - GroupingModificationsAllowed 730
 - GroupLevelLayoutCollection 730
 - GroupNodes 805
 - GroupOptimizationOnInteractionsEnabled 731
 - GroupSortingDataFieldIndex 731
 - GroupSortingOrder 732

- HierarchyDataFieldIndex 732
- HierarchyLevelLayout 733
- HistogramCollection 733
- HistogramSeparationLineColor 734
- HorizontalMovementWhileDraggingAllowed 734
- IdentifyField 806
- IdentifyLayerAt 807
- IdentifyObject 810
- IdentifyObjectAt 811
- ImportConfiguration 813
- InbuiltMouseCursorWhileDraggingEnabled 734
- InfoWindow 735
- InitializeForWebService 813
- InitialRowCount 735
- InPlaceEditingOnGroupsInDiagramEnabled 735
- InPlaceEditingOnGroupsInTableEnabled 736
- InPlaceEditingOnNodesInDiagramEnabled 736
- InPlaceEditingOnNodesInTableEnabled 737
- InsertLinkRecord 813
- InsertNodeRecord 814
- InteractionMode 738
- KeepingNodesTogetherDataFieldIndex 738
- KeyDown 836
- KeyPress 836
- KeyUp 837
- LayerCollection 739
- LayersWithNonWorkInterval 739
- LeavingControlWhileDraggingAllowed 740
- LeftTable 740
- LeftTableDiagramWidthRatio 741
- LeftTableDiagramWidthRatioEx 741
- LegendView 742
- LineFormatCollection 742
- LinkAppearanceCollection 743
- LinkCollection 743
- LinkPredecessorDataFieldIndex 743
- LinksDataTableName 745
- LinkSuccessorDataFieldIndex 746
- LinkTypeDataFieldIndex 747
- Load 815
- MakeARGB 816
- MapCollection 748
- MinimumRowHeight 748
- MouseProcessingEnabled 749
- MoveMode 749
- MovingLayersAsNodeWithShiftKeyAllowed 750
- MultipleBoxMarkingAllowed 750
- NodeCalendarNameDataFieldIndex 751
- NodeCollection 751
- NodeCreationAllowed 752
- NodeCreationAtDroppingEnabled 752
- NodeCreationViaDoubleclick 753
- NodeCreationWithDialog 753
- NodeDurationDataFieldIndex 753
- NodeEndDateDataFieldIndex 754
- NodeLevelLayout 754
- NodeRowNumberDataFieldIndex 754
- NodesDataTableName 755
- NodeSortingDataFieldIndex 756
- NodeSortingOrder 757
- NodeStartDateDataFieldIndex 757
- NodesUseCalendars 757
- NodeToolTipTextDataFieldIndex 758
- NumericScaleCollection 758
- NumericScaleRescalingAllowed 759
- OLEDragViaDiagram 759

- OLEDragViaTable 759
- OptimizeTimeScaleStartEnd 816
- OverlapLayerEnabled 760
- OverlapLayerName 760
- PanningModeAllowed 760
- PartialLoadThreshold 761
- PhantomDrawingWhileDraggingEnabled 762
- PhantomLayerHeight 763
- Printer 763
- PrintEx 817
- PrintToFile 818
- RecalculateAllStructureCodes 819
- Reset 819
- ResourceScheduler2 763
- RightTable 764
- RightTableDiagramWidthRatio 764
- RightTableDiagramWidthRatioEx 765
- RoundedLinkSlantsEnabled 765
- RowHeightReductionEnabled 766
- RowMargins 766
- Sash3DStyleEnabled 767
- SashThickness 767
- SaveAsEx 820
- ScheduleProject 820
- Scheduler 767
- ScrollComponentStartTo 821
- ScrollEventsEnabled 768
- ScrollToDate 822
- ScrollToGroupLine 822
- ScrollToNode 823
- ScrollToNodeLine 824
- SelectedNodesMovingTogether 768
- SelectedRowBackgroundColor 769
- SelectionViaRubberRectAllowed 769
- SetImageResource 825
- ShowAboutDialog 826
- ShowEditGroupDialog 827
- ShowExportGraphicsDialog 827
- ShowLinkEditDialog 828
- ShowNodeEditDialog 829
- ShowPageSetupDialog 830
- ShowPrintDialog 830
- ShowPrinterSetupDialog 830
- ShowPrintPreviewDialog 831
- ShowSnapLines 769
- ShowSnapMarkings 770
- SnapTargetNodesSelectionMode 770
- SortGroups 831
- SortNodes 831
- StartDateForAutomaticScheduling 770
- SubRowMargins 771
- SummaryBarsVisible 771
- SuspendUpdate 832
- TableCollection 772
- TableColumnWidthOptimizationAllowed 772
- TextEntrySupplyingEventEnabled 773
- TimeScaleCollection 773
- TimeScaleDialogEnabled 774
- TimeScaleEnd 774
- TimeScaleRescalingAllowed 775
- TimeScaleStart 775
- TimeUnit 776
- TimeUnitsPerStep 777
- ToolTipChangeDuration 777
- ToolTipDuration 777
- ToolTipPointerDuration 778
- ToolTipShowAfterClick 778
- ToolTipTextSupplyingEventEnabled 779
- TrackingSpaceBackgroundColor 779

- TrackingSpacePattern 780
- TrackingSpacePatternColor 783
- UpdateBehaviorCollection 783
- UpdateLinkRecord 834
- UpdateNodeRecord 834
- UpdateRowNumberFields 835
- UseHigherDiagramHistogramHeightRatioPrecision 784
- UseHigherTableDiagramWidthRatioPrecision 784
- UseSnapTargetsInInteractions 785
- UseTwinLineSashPhantom 785
- VcBoxCreated 838
- VcBoxCreating 838
- VcBoxLeftClicking 839
- VcBoxLeftDoubleClicking 840
- VcBoxModified 841
- VcBoxModifying 842
- VcBoxRightClicking 843
- VcCalendarGridRightClicking 844
- VcComponentScrolled 845
- VcComponentScrolling 848
- VcCurveLeftClicking 851
- VcCurveLeftDoubleClicking 852
- VcCurveModified 853
- VcCurveModifying 853
- VcCurveModifyingEx 855
- VcCurvePointDeleting 856
- VcCurvePointDeletingEx 857
- VcCurvePointInserting 858
- VcCurvePointInsertingEx 860
- VcCurveRightClicking 861
- VcDataModified 862
- VcDataRecordCreated 863
- VcDataRecordCreating 864
- VcDataRecordDeleted 865
- VcDataRecordDeleting 866
- VcDataRecordModified 867
- VcDataRecordModifying 868
- VcDataRecordNotFound 869
- VcDateLineModifying 869
- VcDateLineRightClicking 870
- VcDateShowing 871
- VcDiagramHorizontalScrolled 872
- VcDiagramHorizontalScrolling 873
- VcDiagramLeftClicking 875
- VcDiagramLeftDoubleClicking 876
- VcDiagramRightClicking 877
- VcDragCompleting 878
- VcDragOver 879
- VcDragStarting 880
- VcErrorOccurring 880
- VcFieldSelecting 881
- VcGroupDeleting 882
- VcGroupLeftClicking 883
- VcGroupLeftDoubleClicking 884
- VcGroupModified 885
- VcGroupModifying 885
- VcGroupRightClicking 887
- VcGroupsMarked 888
- VcGroupsMarking 889
- VcHelpRequested 890
- VcHistogramCurveNameShowingInMenu 891
- VcHistogramLeftClicking 892
- VcHistogramLeftDoubleClicking 893
- VcHistogramRightClicking 894
- VcHistogramsHeightChanged 895
- VcHistogramsHeightChanging 895
- VcHistogramsHeightChangingEx 896
- VcInPlaceEditorShowing 897
- VcInteractionEnded 899
- VcInteractionModeChanged 900
- VcInteractionModeChanging 901

