

# Integration with AWS

## **Overview**

OpenLegacy's API integration platform is the fastest and most standard way for legacy applications to be part of the AWS cloud. OpenLegacy quickly and efficiently generates APIs or serverless nodeJS functions for any legacy asset by connecting directly to the legacy system, automating code generation. With a couple of clicks, users can generate a consumable APIs inside containers of Lambda functions. There is no hand coding or additional configuration needed for use with AWS.

# **Key Benefits of OpenLegacy**

- Generates APIs as:
  - Java code for flexibility to deploy in any AWS Endpoint service: Beanstalk, EC2 or others
  - ECF/Fargate ready microservices
  - Serverless NodeJS code Lambda functions
- Direct Connection to almost any core system
- Automatic code generation of APIs inside microservices
- Parses metadata and generates SDK that includes runtime connection to legacy system
- Easily deployed into any infrastructure (Docker, PCF, Tomcat, Kubernetes, OpenShift)

# **Key Benefits of AWS**

- Manage interaction with API consumers to optimize performance
- Wide deployment options for easy consumption, highly reliable Endpoint
- Security at the API and infrastructure layer to add protection to the legacy assets
- Monitoring, Multilayer Architecture, Auto Scaling (in and out), Load Balancing and several services are available to manage and control the APIs
- Services to produce Analytics at the API level

# How OpenLegacy Works: OpenLegacy Can Deploy APIs to AWS in 3 Different Patterns

- 1. Java POJO (Plain Old Java Object) API
- 2. As Microservice
- 3. Serverless Function

#### For this example, we are showing how to deploy on Pattern 1 Java POJO API to EC2:

## 1. Create an SDK project in the OpenLegacy IDE

OpenLegacy SDK	Project Wizard	
This wizard creates	a new OpenLegacy SDK project which enables you to integrate and modernize your Legacy system	
Project Name		
Project Name: Default Package:	ol-sdk com.ol_sdk.openlegacy	

 Populate the connection configuration (host, port, username, pass, etc.). This enables OpenLegacy to build the connection information about the backend into the SDK project. It also can retrieve any metadata from the legacy system for parsing.

### 3. Generate Java code based on metadata of back-end program.

The code goes into the SDK project for use by the APIs.

#### OpenLegacy SDK Project Wizard

This wizard creates a new OpenLegacy SDK project which enables you to integrate and modernize your Legacy system

Host type	MainFrame	
	Open source provider for remote progr	am calls for MainFrame.
CICS Base URL	http://192.86.32.142	

ITEMS.ck	bl		01 IBAN-STRC.	
DPNAC			03 IBAN-CNTRY-CD	PIC X(2).
PLITEN	New	•	03 IBAN-CHECK-DIGITS	PIC 59(2).
➢ trails iii applicatio ▶ Ø src/test/java	Open	₩W ► F3	03 IBAN-BNK-ID 03 IBAN-BRNCH-ID 03 IBAN-ACCOUNT-ID. 05 ACT-ID-SGMNT1	PIC X(4). PIC S9(6). PIC 9(2).
src/test/reso	Open With		05 ACT-ID-SGMNT2	PIC 9(2).
▶ 🛋 JRE System I ▶ 🛋 Maven Deper	📄 Сору	жC	05 ACT-ID-SGMNT3 05 ACT-ID-SGMNT4.	PIC 9(2).
Src	🗎 Copy Qualified Name		07 ACT-ID-SGMNT41	PIC 59(4).
target	🖻 Paste	жv	07 ACT-ID-SGMNT42 05 ACT-ID-SGMNT4-ALPHA REDEF	PIC X.
design-time.	🗙 Delete	X		RAILING SEPARATE CHARACTER.
pom.xml		_	03 FILLER	PIC X(7) VALUE SPACES.
	Remove from Context	℃公駕↓	Ø1 SPACE-COUNT	PIC S9.
	🚮 Mark as Landmark	↑第☆ブ	LINKAGE SECTION.	
	Build Path	•	01 DFHCOMMAREA.	
	Move		03 CRT-ACCOUNT-IN.	
Screen Preview	Rename	F2	05 ACTI-CUSTOMER-ID	PIC X(16).
	Rename	F2	05 ACTI-CUSTOMER-NAME	PIC X(16).
	🚵 Import		05 ACTT-TYPCD	PTC X
	Export		blems 📮 Console 🔀 Ju JUnit egacy	
	🔄 Refresh	F5		bstractCustomizableLifecycleMapping - Moj
	df OpenLegacy	$\triangleright$		bstractCustomizableLifecycleMapping - Moj bstractCustomizableLifecycleMapping -
	Validate Open with Hex Editor		:04.489 [Worker-9] DEBUG o.e.m.c.p.c.Ak :04.489 [Worker-9] DEBUG o.e.m.c.p.c.Ak	bstractCustomizableLifecycleMapping - Moj bstractCustomizableLifecycleMapping - Moj bstractCustomizableLifecycleMapping - Moj

 Create an API project in the OpenLegacy IDE – API data gets populated from the SDK

OpenLegacy API	Project Wizard es a new OpenLegacy API pro	oject			
Microservice P	roject De ervice project configurations	eployment:	Docker	٥	
Project Name:	ol-api				
Default Package:	com.ol_api.openlegacy				
Use Custom Pr	oject Template				

5. Create API inputs and outputs based on the back-end asset generated into the SDK

		Generate API			
Service name: NewService			rallel execution 🗌 Create pool		
type filter text	Input	:			
🔁 ol-api		New name	Original name	Туре	
▼ 🗁 ol-sdk ▼ 🔃 Opnact31	×	crtAccountIn	crtAccountIn	unknown	
V P dfhcommarea (Opnact31Dfhcomn	+				
P crtAccountin (Opnact31CrtAcc					۸
PaccountOut (Opnact31Account					▼
	•				
	Outp	ut:			
		New name	Original name	Type	
	X	accountOut	accountOut	unknown	
	+				
					۸
	•				۲
	•				

