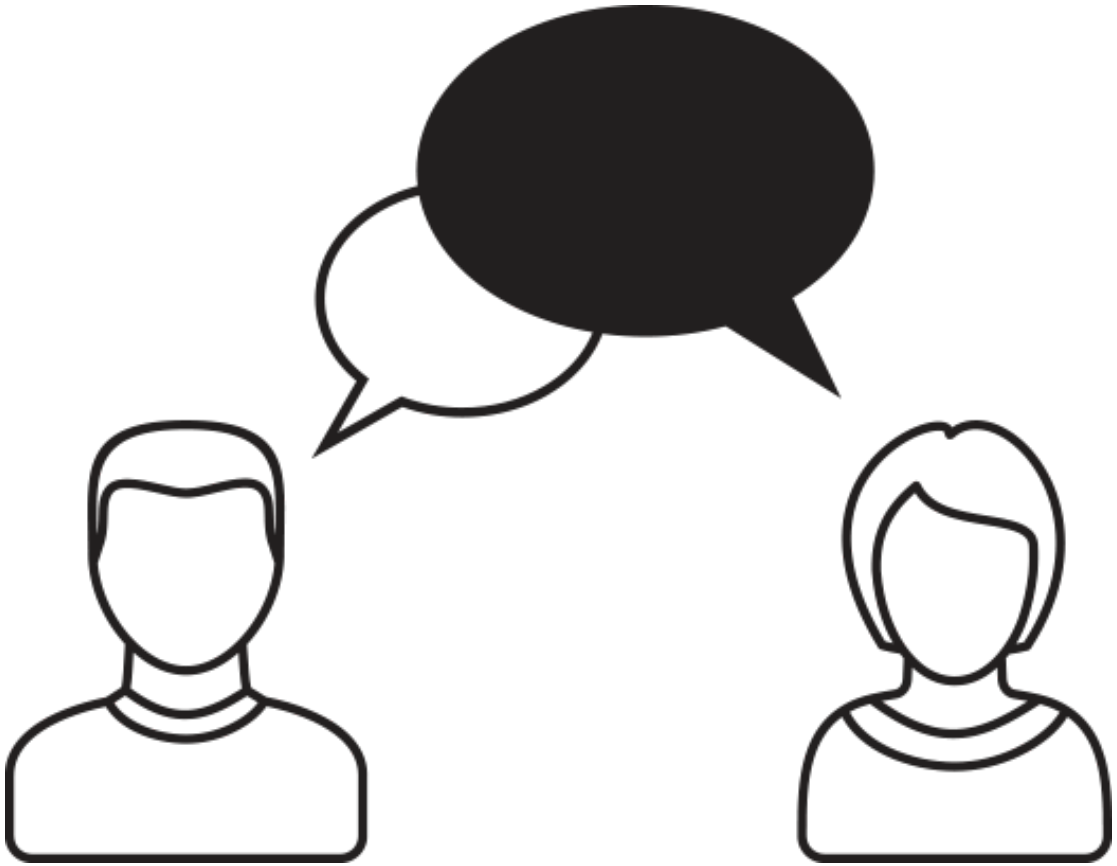

regoUniversity 2017
Gel Scripts |
Beginner
Your Guides: James Gille, Yogesh Renapure

Introductions

- Take 5 Minutes
- Turn to a Person Near You
- Introduce Yourself



Agenda

- What is GEL
- GEL Script Structure
- Commonly Used Tags
- Best Practices
- Example Use Cases
- Exercise

What Is GEL

- GEL stands for Generic Execution Language
- Based on Jelly, a jakarta.apache.org Commons project
- Extended and embedded into CA PPM to enable custom logic to solve business problems
- GEL is the basis for the enterprise application integration framework within CA PPM
- GEL scripts can be executed from within CA PPM by installing them as processes, or from the command line
- GEL supports the use of Java methods, SQL, and web services

What Is GEL

- Additional information can be found in the CA Documentation(Developer Guide) and at the Apache Jelly website at

  http://jakarta.apache.org/commons/jelly/index.html

GEL Script Structure

Header

```
<gel:script
    xmlns:core="jelly:core"
    xmlns:gel="jelly:com.niku.union.gel.GELTagLibrary"
    xmlns:sql="jelly:sql" >
```