- VcInteractionObjectChanged 902
- VcInteractionStarted 903
- VcLegendViewClosed 904
- VcLinkCreated 905
- VcLinkCreating 906
- VcLinkDeleted 907
- VcLinkDeleting 907
- VcLinksLeftClicking 908
- VcLinksLeftDoubleClicking 909
- VcLinksRightClicking 910
- VcNodeCreated 911
- VcNodeCreating 912
- VcNodeDeleted 913
- VcNodeDeleting 914
- VcNodeLeftClicking 915
- VcNodeLeftDoubleClicking 916
- VcNodeModified 917
- VcNodeModifiedEx 917
- VcNodeModifying 919
- VcNodeResizeStarting 920
- VcNodeRightClicking 921
- VcNodesMarked 922
- VcNodesMarking 923
- VcNumericScaleLeftClicking 924
- VcNumericScaleLeftDoubleClicking 925
- VcNumericScaleRescaling 926
- VcNumericScaleRightClicking 927
- VcObjectDrawing 928
- VcObjectDrawn 930
- VcResourceSchedulingProgressing 931
- VcResourceSchedulingWarning 932
- VcSashButtonClicked 934
- VcStatusLineTextShowing 935
- VcTableCaptionLeftClicking 936
- VcTableCaptionLeftDoubleClicking 937
- VcTableCaptionRightClicking 938
- VcTableColumnWidthChanged 939
- VcTableColumnWidthChanging 940
- VcTableColumnWidthOptimizing 941
- VcTableWidthChanging 941
- VcTableWidthChangingEx 942
- VcTextEntrySupplying 943
- VcTimeScaleEndModified 957
- VcTimeScaleLeftClicking 957
- VcTimeScaleLeftDoubleClicking 958
- VcTimeScaleModified 959
- VcTimeScaleRightClicking 959
- VcTimeScaleSectionRescaled 961
- VcTimeScaleSectionRescaledEx 961
- VcTimeScaleSectionRescaling 961
- VcTimeScaleSectionRescalingEx 962
- VcTimeScaleSectionStartModifying 963
- VcTimeScaleStartModified 965
- VcToolTipTextSupplying 965
- VcViewComponentsSizeModified 967
- VcWorldViewClosed 968
- VcZoomFactorModified 969
- VerticalNodeMovementAllowed 785
- VerticalNodeMovementViaTableAllowed 786
- ViewComponentsBackgroundColor 786
- ViewComponentsBorderColor 786
- WaitCursorEnabled 787
- WorldView 787
- Zoom 835
- ZoomFactor 787
- ZoomingPerMouseWheelAllowed 788
- VcGroup 971**
 - BodyCollapsed 972

- DataField 972
- DataRecord 979
- Delete 979
- GroupingLevel 973
- GroupInvisible 974
- ID 974
- Marked 975
- Name 975
- NodeCollection 976
- NodesAndGroupsBelowCollapsed 976
- NodesInHeader 977
- NodesOverlaid 977
- RelatedDataRecord 980
- ReOptimizeNodes 980
- SubGroups 977
- SuperGroup 978
- Update 980
- Visible 978
- VcGroupClickingEventArgs**
 - Event argument objekt of
 - VcGroupLeftClicking 883
 - VcGroupLeftDoubleClicking 884
 - VcGroupRightClicking 888
- VcGroupCollection 982**
 - Count 982
 - FirstGroup 983
 - GetEnumerator 983
 - GroupByName 984
 - NextGroup 984
 - SelectGroups 985
- VcGroupDeleting**
 - Event of
 - VcGantt 882
- VcGroupDeletingEventArgs**
 - Event argument objekt of
 - VcGroupDeleting 882
- VcNodeResizeStarting 920
- VcGroupLeftClicking**
 - Event of
 - VcGantt 883
- VcGroupLeftDoubleClicking**
 - Event of
 - VcGantt 884
- VcGroupLevelLayout 986**
 - AutoCollapseGroups 987
 - AutoExpandTargetGroup 988
 - BodiesCollapsed 988
 - BodiesCollapsedDataFieldIndex 988
 - BodiesCollapsedMapName 989
 - CalendarGridName 989
 - CalendarGridsVisible 989
 - CalendarGridsWithChildGroups 989
 - CalendarNameDataFieldIndex 990
 - DateLineGridName 990
 - DateLineGridsVisible 990
 - DateLineGridsWithChildGroups 991
 - DateLineName 991
 - DateLinesVisible 991
 - DateLinesWithChildGroups 991
 - GroupDataFieldIndex 992
 - GroupNodesVisible 992
 - GroupsInvisible 992
 - GroupsInvisibleCollapsedMapName 993
 - GroupsInvisibleDataFieldIndex 993
 - Level 993
 - ModificationsAllowed 993
 - MovingGroupsVerticallyViaDiagramAllowed 994
 - MovingGroupsVerticallyViaTableAllowed 994
 - Name 994
 - NodesInHeaders 995
 - NodesOverlaid 995

