Developing a RESTful Web application for Liberty in CICS
Introduction

Course introduction
What you’ll see in this course

Eclipse development environment

RESTful web application

CICS region

Liberty JVM server

RESTful web application

JCICS

COBOL program

RESTful client

HTTP Call

JCICS
What you’ll need for this course

- Development workstation
- Ability to install IBM Explorer for z/OS
- Access to sample code
- CICS TS for z/OS V5.1 or later
What you’ll learn by the end of this course

1. Developing a RESTful Java web service
2. Using the CICS Java API
3. Deployment of web applications
Creating the development environment

Installing IBM Explorer for z/OS
Downloading Eclipse

developer.ibm.com/mainframe/products/downloads

Option 5 — I am starting from scratch
For a first time user, download the IM installer for IBM Explorer for z/OS Aqua release by selecting one of the platforms and perform the following steps to install.
Launching IBM Installation Manager

The IBM Installation Manager is a graphical user interface tool used for installing and managing IBM software products. This image shows the installation window where users can select packages to install, including IBM Installation Manager itself and other related software like IBM Rational Developer for System z and IBM CICS Explorer SDK.

---

**Details**

- **IBM CICS Explorer SDK 5.3.0.6**
  - The IBM CICS Explorer SDK is offering [more info](http://public.dhe.ibm.com/ibmdl/export/pub/software/http/tec/tools/sdk/).
Installing packages
Creating the development environment
Configuring CICS Explorer
Configuring connectivity

- Configure connectivity to CICS
- Configure FTP connection
Enabling FTP connection
Connecting to a CICS region

- Create a CMCI connection to CICS
- On your CICS system
  1. Add load libraries SEYUAUTH and SEYULOAD to the CICS JCL
  2. Create a URIMAP
  3. Create a TCPIPSERVICE
  4. Or setup CICSplex SM WUI server
- See Knowledge Center topic Setting up access for CICS Explorer
Summary

- Installed CICS Explorer
- Created FTP and CMCI connections
- Next section - Using CICS Explorer to manage a Liberty JVM
Configuring a CICS Liberty JVM server

Introduction to the Liberty JVM server
Introduction to the Liberty JVM server

- WebSphere Liberty
- Java EE application server
- Tightly integrated with CICS
- Liberty JVM server
- Basic steps of how to configure
Liberty and CICS: A 10,000 feet view
Configuring a CICS Liberty JVM server

Configuring a Liberty JVM server
Configuring Java in CICS

- CICS region started
- SDFJAUTH in STEPLIB
- LE support added
System initialization parameters

USSHOME=/usr/lpp/cicsts/cics700
JVM profile directory

JVMPROFILEDIR=/u/cics1
JVM sample profile

/usr
   /lpp
      /cicsts
         /cics700
            /docs
            /IBM
            /JVMProfiles
          ...         /DFHOSGI.jvmprofile
               /DFHWLP.jvmprofile
USSHOME
Sample Liberty 
JVM profile
JVM sample profile
Copying the sample profile

${USSHOME}/JVMProfiles
/usr/lpp/cicsts/cics700/JVMProfiles
DFHWLP.jvmprofile

${JVMPROFILEDIR}
/u/cics1
Copying the sample profile

${USSHOME}/JVMProfiles
/usr/lpp/cicsts/cics700/JVMProfiles
DFHWLP.jvmprofile

${JVMPROFILEDIR}
/u/cics1
Copying the sample profile

DFHWLP.jvmprofile

DFHWLP.jvmprofile
Reviewing and updating the JVM profile
Reviewing and updating the JVM profile
Enabling auto configure
Setting the TCP/IP host values
Selecting the JVM server resource definition