Comment ⟶ <!-- CODE GOES HERE -->

Footer ⟶ </gel:script>

Note: An entire script always resides within the GEL script tag

GEL Script Structure - Tags

- A GEL script is built from qualified elements bound to java code called tags
- Using namespace declarations, tags are organized into tag libraries which are made available in a script
- CA PPM ships with many out of the box tag libraries

Hello World Example:

```
<gel:script xmlns:core="jelly:core"
            xmlns:gel="jelly:com.niku.union.gel.GELTagLibrary">
    <core:forEach indexVar='i' begin='1' end='3'>
            <gel:out>Hello World ${i}!</gel:out>
    </core:forEach>
</gel:script>
```

GEL Script Structure - Common Namespaces

xmlns:core="jelly:core"
xmlns:gel="jelly:com.niku.union.gel.GELTagLibrary"
xmlns:sql="jelly:sql"
xmlns:xog=http://www.niku.com/xog"
xmlns:soap="jelly:com.niku.union.gel.SOAPTagLibrary"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:file="jelly:com.niku.union.gel.FileTagLibrary"
xmlns:util="jelly:util"
xmlns:email="jelly:email"
xmlns:ftp="jelly:com.niku.union.gel.FTPTagLibrary"

GEL Script Structure - Variables

- Used extensively throughout GEL scripts
- Most tags can set variables
- Ex:
  - <core:set var="var_name">variable value</core:set>
  - <core:set var="var_name" value="variable value"/>
- Referencing a variable
  - ${var_name}
- Typically variables are valid for the current script only

- Conditionals / Loops
- Variables / Parameters
- Database Operations
- Logging
- Email

Commonly Used Tags

Conditionals / Loops

- **<core:if>**

```
<core:if test="${!empty sqlException}">
        …
</core:if>
```

- **<core:choose>**

```
<core:choose>
        <core:when test="${sessionID == null}">
                …
        </core:when>
        <core:otherwise>
                …
        </core:otherwise>
</core:choose>
```

Conditionals / Loops

- **<core:forEach>**

```
<core:forEach trim="true" items="${queryResult.rows}" var="row">
    …
</core:forEach>
```

- **<gel:forEach>**

```
<gel:forEach select="$projectsXML/NikuDataBus/Projects/Project" var="currentPrj">
    …
</gel:forEach>
```

- Two types of forEach loop types.
  - CORE version performs basic FOR looping, iterating over elements.
  - GEL version iterates over XML elements

Conditionals / Loops

- **<core:while>**

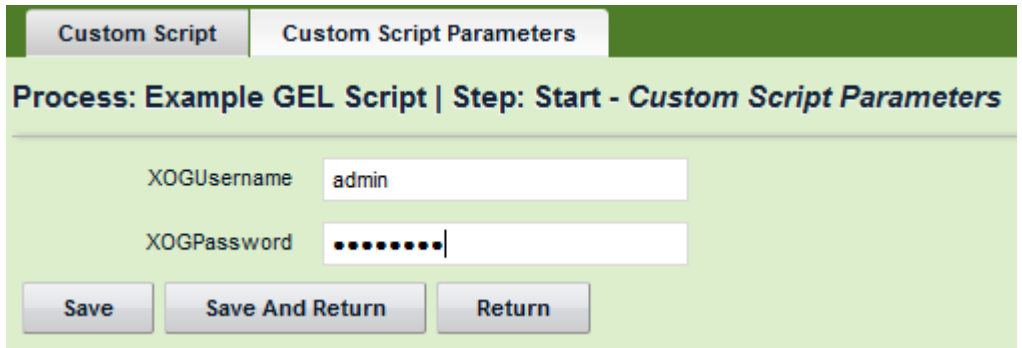
```
<core:while test="${condition == true}">
    …
</core:while>
```

Variables / Parameters - Built-in Parameters

- Custom Action GEL scripts associated with processes have the following parameters available to them:
  - **Object instance ID**
    - If no object is associated with the process, the ID is -1. Otherwise the ${gel_objectInstanceId} parameter contains the object instance ID.
  - **Process ID**
    - ${gel_processId} is the process identifier; all instances of this process share this identifier.
  - **Process instance ID**
    - ${gel_processInstanceId} is the process instance identifier; all instances have a unique value.
- All built-in parameters have a "gel_" prefix and are of data type - numeric.