- OptimizedNodesSortDataFieldIndex 995
- OptimizedNodesSortOrder 996
- OverlaidNodesSortDataFieldIndex 996
- OverlaidNodesSortOrder 996
- PagebreakMode 997
- RestoreAutoCollapsedGroups 997
- RestoreAutoExpandedGroups 997
- RowBackColorAsARGB 998
- RowBackColorDataFieldIndex 998
- RowBackColorMapName 998
- RowPattern 999
- RowPatternColorAsARGB 1002
- RowPatternColorDataFieldIndex 1002
- RowPatternColorMapName 1002
- RowPatternDataFieldIndex 1003
- RowPatternMapName 1003
- SeparationLineColor 1003
- SeparationLineColorDataFieldIndex 1004
- SeparationLineColorMapName 1004
- SeparationLinesVisible 1004
- SeparationLinesVisibleAtTop 1005
- SeparationLineThickness 1005
- SeparationLineType 1006
- SortDataFieldIndex 1007
- SortOrder 1007
- Specification 1008
- SummaryBarsVisible 1008
- Visible 1008
- VcGroupLevelLayoutCollection 1009**
 - Add 1010
 - AddBySpecification 1010
 - Copy 1011
 - Count 1009
 - FirstGroupLevelLayout 1011
 - GetEnumerator 1011
 - GroupLevelLayoutByIndex 1012
 - GroupLevelLayoutByName 1012
 - NextGroupLevelLayout 1012
 - Remove 1013
 - Update 1013
- VcGroupModified**
 - Event of
 - VcGantt 885
- VcGroupModifiedEventArgs**
 - Event argument object of
 - VcGroupModified 885
- VcGroupModifying**
 - Event of
 - VcGantt 885
- VcGroupModifyingEventArgs**
 - Event argument object of
 - VcGroupModifying 886
- VcGroupRightClicking**
 - Event of
 - VcGantt 887
- VcGroupsMarked**
 - Event of
 - VcGantt 888
- VcGroupsMarkedEventArgs**
 - Event argument object of
 - VcGroupsMarked 889
- VcGroupsMarking**
 - Event of
 - VcGantt 889
- VcHelpRequested**
 - Event of
 - VcGantt 890
- VcHelpRequestedEventArgs**
 - Event argument object of
 - VcHelpRequested 890
- VcHierarchyLevelLayout 1014**

- AutoCollapseGroups 1014
- AutoExpandTargetGroup 1015
- BodiesCollapsed 1015
- BodiesCollapsedDataFieldIndex 1015
- BodiesCollapsedMapName 1016
- HierarchyDataFieldIndex 1016
- LevelMaximumForPagebreaks 1016
- NodeSeparationLinesVisible 1016
- NodesInHeaders 1017
- NodesOverlaid 1017
- PagebreakMode 1017
- RestoreAutoCollapsedGroups 1018
- RestoreAutoExpandedGroups 1018
- SeparationLineColor 1018
- SeparationLinesVisible 1019
- SeparationLineThickness 1019
- SeparationLineType 1020
- SummaryBarsVisible 1020
- VcHistogram 1021**
 - CalendarGridsVisible 1022
 - CalendarName 1022
 - CurveCollection 1022
 - FitRangeIntoView 1029
 - GetActualScaleValues 1030
 - GetCurrentYValues 1030
 - Name 1023
 - NominalScaleMaximum 1023
 - NominalScaleMinimum 1024
 - NumericScaleCollection 1024
 - PutInOrderAfter 1030
 - RowBackColorAsARGB 1025
 - RowPattern 1025
 - RowPatternColorAsARGB 1028
 - ScrollToValue 1031
 - Visible 1028
- VcHistogramClickingEventArgs**
- Event argument objekt of
 - VcHistogramLeftClicking 892
 - VcHistogramLeftDoubleClicking 893
 - VcHistogramRightClicking 894
- VcHistogramCollection 1032**
 - Active 1032
 - Count 1033
 - CreateHistogram 1033
 - Delete 1034
 - FirstHistogram 1034
 - GetEnumerator 1035
 - HistogramByIndex 1035
 - HistogramByName 1035
 - NextHistogram 1036
- VcHistogramCurveNameShowingInMenu**
 - Event of
 - VcGantt 891
- VcHistogramCurveNameShowingInMenuEventArgs**
 - Event argument objekt of
 - VcHistogramCurveNameShowingInMenu 891
- VcHistogramLeftClicking**
 - Event of
 - VcGantt 892
- VcHistogramLeftDoubleClicking**
 - Event of
 - VcGantt 893
- VcHistogramRightClicking**
 - Event of
 - VcGantt 894
- VcHistogramsHeightChanged**
 - Event of
 - VcGantt 895
- VcHistogramsHeightChangedEventArgs**

- Event argument objekt of
 - VcHistogramsHeightChanged 895
- VcHistogramsHeightChanging**
 - Event of
 - VcGantt 895
- VcHistogramsHeightChangingEventArgs**
 - Event argument objekt of
 - VcHistogramsHeightChanging 896
- VcHistogramsHeightChangingEx**
 - Event of
 - VcGantt 896
- VcHistogramsHeightChangingExEventArgs**
 - Event argument objekt of
 - VcHistogramsHeightChangingEx 897
- VcInfoWindow 1037**
 - OutputFormatForCenterDate 1037
 - OutputFormatForDuration 1039
 - OutputFormatForEndDate 1040
 - OutputFormatForStartDate 1041
 - ReferenceDate 1043
 - UseReferenceDate 1043
 - Visible 1043
- VcInPlaceEditorShowing**
 - Event of
 - VcGantt 897
- VcInPlaceEditorShowingEventArgs**
 - Event argument objekt of
 - VcInPlaceEditorShowing 897
- VcInteractionEnded**
 - Event of
 - VcGantt 899
- VcInteractionEndedEventArgs**
 - Event argument objekt of
 - VcInteractionEnded 899
- VcInteractionModeChanged**
 - Event of
 - VcGantt 900
- VcInteractionModeChangedEventArgs**
 - Event argument objekt of
 - VcInteractionModeChanged 901
- VcInteractionModeChanging**
 - Event of
 - VcGantt 901
- VcInteractionModeChangingEventArgs**
 - Event argument objekt of
 - VcInteractionModeChanging 901
- VcInteractionObjectChanged**
 - Event of
 - VcGantt 902
- VcInteractionObjectChangedEventArgs**
 - Event argument objekt of
 - VcInteractionObjectChanged 902
- VcInteractionStarted**
 - Event of
 - VcGantt 903
- VcInteractionStartedEventArgs**
 - Event argument objekt of
 - VcInteractionStarted 903
- VcInterval 1044**
 - BackgroundColor 1046
 - CalendarProfileName 1046
 - DayInEndMonth 1046
 - DayInStartMonth 1047
 - Duration 1047
 - EndTime 1047
 - EndMonth 1048
 - EndTime 1048
 - EndWeekday 1048
 - LineColor 1049
 - LineThickness 1049

