OpenLegacy
Digital to the Core

Datasheet

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

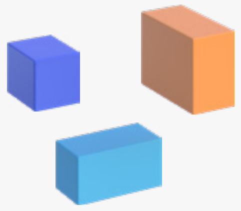
- Generates APIs as:
  - Java code for flexibility to deploy in any AWS Endpoint service: Beanstalk, EC2 or others
  - ECF/Fargate ready microservices
  - Serverless NodeJS code Lambda functions
- Direct Connection to almost any core system
- Automatic code generation of APIs inside microservices
- Parses 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

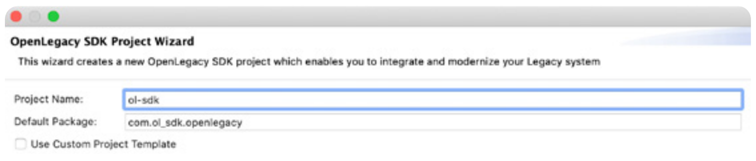
1. Java POJO (Plain Old Java Object) API
2. As Microservice
3. Serverless Function



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

01

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: ol-sdk

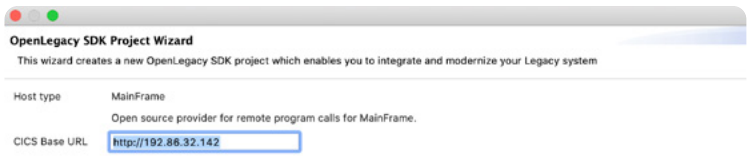
Default Package: com.ol_sdk.openlegacy

☐ Use Custom Project Template

02

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.



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

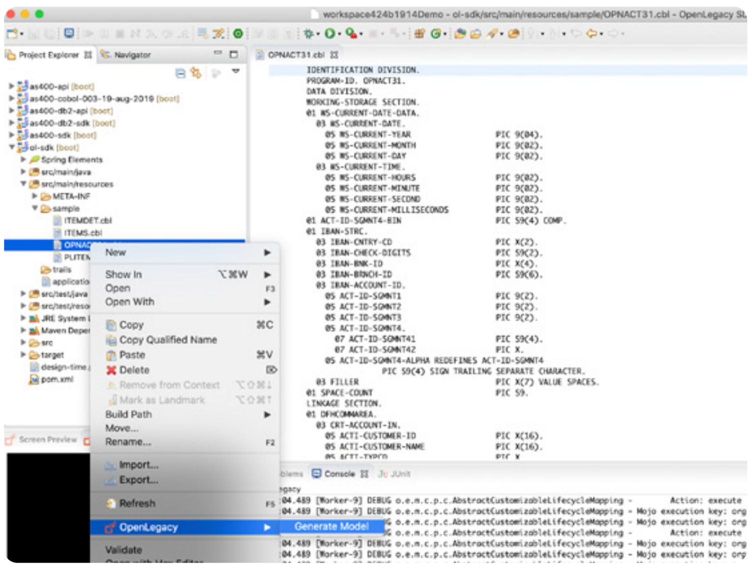
Open source provider for remote program calls for MainFrame.

CICS Base URL: http://192.86.32.142

03

Generate Java code based on metadata of back-end program

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



workspace424b1914Demo - ol-sdk/src/main/resources/sample/OPNACT31.cbl - OpenLegacy SD

