# CICS and Db2 for Java

Steve Fowlkes – fowlkes@us.ibm.com
Leigh Compton – lcompton@us.ibm.com

IBM

# Trademarks and Copyright



IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at

www.ibm.com/legal/copytrade.shtml

# Agenda

CICS and Db2 for Java

- The Db2 API will not be covered
- See:
  - SG24-8038 – CICS and the JVM Server – Developing and Deploying
  - Various Db2 manuals
  - The Internet

Note: This presentation does not show JDBC or SQLJ coding

# Bulleted lists

The following bullet styles are recommended:

Text level one

- Bullet level one
  - Bullet level two
    - Bullet level three
      - » Bullet level four

- Use the **tab** key or the **Indent** button in the navigation menu/ribbon to create bulleted lists.
- Avoid using the Bullets button, which would apply default bullet styles to your text.

# Abstract

There's never been a better time to take advantage of Java technology in CICS.

This topic discusses what you should be aware of in the CICS and Db2 for Java environment, to include making the JDBC and SQLJ drivers available.

This topic addresses how to make Db2 functional in a Java environment

# Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this presentation in other countries.

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PRESENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR CONDITIONS OF NON-INFRINGEMENT, MERCHANTABILTY OR FITNESS FOR A PARTICULAR PURPOSE.

This information could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this presentation at any time without notice.

Any references in this presentation to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

# Using Db2 with Java in CICS

Type 2 driver
- For Java applications running in CICS, the type 2 driver is usually used because processing is controlled by the native CICS-Db2 Attachment facility

Type 4 driver
- In the CICS TS Liberty environment, you can use a type 4 database driver

CICS must be able to read the JVM profiles on zFS (permissions)
- Pointed to by the SIT (System Initialization Table)

Drivers are specified in JVM profile (OSGi-based JVM) or in the server.xml (Liberty-based JVM)

Must add the Db2 SDSNLOD2 library to the STEPLIB
Not needed with APARS PI18798 and PI18799

# Using Db2 with Java in CICS

*Type 2*

- are written partly in the Java programming language and partly in native code
- drivers use a native (non Java) client library specific to the data source to which they connect
  - their code portability is limited because of this native code
- might provide higher performance

*Type 4*

- drivers that are pure Java and implement the network protocol for a specific data source
- client connects directly to the data source
- Db2 supports the IBM Data Server Driver for JDBC and SQLJ

# Db2 Maintenance

http://www-969.ibm.com/software/reports/compatibility/clarity/softwareReqsForProduct.html

Prerequisites:

- http://www-01.ibm.com/support/docview.wss?uid=swg21207399#4

- IBM Data Server Driver for JDBC and SQLJ version requirements for CICS Java applications that use JDBC or SQLJ
  - Described in technote http://www-01.ibm.com/support/docview.wss?uid=swg21428742

The driver that is based on the JDBC 3 specification is no longer supported

# OSGi-based JVM server Profile

Provide access to the Db2 OSGi bundle versions of the JDBC and SQLJ drivers in your JVM server profile

Use OSGI_BUNDLES and LIBPATH_SUFFIX

Add the JDBC 4.0 middleware bundle, along with the Db2 license bundle to the OSGI_BUNDLES option

e.g – in the JVM profile file for the OSGi-based JVM server that will use Db2

```
LIBPATH_SUFFIX=/usr/lpp/db2v12/jdbc/lib

OSGI_BUNDLES=/usr/lpp/db2v12/jdbc/classes/db2jcc4.jar,\
        /usr/lpp/db2v12/jdbc/classes/db2jcc_license_cisuz.jar
```

# Eclipse IDE Environment

Once per Eclipse workspace… For development, add the db2jcc4.jar file to Eclipse

- Transfer the db2jcc4.jar file from zFS to a directory on your workstation

- In Eclipse
  - Click Window->Preferences, then Plug-in Development->Target Platform
  - Select the CICS TS V5.x Runtime and click Edit
  - In the Locations tab, click Add and select the folder containing the db2jcc4.jar file
  - In the Content tab select com.ibm.db2.jcc plugin and then click Finish and OK