- LineType 1050
- Name 1051
- Pattern 1051
- PatternColor 1055
- PutInOrderAfter 1058
- Specification 1055
- StartDateTime 1055
- StartMonth 1056
- StartTime 1056
- StartWeekday 1056
- Text 1057
- TimeUnit 1057
- Type 1057
- UseGraphicalAttributes 1058
- VcIntervalCollection 1060**
 - Add 1061
 - AddBySpecification 1061
 - Copy 1062
 - Count 1061
 - FirstInterval 1062
 - IntervalByIndex 1063
 - IntervalByName 1063
 - NextInterval 1063
 - Remove 1064
 - Update 1064
- VcLayer 1065**
 - BackgroundColor 1067
 - BackgroundColorDataFieldIndex 1067
 - BackgroundColorMapName 1069
 - CalculateCurrentWidth 1106
 - CompletionDataFieldIndex 1070
 - DurationDataFieldIndex 1070
 - EndDataFieldIndex 1071
 - EndSnapTarget 1071
 - FilterName 1071
 - Format 1072
 - GraphicsFileName 1072
 - GraphicsFileNameDataFieldIndex 1074
 - GraphicsFileNameMapName 1075
 - Height 1077
 - HeightDataFieldIndex 1077
 - HeightMapName 1078
 - HorizontalOffset 1079
 - LabelSizeDependence 1079
 - LegendText 1080
 - LineColor 1080
 - LineColorDataFieldIndex 1080
 - LineColorMapName 1081
 - MaximumEndDataFieldIndex 1081
 - MinimumStartDataFieldIndex 1081
 - Movable 1082
 - Name 1082
 - NonWorkIntervalBackgroundColor 1083
 - NonWorkIntervalBackgroundColorDataFieldIndex 1083
 - NonWorkIntervalBackgroundColorMapName 1083
 - NonWorkIntervalLineColor 1084
 - NonWorkIntervalLineColorDataFieldIndex 1084
 - NonWorkIntervalLineColorMapName 1084
 - NonWorkIntervalLineThickness 1085
 - NonWorkIntervalLineType 1085
 - NonWorkIntervalPattern 1086
 - NonWorkIntervalPatternColor 1089
 - NonWorkIntervalPatternColorDataFieldIndex 1089
 - NonWorkIntervalPatternColorMapName 1090
 - NonWorkIntervalPatternDataFieldIndex 1090
 - NonWorkIntervalPatternMapName 1090

- NonWorkIntervalShape 1091
- ObjectDrawEventsEnabled 1091
- Pattern 1091
- PatternColor 1094
- PatternColorDataFieldIndex 1095
- PatternColorMapName 1095
- PatternDataFieldIndex 1096
- PatternMapName 1097
- PutInOrderAfter 1106
- Shape 1099
- Sizeable 1101
- Specification 1102
- StartDataFieldIndex 1102
- StartSnapTarget 1102
- ThreeDEffect 1103
- UsedAsOverlapLayer 1103
- VerticalOffset 1104
- VerticalOffsetDataFieldIndex 1104
- VerticalOffsetMapName 1104
- Visible 1105
- VisibleInLegend 1105
- VcLayerCollection 1108**
 - Add 1109
 - AddBySpecification 1109
 - Copy 1110
 - Count 1108
 - FirstLayer 1110
 - GetEnumerator 1111
 - LayerByIndex 1111
 - LayerByName 1111
 - NextLayer 1112
 - Remove 1112
 - Update 1113
- VcLayerFormat 1114**
 - CopyFormatField 1115
 - FormatField 1114
 - FormatFieldCount 1115
 - GetEnumerator 1115
 - RemoveFormatField 1116
- VcLayerFormatField 1117**
 - Alignment 1118
 - BottomMargin 1118
 - CalculateLineCount 1124
 - ConstantText 1118
 - FormatName 1119
 - Index 1119
 - LeftMargin 1119
 - MinimumWidth 1119
 - Priority 1120
 - RightMargin 1120
 - TextDataFieldIndex 1120
 - TextFont 1121
 - TextFontColor 1121
 - TextFontColorDataFieldIndex 1121
 - TextFontColorMapName 1121
 - TextFontDataFieldIndex 1122
 - TextFontMapName 1122
 - TextLineCount 1122
 - TextLineCountDataFieldIndex 1123
 - TextLineCountMapName 1123
 - TopMargin 1123
 - TruncatedTextSuppressed 1124
- VcLegendView 1125**
 - Border 1125
 - BorderColor 1126
 - Height 1126
 - HeightActualValue 1127
 - Left 1127
 - LeftActualValue 1128
 - ScrollBarMode 1128
 - Top 1129
 - TopActualValue 1129
 - Update 1132
 - Visible 1130

- Width 1130
- WidthActualValue 1130
- WindowMode 1131
- VcLegendViewClosed**
 - Event of
 - VcGantt 904
- VcLegendViewClosedEventArgs**
 - Event argument objekt of
 - VcLegendViewClosed 905
- VcLineFormat 1133**
 - CopyFormatField 1135
 - FormatField 1133
 - FormatFieldCount 1134
 - Name 1134
 - RemoveFormatField 1135
 - Specification 1134
- VcLineFormatCollection 1136**
 - Add 1137
 - AddBySpecification 1137
 - Copy 1138
 - Count 1136
 - FirstFormat 1138
 - FormatByIndex 1139
 - FormatByName 1139
 - NextFormat 1140
 - Remove 1140
- VcLineFormatField 1142**
 - Alignment 1143
 - ConstantText 1143
 - DateOutputFormat 1143
 - FormatName 1145
 - Index 1145
 - PatternBackgroundColorAsARGB 1145
 - PatternBackgroundColorDataFieldIndex 1146
 - PatternBackgroundColorMapName 1146
- PatternColorAsARGB 1146
- PatternColorMapName 1147
- PatternEx 1147
- PatternExDataFieldIndex 1150
- PatternExMapName 1151
- TextDataFieldIndex 1151
- TextFontColor 1151
- TextFontColorDataFieldIndex 1152
- TextFontColorMapName 1152
- TextFontDataFieldIndex 1152
- TextFontMapName 1153
- TextLineCount 1153
- VcLink 1154**
 - AllData 1154
 - DataField 1155
 - DataRecord 1157
 - Delete 1157
 - ID 1156
 - PredecessorNode 1156
 - RelatedDataRecord 1158
 - SuccessorNode 1157
 - Update 1158
- VcLinkAppearance 1160**
 - FilterName 1160
 - LineColor 1161
 - LineThickness 1161
 - LineType 1163
 - Name 1164
 - PredecessorLayerName 1164
 - PredecessorPortSymbol 1165
 - PutInOrderAfter 1168
 - RoutingType 1165
 - SuccessorLayerName 1166
 - SuccessorPortSymbol 1167
 - Visible 1167
- VcLinkAppearanceCollection 1169**
 - Add 1170