Variables / Parameters

- **<gel:parameter>**
  - Allows values to be passed into a GEL script from a CA Clarity PPM process.
  - Refer to the parameter as you would any other variable, using the ${variablename} syntax.
  - Optional attribute secure="true" causes CA Clarity PPM to hide the actual value in the user interface with asterisks (*).
- Ex:
  - <gel:parameter var="XOGUsername" default="admin"/>
  - <gel:parameter var="XOGPassword" default="password" secure="true"/>



Variables / Parameters

- **<core:set>**
  - Used to set basic variables

```
<core:set value="1" var="yes"/>
<gel:out>${yes}</gel:out>
```

  - You can do some basic math on the variable:

```
<gel:out>${yes+2}</gel:out>
```

- **<gel:set>**
  - Use this tag when it is necessary to extract the value of the variable from an XML document.
  - Requires a select attribute which in turn requires an XPath statement.
  - Ex: the select statement points the way to the Statistics node of a XOG output file.

```
<gel:set asString="true" select="$result//XOGOutput/Statistics" var="xogStats"/>
```

Variables / Parameters

- **<gel:persist>** Allows you to set variables with a scope that extends beyond the current script.

- **<gel:parse>** Used to create an XML document in memory. This is how you will build XOG requests. The tag can be used to generate an entire XML document, or specific nodes that can later be attached into an existing XML document.

```
<gel:parse var="loadContent">
 <NikuDataBus xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
  xsi:noNamespaceSchemaLocation="../xsd/nikuxog_resource.xsd">
 <Header version="12.0.0.5028" action="write" objectType="resource“ externalSource=“PS"/>
  <Resources>
   <Resource resourceId="abc" isActive="true">
    <PersonalInformation lastName="doe" firstName="john" emailAddress="jdoe@ca.com"/>
   </Resource>
  </Resources>
 </NikuDataBus>
</gel:parse>
```

Variables / Parameters - Case Sensitive Issues

- Information contained within GEL tags is case sensitive. For example, if you declare a variable as follows:

```
<core:set var="v_ProjectID">PRJ-123456</core:set>
```

- Then reference the variable as follows:

```
<gel:out>${v_projectid}</gel:out>
```

- This will not output PRJ-123456. However, this will not cause an actual error to occur, so you can't "catch" this error.

Database Operations

- Datasources
- SQL Queries
- SQL Updates
- SQL Bind Variables / Parameters

Database Operations - Datasources

- **<gel:setDataSource/>**
  - Uses the connection properties from CA PPM's properties (set in the CSA)
  - On Premise installations can create additional external database definitions and reference them in scripts as well
  - Var attribute is optional. If not specified and only one datasource is set, then all SQL calls will use that datasource.

```
<gel:setDataSource dbId="Niku" var="clarityDS"/>
<sql:query dataSource="${clarityDS}" var="result">
    SELECT 1 from dual
</sql:query>
```

Database Operations - Datasources

- **<sql:setDataSource>**
  - Reasons to use the <sql:setDataSource> tag
    - You are unable to add the database connection to the CSA
    - You need to connect to a DB that isn't Oracle or SQL Server
    - You want to run the script from the command line and don't have a copy of the properties.xml file on your computer

```
<sql:setDataSource url="jdbc:oracle:thin:@localhost:1521:NIKU"
driver="oracle.jdbc.driver.OracleDriver" user="${ClarityUser}"
password="${ClarityPassword}" var="clarityDS"/>
```

Database Operations - SQL Queries

- **<sql:query>**
  - The result set is stored in the variable declared within the tag

```
<sql:query dataSource="${clarityDS}" escapeText="0" var="result">
        <![CDATA[
        SELECT  r.full_name resName,
                r.email resEmail
        FROM    srm_resources r
        ]]>
</sql:query>
```

- **Note**: Due to the nature of XML, using certain characters (‘<‘, ‘>’) within your query would cause an error without enclosing the query in the CDATA tag. Because of this, it is a best practice to include this tag in all queries.

Database Operations - SQL Queries

- Retrieving the data from the query

```
<!-- By Index -->
<core:forEach trim="true" items="${result.rowsByIndex}" var="row">
    <gel:out>Resource Name: ${row[0]}</gel:out>
</core:forEach>

<!-- By Column Name (Recommended Method) -->
<core:forEach trim="true" items="${result.rows}" var="row">
    <gel:out>Resource Name: ${row.resName}</gel:out>
</core:forEach>
```

- **Note**: When possible, avoid setting a bunch of variables with the results of the query.