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Change Time</th>
<th>JVM Profile Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPHAXIS</td>
<td>CICS JVM server to run Axis2 samples</td>
<td>26-Apr-2016 14:21:51</td>
<td>DPHIVMAX</td>
</tr>
<tr>
<td>DPHVMS</td>
<td>CICS JVM server to run OSGi samples</td>
<td>26-Apr-2016 14:21:51</td>
<td>DPHOSGI</td>
</tr>
<tr>
<td>DPHWLP</td>
<td>CICS JVM server to run WLP samples</td>
<td>26-Apr-2016 14:22:51</td>
<td>DPHWLP</td>
</tr>
<tr>
<td>DPHSTTS</td>
<td>CICS Security Token Service</td>
<td>26-Apr-2016 14:22:52</td>
<td>DPHIVMST</td>
</tr>
</tbody>
</table>
Summary

- Created JVM profile using supplied template DFHWLP.jvmprofile
- Created CICS JVMSERVER resource definition
- Next lecture - Install and enable the JVM server
Configuring a CICS Liberty JVM server

Starting and validating the Liberty JVM server
Starting the Liberty JVM server
Verifying the Liberty JVM server is running

<table>
<thead>
<tr>
<th>Region</th>
<th>Name</th>
<th>Enable Status</th>
<th>Max Threads</th>
<th>Threads</th>
<th>Install Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>IYK2Z3E</td>
<td>DFH-WLP</td>
<td>ENABLED</td>
<td>15</td>
<td>8</td>
<td>31-Aug-2016 20:24:48</td>
</tr>
</tbody>
</table>
Accessing the Liberty server default welcome page
Viewing the Liberty message log
Verifying the logs
Verifying the logs
Editing the server.xml file
Adding the JAX-RS feature
Enabling dropins
Editing the server.xml file
Verifying the changes
Creating a RESTful service for Liberty in CICS

Introduction to our RESTful web application
RESTful web applications

- REST
- JAX-RS
- Java EE 6
- JSON
Sample application architecture

```
CICS Region
Sample application architecture
RESTful 
Client
Liberty JVM Server
JAX-RS Runtime
Web Application
JAX-RS
HTTP 
GET request
{json}
method invoke
method return
Response
 Data
```
Creating a RESTful service for Liberty in CICS

Developing the application
Sample application

github.com/cicsdev/cics-java-liberty-restapp
New project dialog
New dynamic web project wizard
New dynamic web project wizard
Sample application architecture

- **CICS region**
- **Liberty JVM server**
- **JAX-RS runtime**
- **JAX-RS**
- **Web application**

RESTful client

- HTTP GET request
- {json} method invoke

JAX-RS runtime

- method return
- Response data

Web application

- method return
Java bean classes

CICSInformation
  
  APPLID
  JVM server
  Current time
  CICS environment

CICSEnvironment
  
  Product name
  Product version
getCICSInformation method
Importing the code
Importing the code

Select how files and folders should be imported into the project:
- Copy files and folders
- Link to files and folders
- Link to files and recreate folder structure with virtual folders

Create link locations relative to: PROJECT_LOC
Verifying packages
Configuring the build path
Adding libraries to the build path
Selecting libraries to add
Selecting the target CICS version for deployment
JAXB annotations
Sample application architecture

RESTful Client → CICS Region

HTTP GET request → JAX-RS Runtime

{json} → method invoke

JAX-RS Runtime → JAX-RS

method return → Response Data

Response Data → Web Application
JAX-RS annotations
JAX-RS annotations
getCICSInformation method
CICSInformation class
Summary

- Completed coding of our application
- Ready to deploy into our Liberty JVM server
- We will extend our application in the next lecture
Creating a RESTful service for Liberty in CICS

Deploying the application to the dropins directory
Deploying via the dropins directory
Exporting a WAR file
Copying WAR file to z/OS
Messages.log file
Messages.log file
Testing the application

The URL

Sample application architecture

- RESTful Client
- HTTP GET request
- {json}
- JAX-RS Runtime
- method invoke
- Response Data
- method return
- JAX-RS
- Web Application
- Liberty JVM Server
- JAX-RS Runtime
- CICS Region
JSON response
Summary

- Coded
- Deployed
- Tested
Creating a RESTful service for Liberty in CICS

Deploying the application in a CICS bundle
Deploying the application in a CICS bundle