- AddBySpecification 1171
- Copy 1171
- Count 1169
- FirstLinkAppearance 1171
- GetEnumerator 1172
- LinkAppearanceByIndex 1172
- LinkAppearanceByName 1173
- NextLinkAppearance 1173
- Remove 1174
- Update 1174
- VcLinkCollection 1176**
 - Count 1176
 - FirstLink 1177
 - GetEnumerator 1177
 - NextLink 1177
 - SelectLinks 1178
- VcLinkCreated**
 - Event of
 - VcGantt 905
- VcLinkCreatedEventArgs**
 - Event argument objekt of
 - VcLinkCreated 905
- VcLinkCreating**
 - Event of
 - VcGantt 906
- VcLinkCreatingEventArgs**
 - Event argument objekt of
 - VcLinkCreating 906
- VcLinkDeleted**
 - Event of
 - VcGantt 907
- VcLinkDeletedEventArgs**
 - Event argument objekt of
 - VcLinkDeleted 907
- VcLinkDeleting**
 - Event of
 - VcGantt 907
- VcLinkDeletingEventArgs**
 - Event argument objekt of
 - VcLinkDeleting 908
- VcLinksClickingEventArgs**
 - Event argument objekt of
 - VcLinksLeftClicking 909
 - VcLinksLeftDoubleClicking 910
 - VcLinksRightClicking 911
- VcLinksLeftClicking**
 - Event of
 - VcGantt 908
- VcLinksLeftDoubleClicking**
 - Event of
 - VcGantt 909
- VcLinksRightClicking**
 - Event of
 - VcGantt 910
- VcMap 1180**
 - ConsiderFilterEntries 1180
 - Count 1181
 - CreateEntry 1183
 - DeleteEntry 1184
 - FirstMapEntry 1185
 - GetEnumerator 1181
 - GetMapEntry 1185
 - Name 1181
 - NextMapEntry 1185
 - Specification 1182
 - Type 1183
- VcMapCollection 1187**
 - Add 1188
 - AddBySpecification 1188
 - Copy 1189
 - Count 1187
 - FirstMap 1189
 - GetEnumerator 1190
 - MapByIndex 1190

- MapByName 1190
- NextMap 1191
- Remove 1192
- SelectMaps 1192
- Update 1193
- VcMapEntry 1194**
 - Color 1194
 - DataFieldValue 1195
 - FontBody 1196
 - FontName 1196
 - FontSize 1197
 - GraphicsFileName 1197
 - LegendText 1199
 - Millimeter 1199
 - Number 1200
 - Pattern 1200
- VcNode 1204**
 - AllData 1205
 - DataField 1205
 - DataRecord 1210
 - Delete 1211
 - GetPositionInView 1211
 - ID 1206
 - IncomingLinks 1206
 - Marked 1207
 - NodeRowInView 1212
 - OutgoingLinks 1208
 - OutlineIndent 1212
 - OutlineOutdent 1213
 - RelatedDataRecord 1214
 - SetPositionInView 1214
 - SnapTargetMode 1209
 - SuperGroup 1209
 - Update 1215
 - UpdateBehaviorName 1210
- VcNodeClickingEventArgs**
 - Event argument objekt of
- VcNodeLeftClicking 915
- VcNodeLeftDoubleClicking 916
- VcNodeRightClicking 921
- VcNodeCollection 1216**
 - Count 1216
 - FirstNode 1217
 - GetEnumerator 1217
 - NextNode 1217
 - SelectNodes 1218
- VcNodeCreated**
 - Event of
 - VcGantt 911
- VcNodeCreatedEventArgs**
 - Event argument objekt of
 - VcNodeCreated 912
- VcNodeCreating**
 - Event of
 - VcGantt 912
- VcNodeCreatingEventArgs**
 - Event argument objekt of
 - VcNodeCreating 913
- VcNodeDeleted**
 - Event of
 - VcGantt 913
- VcNodeDeletedEventArgs**
 - Event argument objekt of
 - VcNodeDeleted 913
- VcNodeDeleting**
 - Event of
 - VcGantt 914
- VcNodeDeletingEventArgs**
 - Event argument objekt of
 - VcNodeDeleting 914
- VcNodeLeftClicking**
 - Event of
 - VcGantt 915
- VcNodeLeftDoubleClicking**

- Event of
 - VcGantt 916
- VcNodeLevelLayout 1220**
 - CalendarGridName 1221
 - CalendarGridsVisible 1221
 - DateLineName 1221
 - DateLinesVisible 1221
 - RowBackgroundColorAsARGB 1222
 - RowBackgroundColorDataFieldIndex 1222
 - RowBackgroundColorMapName 1222
 - RowPattern 1223
 - RowPatternColorAsARGB 1223
 - RowPatternColorDataFieldIndex 1223
 - RowPatternColorMapName 1224
 - RowPatternDataFieldIndex 1224
 - RowPatternMapName 1224
 - SeparationLineColor 1225
 - SeparationLineInterval 1225
 - SeparationLinesVisible 1225
 - SeparationLinesVisibleAtTop 1226
 - SeparationLineThickness 1226
 - SeparationLineType 1227
 - SortDataFieldIndex 1227
 - SortOrder 1227
- VcNodeModified**
 - Event of
 - VcGantt 917
- VcNodeModifiedEventArgs**
 - Event argument objekt of
 - VcNodeModified 917
- VcNodeModifiedEx**
 - Event of
 - VcGantt 917
- VcNodeModifiedExEventArgs**
 - Event argument objekt of
 - VcNodeModifiedEx 918
- VcNodeModifying**
 - Event of
 - VcGantt 919
- VcNodeModifyingEventArgs**
 - Event argument objekt of
 - VcNodeModifying 919
- VcNodeResizeStarting**
 - Event of
 - VcGantt 920
- VcNodeRightClicking**
 - Event of
 - VcGantt 921
- VcNodesMarked**
 - Event of
 - VcGantt 922
- VcNodesMarkedEventArgs**
 - Event argument objekt of
 - VcNodesMarked 922
- VcNodesMarking**
 - Event of
 - VcGantt 923
- VcNodesMarkingEventArgs**
 - Event argument objekt of
 - VcGroupsMarking 889
 - VcNodesMarking 923
- VcNumericScale 1229**
 - DoubleOutputFormat 1230
 - Font 1230
 - FontColor 1231
 - Histogram 1231
 - LineColor 1232
 - MajorTicks 1232
 - MajorTicksEx 1233
 - MinorTicks 1234
 - MinorTicksEx 1234
 - Name 1235