Database Operations - SQL Updates

- Processes often require updating the values of specific attributes.
- One of the ways to do this is with a SQL Update statement.
- Keep in mind the following when performing direct database updates
  - Avoid Insert statements – these are best suited for XOG
  - Direct database updates will not trigger a process to start – XOG updates typically will trigger processes to start
  - Avoid updating OOTB tables when possible – using XOG will ensure all Clarity business rules are followed
  - It is generally safe to update custom attributes with a SQL update. These are typically found in tables beginning with odf_ca
  - Extra caution should also be used within On-demand environments

Database Operations - SQL Updates

- **<sql:update>**
  - The declared variable will contain the number of rows that the update statement affects

```
<sql:update dataSource="${clarityDS}" escapeText="0" var="updateCnt">
    <![CDATA[
    UPDATE   odf_ca_project ocp
    SET      ocp.rego_appr_date = sysdate
    WHERE    ocp.id = ${gel_objectInstanceId}
    ]]>
</sql:update>
```

- **Note**: Using CDATA tags in update statements is also recommended.

Database Operations - Bind Variables

- Ensures data types are correct
- Prevents SQL injection
- Allows reuse of SQL statements for performance gains

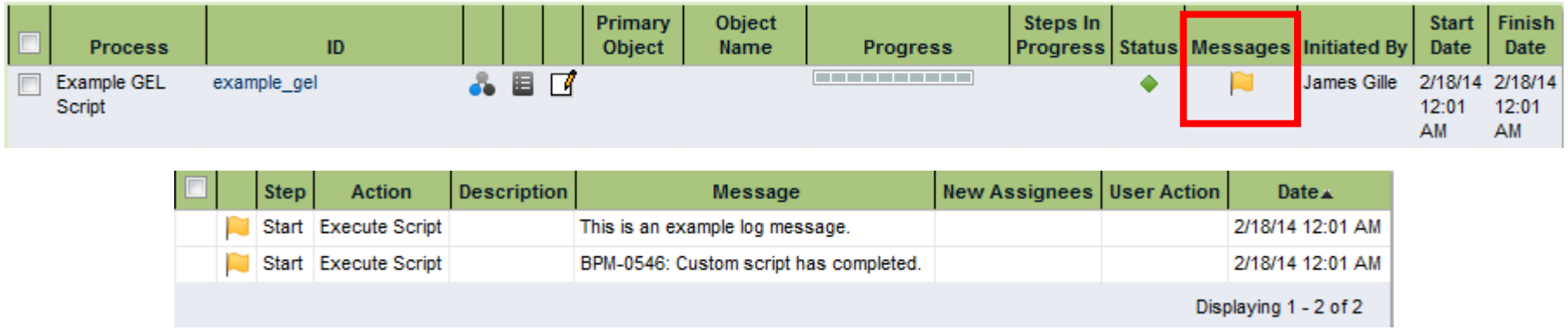
```
<sql:update dataSource="${clarityDS}" escapeText="0" var="updateCnt">
        <![CDATA[
        UPDATE   odf_ca_project ocp
        SET      ocp.rego_appr_date = sysdate,
                 ocp.rego_resource = ?
        WHERE    ocp.id = ?
        ]]>
        <sql:param value="${row.resourceId}"/>
        <sql:param value="${gel_objectInstanceId}"/>
</sql:update>
```

- **Note**: You must have a <sql:param> tag for each ? in the SQL statement, even if they are using the same value. The parameters also must be placed in the order that they appear in the statement.

Logging

- **<gel:log>**
  - Use this tag to insert status messages into the process engine log table
  - Very useful when debugging scripts
  - Avoid using too many logging statements, as it can impact performance
  - Different levels of logging – INFO, WARN, ERROR

```
<gel:log level="INFO">This is an example log message.</gel:log>
```



Process	ID	Primary Object	Object Name	Progress	Steps In Progress	Status	Messages	Initiated By	Start Date	Finish Date
Example GEL Script	example_gel							James Gille	2/18/14 12:01 AM	2/18/14 12:01 AM

Step	Action	Description	Message	New Assignees	User Action	Date
Start	Execute Script		This is an example log message.			2/18/14 12:01 AM
Start	Execute Script		BPM-0546: Custom script has completed.			2/18/14 12:01 AM