OPNACT31.cbl

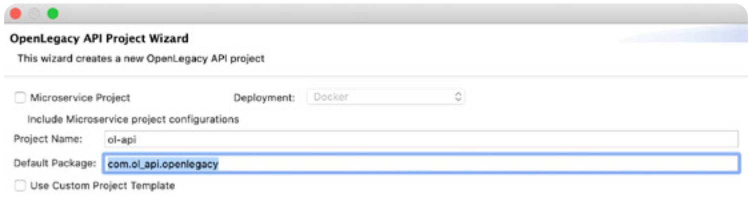
```
       IDENTIFICATION DIVISION.
       PROGRAM-ID. OPNACT31.
       DATA DIVISION.
       WORKING-STORAGE SECTION.
       01 WS-CURRENT-DATE-DATA.
         03 WS-CURRENT-DATE.
           05 WS-CURRENT-YEAR                PIC 9(04).
           05 WS-CURRENT-MONTH               PIC 9(02).
           05 WS-CURRENT-DAY                 PIC 9(02).
         03 WS-CURRENT-TIME.
           05 WS-CURRENT-HOURS               PIC 9(02).
           05 WS-CURRENT-MINUTE              PIC 9(02).
           05 WS-CURRENT-SECOND              PIC 9(02).
           05 WS-CURRENT-MILLISECONDS        PIC 9(02).
       01 ACT-ID-SGMNT4-BIN                  PIC S9(4) COMP.
       01 IBAN-STRC.
         03 IBAN-CNTRY-CD                    PIC X(2).
         03 IBAN-CHECK-DIGITS                PIC S9(2).
         03 IBAN-BNK-ID                      PIC X(4).
         03 IBAN-BRNCH-ID                    PIC S9(6).
         03 IBAN-ACCOUNT-ID.
           05 ACT-ID-SGMNT1                  PIC 9(2).
           05 ACT-ID-SGMNT2                  PIC 9(2).
           05 ACT-ID-SGMNT3                  PIC 9(2).
           05 ACT-ID-SGMNT4.
             07 ACT-ID-SGMNT41               PIC S9(4).
             07 ACT-ID-SGMNT42               PIC X.
           05 ACT-ID-SGMNT4-ALPHA REDEFINES ACT-ID-SGMNT4
                     PIC S9(4) SIGN TRAILING SEPARATE CHARACTER.
         03 FILLER                           PIC X(7) VALUE SPACES.
       01 SPACE-COUNT                        PIC S9.
       LINKAGE SECTION.
       01 DFHCOMMAREA.
         03 CRT-ACCOUNT-IN.
           05 ACTI-CUSTOMER-ID               PIC X(16).
           05 ACTI-CUSTOMER-NAME             PIC X(16).
           05 ACTI-TYPCD                     PIC X
```

```
egacy
:04.489 [Worker-9] DEBUG o.e.m.c.p.c.AbstractCustomizableLifecycleMapping -         Action: execute
:04.489 [Worker-9] DEBUG o.e.m.c.p.c.AbstractCustomizableLifecycleMapping - Mojo execution key: org
                      G o.e.m.c.p.c.AbstractCustomizableLifecycleMapping - Mojo execution key: org
                      G o.e.m.c.p.c.AbstractCustomizableLifecycleMapping -         Action: execute
:04.489 [Worker-9] DEBUG o.e.m.c.p.c.AbstractCustomizableLifecycleMapping - Mojo execution key: org
:04.489 [Worker-9] DEBUG o.e.m.c.p.c.AbstractCustomizableLifecycleMapping - Mojo execution key: org
```