6. Put the microservice-based API into a JAR file by choosing "Maven Install"

Paste	ЖV		
💢 Delete	$\boxtimes$		
Remove from Context 7	1第①2		
Build Path	•		
Refactor \C#	T 🕨		
Import	•		
Export	•		
街 Refresh	F5		
Close Project			
Close Unrelated Projects			
🗗 OpenLegacy	•		
Validate			
Run As	•	📙 1 Run on Server	℃ŵΧR
Debug As	•	2 Java Application	~€₩X J
Profile As	•	Ju 3 JUnit Test	∖тжхт
Restore from Local History		m2 4 Maven build	τοxΜ
Java EE Tools	•	m2 5 Maven build	
Maven	•	m2 6 Mayen clean	
TypeScript	•	m2 7 Maven generate-source	00
Team		m2 8 Mayen install	.03
Compare With			
Configure	•	m2 9 Maven test	
Source	•	🥖 Spring Boot App	℃�XB
Spring Tools	•	Spring Devtools Client	
Properties	<b>%</b> I	Run Configurations	

aws

New EC2 Experience

Services 🖌 Resource Groups 🗸 🛧

Launch Instance - Connect Actions \*

- 7. Goto AWS and configure and Launch an EC2 Instance to deploy
- EC2 Dashboard New C Top tags and attributers or search by keyword C C Table C DAS (Pr44) Pr44 (Pr4

4 0 0 0

8. Copy the Java POJO Jar file you created in OpenLegacy into the EC2 Instance



9. Execute the API inside the EC2 instance



**10.** Test the API using the OpenLegacy generated Swagger page

Name		Description			
olApiin * record		olApiln			
(body)		Example Value Model			
		{ "crtAccount[n": { "ectionCdT: 1124" "ectionCdT: 11246, "ectionCdT: 11246, "ectionCdT: 11246, "ecticatorCdT: 11246, "ecticatorCdT: 11246, "ecticatorCdT: 11246, "ectionCdT: 11247, "ectivatorCdT: 11247, "ectivator	". Sofett", Sofett",		
		Cancel Parameter centent type application/json			
				Clear	
Responses	Execute			Response content type	application/json ~
Curl		ngeti spilioni infjorf -i "sudor 11 izdeg (testionryndit: (til 12 ivecti, (testionryndit: (til) 12 ivecti, (testiyndit: (til)	ation: Borror \$25435-510-505-510 $_{\rm A}$ (*actionreso)(*; \*001*, \*actions		
uri carl -x POST "http://184.73 \"ortwccauntIn\": { \"ectia Worlett\", \"ectianitialoep Request URL	141,16415868/6pi/clapi*_+**ec kdd/ti V1254/ti V=tettBrach1d sti(**:1405.66, \*ectisAdrypcd	cg#1 agg1ingtion/joon" -H "muthor V 122480 (Vectourygod*1 (muthor V 1 (methygod*1 (muthor))	arfor Barr 1943-44-44-44 J <sup>(a</sup> ctionrony)'' (*1921'', )*ection		
luf curl - X POST "http://184.75 YoruAccountInt": { { Pectia Roviet21, } tectionisialesp Request URL http://184.73.144.8008/	141,16415868/6pi/clapi*_+**ec kdd/ti V1254/ti V=tettBrach1d sti(**:1405.66, \*ectisAdrypcd	ggel ggl kestion/joon" di "muther 1991 judde, Venetourged" i Venet 1 felder, Venetourged('i felder))	antion daaren dibatta-shar deb kete A (*actioarreng)*t (*a2)(*, )*actioar		
Curl curl - X POST "http://JA4.72 YorLaccountIn\': { / Pectia Morlett\', \'mactiznitialegp Request URL http://JA4.73.144.5000/ Gener response	jau, las lassi/gr/(lage* si *gr lad * 1*224 - 1*8218 - 1*8218 - 1*8218 sttl* 1200 00 1*8218 - 1*8218 pt/clage	ngari gang kana kana jugan di mandan 19 juganda ( mana kana gang ( 1997) 19 mary - Sana Sana Sana Sana Sana Sana 19 mary - Sana Sana Sana Sana Sana Sana Sana Sa	nefor beer water-te-de-bet A bectarrecy's Vall', Vestad		
Request URL http://184.73.143.18418888/ Server response	141,16415868/6pi/clapi*_+**ec kdd/ti V1254/ti V=tettBrach1d sti(**:1405.66, \*ectisAdrypcd	nar: antication(jan" ar mahar Vi zada Vantanga) - Vi Vi Yaqi", Vantanga) - Vi Vi Yaqi", Vantagadi (VV )	antioni Ameror Allalis-clam 462-bills A Ventionency (* Venty , Ventione		

# **About OpenLegacy**

OpenLegacy accelerates delivery of innovative digital services from legacy systems in days or weeks versus months. Our microservices-based API integration and management software reduces manual effort by automating API creation, simplifies the process by avoiding layers of complexity, and improves staff efficiency and API performance. Our software directly accesses and extends business logic to web, mobile or cloud innovations in the form of Java objects, REST APIs, ODATA APIs, or SOAP. Most importantly, this process is not only fast, easy and secure, but also does not require special staff skills or changes to existing systems or architecture. Together, business and IT teams can quickly, easily and securely meet consumer, partner or employee demands for digital services without altering or replacing core systems. Learn why leading companies choose OpenLegacy at www.openlegacy.com.



www.openlegacy.com sales@openlegacy.com