- PatternBackgroundColorAsARGB 1235
- PatternColorAsARGB 1235
- PatternEx 1236
- ThreeDEffect 1239
- TickColor 1240
- Title 1240
- Unit 1241
- UnitEx 1241
- UnitLabel 1241
- UnitWidth 1242
- UpdateBehaviorName 1242
- VcNumericScaleClickingEventArgs**
 - Event argument objekt of
 - VcNumericScaleLeftClicking 924
 - VcNumericScaleLeftDoubleClickin
g 925
 - VcNumericScaleRightClicking 927
- VcNumericScaleCollection 1244**
 - Active 1244
 - Count 1245
 - FirstNumericScale 1245
 - GetEnumerator 1246
 - NextNumericScale 1246
 - NumericScaleByIndex 1247
 - NumericScaleByName 1247
- VcNumericScaleLeftClicking**
 - Event of
 - VcGantt 924
- VcNumericScaleLeftDoubleClicking**
 - Event of
 - VcGantt 925
- VcNumericScaleRescaling**
 - Event of
 - VcGantt 926
- VcNumericScaleRescalingEventArgs**
 - Event argument objekt of
 - VcNumericScaleRescaling 926
- VcNumericScaleRightClicking**
 - Event of
 - VcGantt 927
- VcObjectDrawing**
 - Event of
 - VcGantt 928
- VcObjectDrawingEventArgs**
 - Event argument objekt of
 - VcObjectDrawing 929
- VcObjectDrawn**
 - Event of
 - VcGantt 930
- VcObjectDrawnEventArgs**
 - Event argument objekt of
 - VcObjectDrawn 930
- VcPrinter 1249**
 - AbsoluteBottomMarginInInches 1250
 - AbsoluteLeftMarginInCM 1251
 - AbsoluteLeftMarginInInches 1251
 - AbsoluteRightMarginInCM 1252
 - AbsoluteRightMarginInInches 1252
 - AbsoluteTopMarginInCM 1253
 - AbsoluteTopMarginInInches 1253
 - Alignment 1254
 - AllBorderBoxesShownOnCombinedC
ontrols 1254
 - CombiningControlsEnabled 1255
 - CurrentHorizontalPagesCount 1255
 - CurrentVerticalPagesCount 1256
 - CurrentZoomFactor 1256
 - CuttingMarks 1256
 - DateFormat 1257
 - DefaultPrinterName 1258
 - DiagramEnabled 1258, 1259
 - DocumentName 1259
 - FitToPage 1259

- FoldingMarksType 1260
- MarginsShownInInches 1262
- MaxHorizontalPagesCount 1262
- MaxVerticalPagesCount 1263
- Orientation 1263
- PageDescription 1264
- PageDescriptionString 1264
- PageFrame 1265
- PageNumberMode 1265
- PageNumbers 1266
- PagePaddingEnabled 1266
- PaperSize 1267
- PrintDate 1267
- PrinterName 1268
- PrintPreviewWithFirstPage 1268
- ReOptimizeNodesInGroupsEnabled 1269
- ScalingMode 1269
- TableColumnRanges 1270
- TableTimeScaleOnAllPages 1270
- TableWidthAdoptionFromViewOnScreen 1271
- TimeColumnEndDate 1271
- TimeColumnStartDate 1272
- TimeScaleAdjustment 1272
- VcCalendarGrid 1273
- ZoomFactorAsDouble 1273
- VcRect 1274**
 - Bottom 1274
 - Height 1274
 - Left 1275
 - Right 1276
 - Top 1276
 - Width 1277
- VcResourceScheduler2 1278**
 - AssignmentDataTableName 1281
 - AssignmentIsResultFieldIndex 1283
 - AssignmentIsVisibleFieldIndex 1283
 - AssignmentLoadOrConsumptionPerItemFieldIndex 1284
 - AssignmentMaximumLoadFieldIndex 1285
 - AssignmentMinimumLoadFieldIndex 1285
 - AssignmentMinimumMaximumLoadType 1286
 - AssignmentOperationIDFieldIndex 1287
 - AssignmentResourceIDFieldIndex 1287
 - AssignmentResourceSelectionStrategyFieldIndex 1288
 - BaseCalendarUsageForSupplementTimes 1289
 - BaseTimeUnit 1290
 - BaseTimeUnitsPerStep 1290
 - DataRecordEventsEnabled 1291
 - DefaultOperationMaximumInterruptionTime 1291
 - DefaultResourceCalendarName 1292
 - DetermineIDOfFirstOperationByTaskID 1341
 - DetermineIDOfLastOperationByTaskID 1342
 - FullUsageOfPlanningUnitsEnabled 1292
 - LinkDataTableName 1293
 - LinkDurationFieldIndex 1295
 - LinkPredecessorOperationIDFieldIndex 1295
 - LinkPredecessorTaskIDFieldIndex 1296
 - LinkSuccessorOperationIDFieldIndex 1297
 - LinkSuccessorTaskIDFieldIndex 1297
 - OperationDataTableName 1298

- OperationLoadPerItemFieldIndex 1300
- OperationMaximumInterruptionTimeFieldIndex 1300
- OperationMinimumSupplementTimeFieldIndex 1301
- OperationOverlapQuantityFieldIndex 1302
- OperationPostLoadFieldIndex 1304
- OperationPostOffsetFieldIndex 1304
- OperationPreparationLoadFieldIndex 1305
- OperationPreparationOffsetFieldIndex 1306
- OperationResultEndDateFieldIndex 1306
- OperationResultPostEndDateFieldIndex 1307
- OperationResultPreparationStartDateFieldIndex 1308
- OperationResultProcessingTimeFieldIndex 1308
- OperationResultSelectedTimingResourceIDFieldIndex 1309
- OperationResultStartDateFieldIndex 1309
- OperationResultStatusFieldIndex 1310
- OperationRouteFieldIndex 1311
- OperationSequenceNumberFieldIndex 1311
- OperationStartLockDateFieldIndex 1312
- OperationTaskIDFieldIndex 1313
- OperationWorkInProgressFieldIndex 1313
- PlanningEndDate 1314
- PlanningStartDate 1315
- PlanningStrategy 1316
- Process 1342
- ResourceCalendarNameFieldIndex 1317
- ResourceCapacityType 1317
- ResourceCapacityTypeFieldIndex 1318
- ResourceConstraintTypeFieldIndex 1319
- ResourceDataTableName 1320
- ResourceEfficiencyFieldIndex 1322
- ResourceGroupDataTableName 1323
- ResourceGroupIDFieldIndex 1324
- ResourceNameFieldIndex 1324
- ResourceResultLoadCurveNamePrefix 1325
- ResourceResultStockCurveNamePrefix 1326
- ResourceSelectionStrategy 1327
- ResourceType 1328
- ResultProcessingStepCount 1329
- TaskDataTableName 1330
- TaskDueDateFieldIndex 1331
- TaskPlanningStrategyFieldIndex 1331
- TaskPriorityFieldIndex 1332
- TaskQuantityFieldIndex 1333
- TaskReleaseDateFieldIndex 1334
- TaskResultEndDateFieldIndex 1335
- TaskResultPostEndDateFieldIndex 1335
- TaskResultPreparationStartDateFieldIndex 1336
- TaskResultProcessingStepFieldIndex 1336
- TaskResultProcessingTimeFieldIndex 1337
- TaskResultRouteFieldIndex 1338
- TaskResultStartDateFieldIndex 1338
- ToleranceTimeOnASAPDueDates 1339
- ToleranceTimeOnJITReleaseDates 1339