04

Create an API project in the OpenLegacy IDE

API data gets populated from the SDK


OpenLegacy API Project Wizard

This wizard creates a new OpenLegacy API project

☐ Microservice Project Deployment: Docker

Include Microservice project configurations

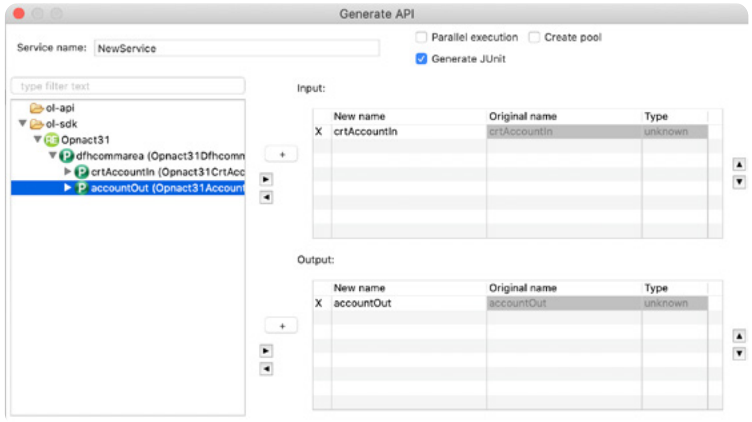
Project Name: ol-api

Default Package: com.ol_api.openlegacy

☐ Use Custom Project Template

05

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


Generate API

Service name: NewService

☐ Parallel execution ☐ Create pool

☑ Generate JUnit

type filter text
ol-api
ol-sdk
Opnact31
dfhcommarea (Opnact31Dfhcomm
crtAccountIn (Opnact31CrtAcc
accountOut (Opnact31Account

Input:

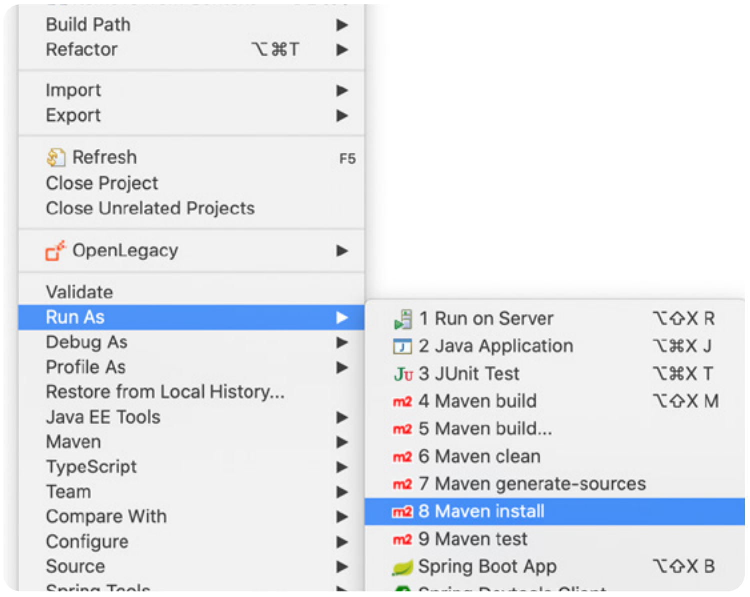
	New name	Original name	Type
X	crtAccountIn	crtAccountIn	unknown

Output:

	New name	Original name	Type
X	accountOut	accountOut	unknown

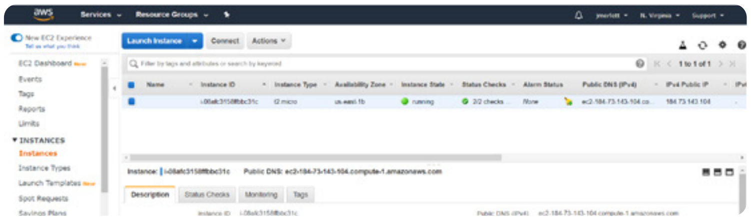
06

Put the microservice-based API into a JAR file by choosing “Maven Install”



07

Go to AWS and configure and Launch an EC2 Instance to deploy



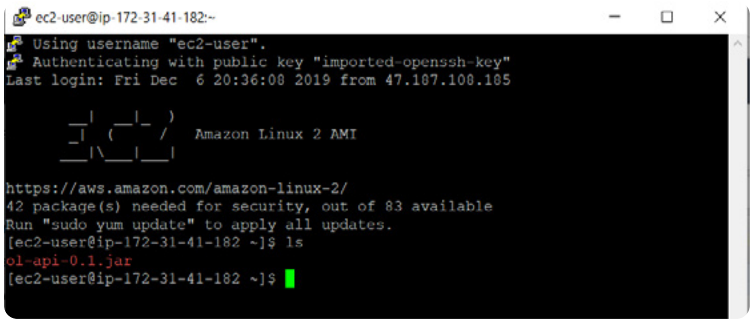
Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP
	i-08afc3158fbbc31c	t2.micro	us-east-1b	running	2/2 checks ...	None	ec2-184-73-143-104.co...	184.73.143.104