- RESTful web application with dropins
- Useful for rapid application development
- Deploy the WAR file using a CICS bundle
- First delete any existing restapp WAR files from dropins
What is a CICS bundle?
CICS bundle project
CICS bundle project

Create a new project containing the files for deployment in a CICS Bundle

Project name: com.ibm.cicsdevrestapp.bundle

Location: C:\Workspaces\OS\Explorer\com.ibm.cicsdevrestapp.bundle

Choose file system: default

Bundle properties
ID: com.ibm.cicsdevrestapp.bundle
Version: 1.0.0

Finish
CICS bundle project
Adding a web project to a CICS bundle
Adding a web project to a CICS bundle
Adding a web project to a CICS bundle
Exporting a CICS bundle project
Exporting a CICS bundle project

Export to z/OS UNIX File System

Export Bundle
Choose where to export this bundle.

- Export to the home directory of a Platform.
- Export to a specific location in the file system.

< Back   Next >   Finish   Cancel
Exporting a CICS bundle project

The image shows a dialog box for exporting a bundle project, with options to specify the bundle project name, connections, parent directory, bundle directory, and various export options. The highlighted options are to resolve variables using properties file and clear existing contents of Bundle directory.
Exporting a CICS bundle project

z/OS
z/OS UNIX folder=/u/iburnet/liberty/com.ibm.cicsdev.restapp.bundle_1.0.0/ - created
z/OS UNIX folder=/u/iburnet/liberty/com.ibm.cicsdev.restapp.bundle_1.0.0/META-INF/ - created
z/OS UNIX file=/u/iburnet/liberty/com.ibm.cicsdev.restapp.bundle_1.0.0/META-INF/cics.xml - created
z/OS UNIX file=/u/iburnet/liberty/com.ibm.cicsdev.restapp.bundle_1.0.0/com.ibm.cicsdev.restapp.warbundle - created
z/OS UNIX file=/u/iburnet/liberty/com.ibm.cicsdev.restapp.bundle_1.0.0/com.ibm.cicsdev.restapp.war - created
Creating a CICS bundle definition
Creating a CICS bundle definition
Creating a CICS bundle definition
Installing a CICS bundle definition
Installing a CICS bundle definition
Installing a CICS bundle definition
Installing a CICS bundle definition
Liberty messages.log file
Testing the application
Summary

- Created sample application
- Deployed into CICS
  - Dropins directory
  - CICS bundles
Linking to a CICS COBOL program

Introduction to linking to a CICS program
Introduction to linking to a CICS program

- CICS concepts
- JCICS
  - Calling an existing COBOL program
- EDUCHAN sample
  - github.com/cicsdev/cics-java-liberty-restapp
What is a task?
Linking to a CICS COBOL program
Using JCICS to link to a CICS program
Application diagram

CICS region
Liberty JVM server
Web application
COBOL program

RESTful client

HTTP request

{json}

JAX-RS runtime

method invoke

Response data

method return

JAX-RS

JCICS

REPORT

LINK
data

Return data

JAX-RS runtime

Web application

JAX-RS

JCICS

Return data
Reviewing and updating the sample application
Reviewing and updating the sample application
Reviewing and updating the sample application
Reviewing and updating the sample application
Reviewing and updating the sample application
Deploying the application
Testing the reverse application
Test the application
Using JCICS to link to a CICS program

- Expose an existing COBOL program as a RESTful web application
Summary

- JCICS API to access CICS tasks and programs
- Passing data using channels and containers
- Next section - Summary and highlights
Summary

Review of what we have learned
What we learned in this course

1. Creation of a development environment
2. Configuration of a Liberty JVM server
3. Development using JAX-RS
4. Use of the JCICS API
5. Application deployment
Download and install the sample applications

- Install sample RESTful web application
  
  cics-java-liberty-restapp
  
  Sample RESTful web application for deployment to a Liberty JVM server in CICS

- Install EDUCHAN COBOL program