- ToleranceTimeOnStartLockDates 1340
- WorkInProcessType 1340
- WritingDebugFilesEnabled 1341
- VcResourceSchedulingProgressing**
 - Event of
 - VcGantt 931
- VcResourceSchedulingProgressingEventArgs**
 - Event argument objekt of
 - VcResourceSchedulingProgressing 931
- VcResourceSchedulingWarning**
 - Event of
 - VcGantt 932
- VcRibbon 1344**
 - CalendarName 1345
 - DateOutputFormat 1345
 - Font 1347
 - FontColor 1347
 - MajorTicks 1348
 - MinorTicks 1348
 - ObserveDST 1349
 - PatternBackgroundColorAsARGB 1349
 - PatternColorAsARGB 1350
 - PatternEx 1350
 - Position 1354
 - ReferenceDate 1354
 - TextAlignment 1354
 - TickColor 1355
 - TickPosition 1355
 - Type 1355
 - UnitSeparation 1356
 - UseReferenceDate 1357
- VcSashButtonClicked**
 - Event of
 - VcGantt 934
- VcSashButtonClickedEventArgs**
 - Event argument objekt of
 - VcSashButtonClicked 935
- VcScheduler 1358**
 - ActualEndDateDataFieldIndex 1359
 - ActualStartDateDataFieldIndex 1359
 - AutomaticSchedulingEnabled 1359
 - DurationDataFieldIndex 1360
 - EarlyEndDateDataFieldIndex 1360
 - EarlyStartDateDataFieldIndex 1360
 - EndDateForAutomaticScheduling 1360
 - EndDateNotLaterThanDataFieldIndex 1361
 - FreeFloatDataFieldIndex 1361
 - LateEndDateDataFieldIndex 1361
 - LateStartDateDataFieldIndex 1362
 - LinkDurationDataFieldIndex 1362
 - ScheduledProjectEndDate 1362
 - ScheduledProjectStartDate 1363
 - ScheduleProject 1364
 - ScheduleSuccessorsOnlyEnabled 1363
 - StartDateForAutomaticScheduling 1363
 - StartDateNotEarlierThanDataFieldIndex 1363
 - TotalFloatDataFieldIndex 1364
- VcSection 1366**
 - CalendarGrid 1366
 - DateLineGrid 1367
 - LineColor 1367
 - NonWorkIntervalsCollapsed 1368
 - Ribbon 1369
 - StartDate 1370
 - TimeUnit 1370
 - UnitWidth 1371
 - UnitWidthEx 1371

VcStatusLineTextShowing

Event of

VcGantt 935

VcStatusLineTextShowingEventArgs

Event argument objekt of

VcStatusLineTextShowing 935

VcTable 1372

ColumnTitle 1372

ColumnWidth 1373

IdentifyFormatField 1376

Name 1374

NoOfColumns 1374

OptimizeColumnWidth 1376

Position 1374

TableFormatCollection 1375

UpdateBehaviorName 1375

Visible 1375

VcTableCaptionClickingEventArgs

Event argument objekt of

VcTableCaptionLeftClicking 936

VcTableCaptionLeftDoubleClicking
937

VcTableCaptionRightClicking 938

VcTableCaptionLeftClicking

Event of

VcGantt 936

VcTableCaptionLeftDoubleClicking

Event of

VcGantt 937

VcTableCaptionRightClicking

Event of

VcGantt 938

VcTableCollection 1377

Active 1377

Count 1378

FirstTable 1378

GetEnumerator 1378

NextTable 1378

TableByIndex 1379

TableByName 1379

VcTableColumnWidthChanged

Event of

VcGantt 939

VcTableColumnWidthChangedEventArgs

Event argument objekt of

VcTableColumnWidthChanged
939

VcTableColumnWidthChanging

Event of

VcGantt 940

VcTableColumnWidthChangingEventArgs

Event argument objekt of

VcTableColumnWidthChanging
940

VcTableColumnWidthOptimizing

Event of

VcGantt 941

VcTableColumnWidthOptimizingEventArgs

Event argument objekt of

VcTableColumnWidthOptimizing
941

VcTableFormat 1380

CollapseColumn 1381

FieldsSeparatedByLines 1381

FilterName 1382

FormatField 1382

FormatFieldCount 1383

GetEnumerator 1386

IndentColumn 1383

IndentWidth 1384

Name 1384

SeparationLineColor 1385

ThreeDEffect 1385

VcTableFormatCollection 1387

Count 1387
 FirstFormat 1388
 FormatByIndex 1388
 FormatByName 1389
 GetEnumerator 1389
 NextFormat 1390

VcTableFormatField 1391

Alignment 1392
 BottomMargin 1392
 ConstantText 1393
 FormatName 1393
 GraphicsFileName 1393
 GraphicsFileNameDataFieldIndex 1394
 GraphicsFileNameMapName 1394
 GraphicsHeight 1395
 Index 1395
 LeftMargin 1395
 MaximumTextLineCount 1396
 MinimumTextLineCount 1396
 MultiState 1396
 PatternBackgroundColorAsARGB 1397
 PatternBackgroundColorDataFieldIndex 1397
 PatternBackgroundColorMapName 1397
 PatternColorAsARGB 1398
 PatternColorDataFieldIndex 1398
 PatternColorMapName 1398
 PatternEx 1399
 PatternExDataFieldIndex 1402
 PatternExMapName 1402
 RightMargin 1403
 TextAndGraphicsCombined 1403
 TextDataFieldIndex 1403
 TextFont 1403

TextFontColor 1404
 TextFontColorDataFieldIndex 1404
 TextFontColorMapName 1404
 TextFontDataFieldIndex 1405
 TextFontMapName 1405
 TopMargin 1405
 Type 1405