Instance: i-08afc3158fbbc31c Public DNS: ec2-184-73-143-104.compute-1.amazonaws.com

Instance ID i-08afc3158fbbc31c Public DNS (IPv4) ec2-184-73-143-104.compute-1.amazonaws.com

08

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



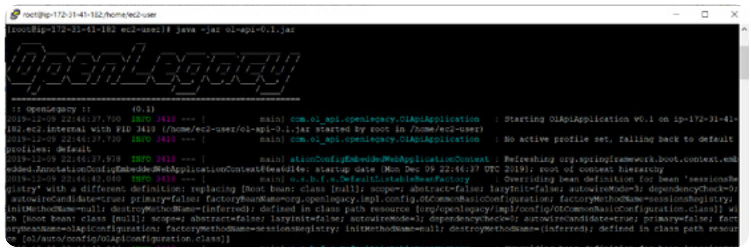
```
Using username "ec2-user".
Authenticating with public key "imported-openssh-key"
Last login: Fri Dec  6 20:36:08 2019 from 47.187.108.185

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
42 package(s) needed for security, out of 83 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-41-182 ~]$ ls
ol-api-0.1.jar
[ec2-user@ip-172-31-41-182 ~]$
```

09

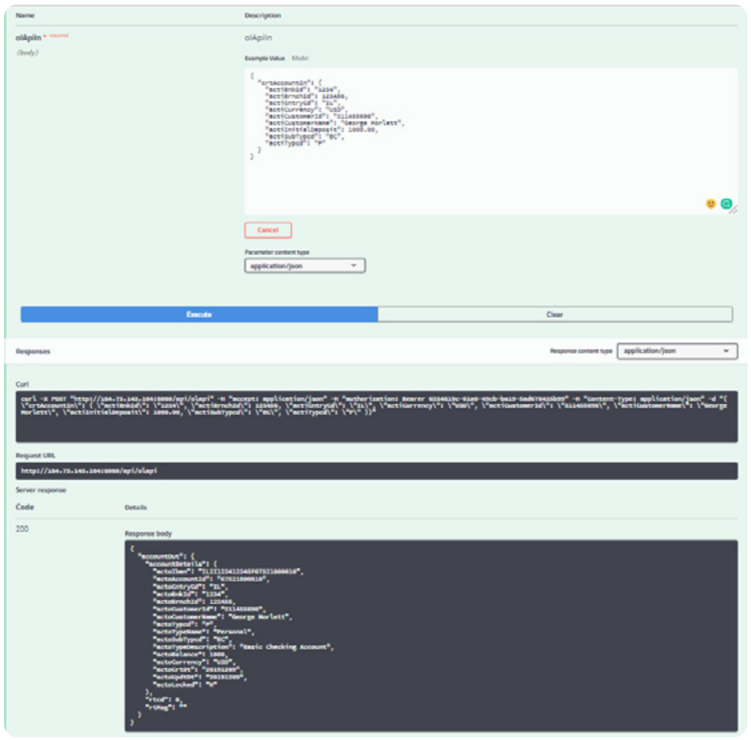
Execute the API inside the EC2 instance



```
[root@ip-172-31-41-182 ec2-user]# java -jar ol-api-0.1.jar
  OpenLegacy
 ====================================================
 :: OpenLegacy ::         (0.1)
2019-12-09 22:46:37.700  INFO 3418 --- [           main] com.ol_api.openlegacy.OlApiApplication   : Starting OlApiApplication v0.1 on ip-172-31-41-
182.ec2.internal with PID 3418 (/home/ec2-user/ol-api-0.1.jar started by root in /home/ec2-user)
2019-12-09 22:46:37.730  INFO 3418 --- [           main] com.ol_api.openlegacy.OlApiApplication   : No active profile set, falling back to default
profiles: default
2019-12-09 22:46:37.978  INFO 3418 --- [           main] ationConfigEmbeddedWebApplicationContext : Refreshing org.springframework.boot.context.emb
edded.AnnotationConfigEmbeddedWebApplicationContext@6ea6d14e: startup date [Mon Dec 09 22:46:37 UTC 2019]; root of context hierarchy
2019-12-09 22:46:42.040  INFO 3418 --- [           main] o.s.b.f.s.DefaultListableBeanFactory     : Overriding bean definition for bean 'sessionsRe
gistry' with a different definition: replacing [Root bean: class [null]; scope=; abstract=false; lazyInit=false; autowireMode=3; dependencyCheck=0;
 autowireCandidate=true; primary=false; factoryBeanName=org.openlegacy.impl.config.OLCommonBasicConfiguration; factoryMethodName=sessionsRegistry;
initMethodName=null; destroyMethodName=(inferred); defined in class path resource [org/openlegacy/impl/config/OLCommonBasicConfiguration.class]] wi
th [Root bean: class [null]; scope=; abstract=false; lazyInit=false; autowireMode=3; dependencyCheck=0; autowireCandidate=true; primary=false; fact
oryBeanName=olApiConfiguration; factoryMethodName=sessionsRegistry; initMethodName=null; destroyMethodName=(inferred); defined in class path resour
ce [ol/auto/config/OlApiConfiguration.class]]
```