# Create a New Project in Eclipse

File->New->Plug-in Project

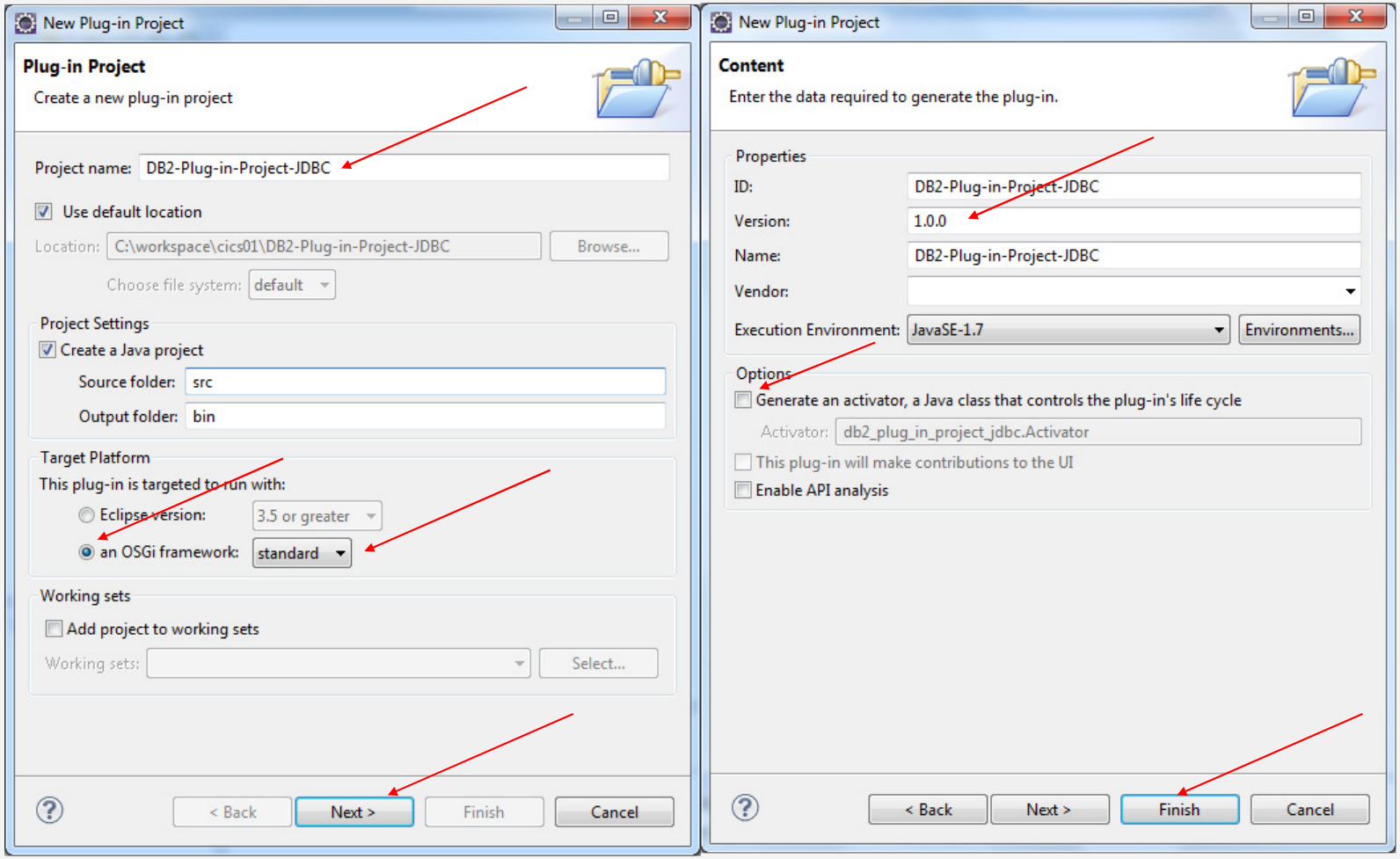
Fill in:
- Project name
- An OSGi Framework: standard
- Next button

- Remove from version: .qualifier
- Uncheck: Generate an activator
- Finish button

Edit the manifest

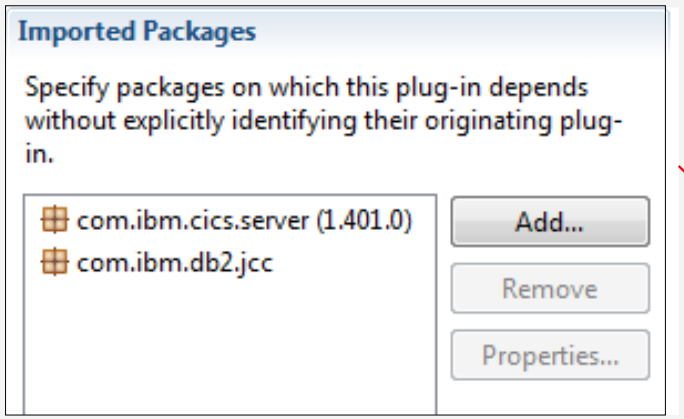
The steps are illustrated on the next few pages