VcTableWidthChanging

Event of
 VcGantt 941

VcTableWidthChangingEventArgs

Event argument object of
 VcTableWidthChanging 942

VcTableWidthChangingEx

Event of
 VcGantt 942

VcTableWidthChangingExEventArgs

Event argument object of
 VcTableWidthChangingEx 943

VcTextEntrySupplying

Event of
 VcGantt 943

VcTextEntrySupplying event 231**VcTextEntrySupplyingEventArgs**

Event argument object of
 VcTextEntrySupplying 943

VcTimeScale 1407

BackgroundColor 1407
 CalendarGridsVisible 1408
 DateGridsVisible 1408
 Font 1409
 FontColor 1409
 Name 1409
 Ribbon 1410
 Section 1410
 ThreeDEffect 1411
 UpdateBehaviorName 1411

VcTimeScaleClickingEventArgs

Event argument objekt of

VcTimeScaleLeftClicking 958

VcTimeScaleLeftDoubleClicking
959

VcTimeScaleRightClicking 960

VcTimeScaleCollection 1413

Active 1413

Count 1414

FirstTimeScale 1414

GetEnumerator 1415

NextTimeScale 1415

TimeScaleByIndex 1416

TimeScaleByName 1416

VcTimeScaleEndModified

Event of

VcGantt 957

VcTimeScaleLeftClicking

Event of

VcGantt 957

VcTimeScaleLeftDoubleClicking

Event of

VcGantt 958

VcTimeScaleModified

Event of

VcGantt 959

VcTimeScaleRightClicking

Event of

VcGantt 959

VcTimeScaleSectionRescaled

Event of

VcGantt 961

VcTimeScaleSectionRescaledEx

Event of

VcGantt 961

VcTimeScaleSectionRescaling

Event of

VcGantt 961

VcTimeScaleSectionRescalingEventArgs

Event argument objekt of

VcTimeScaleSectionRescaling
962

VcTimeScaleSectionRescalingEx

Event of

VcGantt 962

VcTimeScaleSectionRescalingEventArgs

Event argument objekt of

VcTimeScaleSectionRescalingEx
963

VcTimeScaleSectionStartModifying

Event of

VcGantt 963

VcTimeScaleSectionStartModifyingEventArgs

Event argument objekt of

VcTimeScaleSectionStartModifying
964

VcTimeScaleStartModified

Event of

VcGantt 965

VcToolTipTextSupplying

Event of

VcGantt 965

VcToolTipTextSupplying event 231

VcToolTipTextSupplyingEventArgs

Event argument objekt of

VcToolTipTextSupplying 965

VcUpdateBehavior 1417

IsEditable 1417

Name 1418

PutInOrderAfter 1419

Specification 1418

VcUpdateBehaviorCollection 1420

Active 1420

- Add 1421
- AddBySpecification 1422
- Copy 1422
- Count 1421
- FirstUpdateBehavior 1423
- GetEnumerator 1424
- NextUpdateBehavior 1424
- Remove 1425
- UpdateBehaviorByIndex 1425
- UpdateBehaviorByName 1426
- VcUpdateBehaviorContext 1427**
 - DelayTime 1427
 - IsEditable 1428
 - Type 1428
 - UpdateMode 1429
- VcViewComponentsSizeModified**
 - Event of
 - VcGantt 967
- VcViewComponentsSizeModifiedEventArgs**
 - Event argument objekt of
 - VcViewComponentsSizeModified 967
- VcWorldView 1431**
 - Border 1431
 - BorderColor 1432
 - Height 1432
 - HeightActualValue 1433
 - Left 1433
 - LeftActualValue 1434
 - MarkingColor 1434
 - Mode 1435
 - ScrollBarMode 1435
 - Top 1436
 - TopActualValue 1436
 - UpdateBehaviorName 1437
 - Visible 1437
 - Width 1438
 - WidthActualValue 1438
- VcWorldViewClosed**
 - Event of
 - VcGantt 968
- VcWorldViewClosedEventArgs**
 - Event argument objekt of
 - VcWorldViewClosed 969
- VcZoomFactorModified**
 - Event of
 - VcGantt 969
- VcZoomFactorModifiedEventArgs**
 - Event argument objekt of
 - VcZoomFactorModified 969
- VerticalNodeMovementAllowed**
 - Property of
 - VcGantt 785
- VerticalNodeMovementViaTableAllowed**
 - Property of
 - VcGantt 786
- VerticalOffset**
 - Property of
 - VcLayer 1104
- VerticalOffsetDataFieldIndex**
 - Property of
 - VcLayer 1104
- VerticalOffsetMapName**
 - Property of
 - VcLayer 1104
- ViewComponentsBackgroundColor**
 - Property of
 - VcGantt 786
- ViewComponentsBorderColor**
 - Property of
 - VcGantt 786
- Viewer Metafile (*.vmf) 219**

Visible

Property of

- VcBox 473
- VcCalendarGrid 535
- VcCurve 577
- VcDateLine 653
- VcDateLineGrid 674
- VcGroup 978
- VcGroupLevelLayout 1008
- VcHistogram 1028
- VcInfoWindow 1043
- VcLayer 1105
- VcLegendView 1130
- VcLinkAppearance 1167
- VcTable 1375
- VcWorldView 1437

VisibleDataFieldIndex

Property of

- VcCalendarGrid 536
- VcDateLine 654
- VcDateLineGrid 674

VisibleInLegend

Property of

- VcLayer 1105

VisibleMapName

Property of

- VcCalendarGrid 536
- VcDateLine 654
- VcDateLineGrid 674

W**WaitCursorEnabled**

Property of

- VcGantt 787

Width

Property of

- VcLegendView 1130

VcRect 1277

VcWorldView 1438

Width ratio

table/complete diagram 248

Width ratio of table/diagram

more accurate method 248, 249

WidthActualValue

Property of

- VcLegendView 1130
- VcWorldView 1438

WindowMode

Property of

- VcLegendView 1131

WorkInProcessType

Property of

- VcResourceScheduler2 1340

World view 220, 418**WorldView**

name of UpdateBehavior 1437

Property of

- VcGantt 787

see also

- VcWorldView 1431

WritingDebugFilesEnabled

Property of

- VcResourceScheduler2 1341

X**xgx_con_diaEditBoxFormatE.gif 1033****Z****Zoom**

adjust the diagram to window size
while keeping the height-to-width-
ratio 795

Method of

- VcGantt 835

ZoomFactor

Property of

VcGantt 787

ZoomFactorAsDouble

Property of

VcPrinter 1273

Zooming 382

per mouse wheel 230

ZoomingPerMouseWheelAllowed

Property of

VcGantt 788