10

Test the API using the OpenLegacy generated Swagger page



Parameter content type: application/json

```
{
  "crtAccountIn": {
    "actiBnkId": "1234",
    "actiBrnchId": 123456,
    "actiCntryId": "IL",
    "actiCurrency": "USD",
    "actiCustomerId": "311455696",
    "actiCustomerName": "George Morlett",
    "actiInitialDeposit": 1000.00,
    "actiSubTypcd": "BC",
    "actiTypcd": "P"
  }
}
```

Response content type: application/json

Curl

```
curl -X POST "http://[illegible]:8080/api/olapi" -H "accept: application/json" -H "Authorization: Bearer [illegible]" -H "Content-Type: application/json" -d "{ \"crtAccountIn\": { \"actiBnkId\": \"1234\", \"actiBrnchId\": 123456, \"actiCntryId\": \"IL\", \"actiCurrency\": \"USD\", \"actiCustomerId\": \"311455696\", \"actiCustomerName\": \"George Morlett\", \"actiInitialDeposit\": 1000.00, \"actiSubTypcd\": \"BC\", \"actiTypcd\": \"P\" }}"
```

Request URL

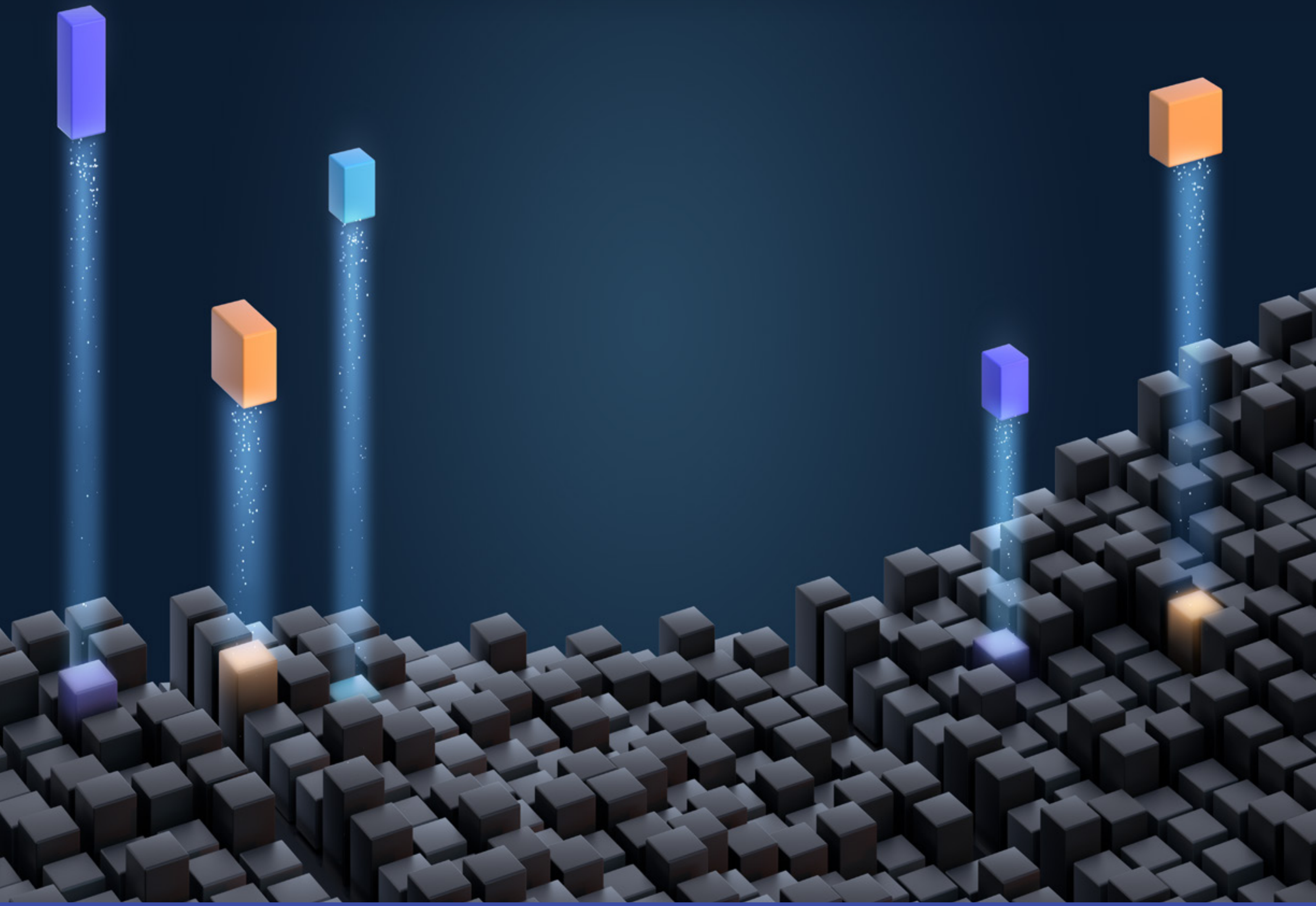
```
http://[illegible]:8080/api/olapi
```

200

```
{
  "accountOut": {
    "accountDetails": {
      "actoIban": "[illegible]",
      "actoAccountId": "[illegible]",
      "actoCntryCd": "IL",
      "actoBnkId": "1234",
      "actoBrnchId": 123456,
      "actoCustomerId": "311455696",
      "actoCustomerName": "George Morlett",
      "actoTypcd": "P",
      "actoTypeName": "Personal",
      "actoSubTypcd": "BC",
      "actoTypeDescription": "Basic Checking Account",
      "actoBalance": 1000,
      "actoCurrency": "USD",
      "actoCrtDt": "20191209",
      "actoUpdtDt": "20191209",
      "actoLocked": "N"
    },
    "rtcd": 0,
    "rtMsg": ""
  }
}
```

About OpenLegacy

OpenLegacy's Digital-Driven Integration enables organizations with legacy systems to release new digital services faster and more efficiently than ever before. It connects directly to even the most complex legacy systems, bypassing the need for extra layers of technology. It then automatically generates APIs in minutes, rapidly integrating those assets into exciting new innovations. Finally, it deploys them as standard microservices or serverless functions, giving organizations speed and flexibility while drastically cutting costs and resources. With OpenLegacy, industry-leading companies release new apps, features, and updates in days instead of months, enabling them to truly become digital to the core.



© OpenLegacy Inc. All Rights Reserved
DS_AWS_11Apr2020