Displaying 1 - 2 of 2

Email

- **<gel:email>**
  - Email server information is derived from the properties.xml of the installation.
  - Supports HTML
  - Separate multiple email addresses with a ‘;’

```
<gel:script xmlns:gel="jelly:com.niku.union.gel.GELTagLibrary">

   <gel:email from="clarity-do-not-reply@ca.com" subject="Clarity - Test Email" to="john@gmail.com">
      <![CDATA[
      This is a test email.
      <br/><br/>----------------------------------------------------------------<br/>
      This is an automated message, please do not reply.
      ]]>
  </gel:email>

</gel:script>
```

Email

- **<email:email>**
  - Mail Server must be specified within the tag
  - Does not support HTML
  - Supports Attachments

```
<gel:script xmlns:core="jelly:core" xmlns:email="jelly:email">

  <core:invokeStatic className="java.lang.System" method="getenv" var="NIKU_HOME">
    <core:arg value="NIKU_HOME"/>
  </core:invokeStatic>
  <gel:parse file="${NIKU_HOME}/config/properties.xml" var="properties"/>
  <gel:set asString="true" select="$properties/properties/mailServer/@host" var="mailServer"/>

  <email:email to="john@gmail.com" from="clarity-do-not-reply@ca.com" subject="app-ca.log file" server="${mailServer}"
      attach="${NIKU_HOME}/logs/app-ca.log">
  App-ca.log File
  </email:email>

</gel:script>
```

Best Practices

- If a variable can be changed by an admin, or stores a password, use the <gel:parameter> tag instead of <core:set>
- Comment your code
- Properly format and indent your code
- Place all code within the <gel:script> tags, even comments
- Avoid excessive logging, but keep enough for debugging purposes
- When possible, pull server info from properties file on server
- Avoid sending emails in non-prod environments by either disabling mail server or masking emails

Example Use Cases

- Timesheet Reminder Emails
- Updating Resource Rights
- Data Extraction
- Integrations with HR system
- Capturing Action Item details on Object for Reporting

Exercises

regoUniversity 2017

Let Rego be your guide.

Exercise

- Write a script that will email a list of all active projects with managers that are not active users in Clarity to the resources within the following group:

Group		ID	Description	Status
Rego U - GEL Script Group		regou_gel_script	This group is to be used in the Rego U GEL Script Class for Exercise 2	Active

Exercise - SQL Hints

- **SQL to get Resources in Group**:

```
SELECT   r.first_name || ' ' || r.last_name resName,
         r.email resEmail
FROM     cmn_sec_user_groups ug
         JOIN cmn_sec_groups g ON ug.group_id = g.id AND g.group_code = 'regou_gel_script‘
         JOIN srm_resources r ON r.user_id = ug.user_id
```

- **SQL to get Projects**:

```
SELECT   i.id prjId,
         i.code prjCode,
         i.name prjName,
         r.full_name prjMgr
FROM     inv_investments I
         JOIN inv_projects ip ON ip.prid = i.id AND ip.is_template = 0 AND ip.is_program = 0 AND (i.purge_flag = 0 OR i.purge_flag IS NULL)
         JOIN odf_ca_project ocp ON i.id = ocp.id
         JOIN cmn_sec_users u ON i.manager_id = u.id AND u.user_status_id != 200
         JOIN srm_resources r ON u.id = r.user_id
```

Questions?

regoUniversity 2017

Let Rego be your guide.

Thank You For Attending regoUniversity

Instructions for PMI credits

- Access your account at pmi.org
- Click on **Certification**
- Click on **Maintain My Certification**
- Scroll down to **Report PDU's**
- Click on Course Training (or other appropriate category)
- Enter **Rego Consulting**
- Enter Activity- **Enter Name of Course**
- Enter **Description**
- Enter **Date Started**
- Enter **Date Completed**
- Provide Contact Person **Name of Person to Contact**
- Provide Contact E-Mail **E-Mail of Person to Contact**
- Enter Number of **PDU's Claimed** (1 PDU per course hour)
- Click on the **I agree this claim is accurate box**
- Click **Submit** button

Let us know how we can improve!
Don't forget to fill out the class survey.

Phone
888.813.0444

Email
info@regouniversity.com

Website
www.regouniversity.com