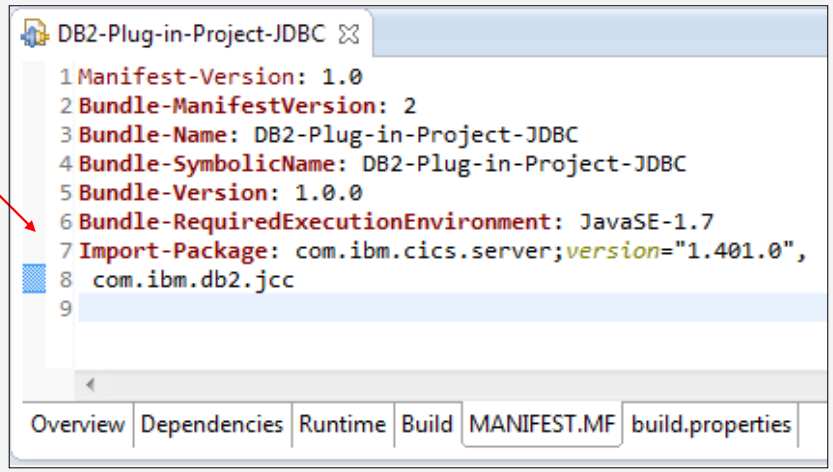
# Create a Project in Eclipse



# Change the Manifest for the Project (JDBC)

For JDBC:




Plus any other dependencies

Plus Exported Packages

– If any packages in this bundle will be used by other bundles

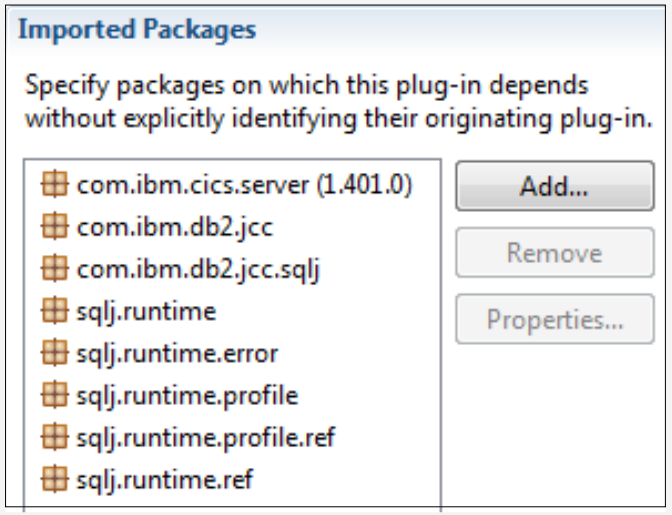
Plus a CICS-MainClass

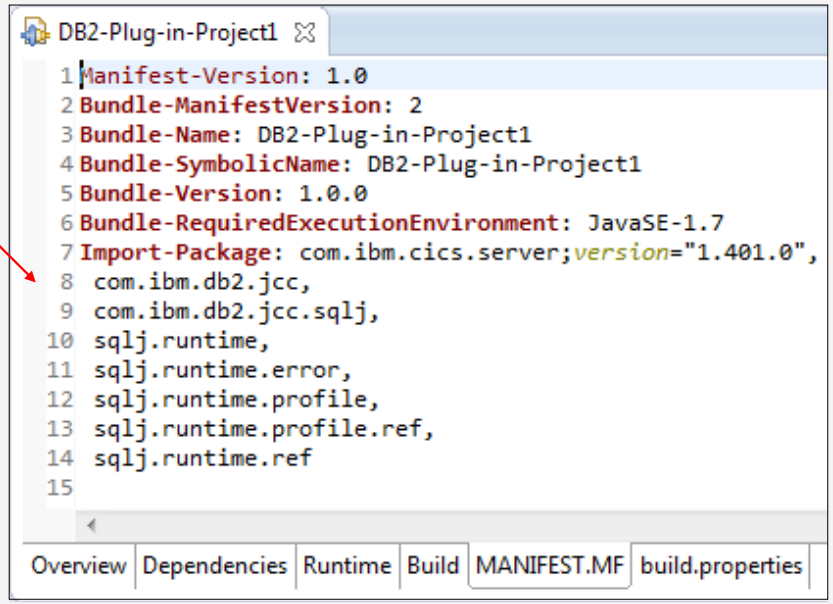
– If you will have a CICS PROGRAM definition point to a class

* A range of levels is preferred for com.ibm.cics.server

# Change the Manifest for the Project (SQLJ)

For SQLJ:




Plus any other dependencies

Plus Exported Packages

– If any packages in this bundle will be used by other bundles

Plus a CICS-MainClass

– If this contains a CICS PROGRAM entry

* A range of levels is preferred for com.ibm.cics.server

# Liberty JDBC Features: local Db2: type 2 driver: in server.xml

Add the jdbc-4.x feature

- To enable Db2 type 2 connectivity when you are running Java 11, add LIBPATH_SUFFIX=/usr/lpp/db2v12/jdbc/lib to the JVM profile

Add a library element to the server.xml file to specify the location on zFS of the JDBC driver

```
<library id="db2Type2Driver">
   <fileset dir="/usr/lpp/db2v12/jdbc/classes" includes="db2jcc4.jar db2jcc_license_cisuz.jar"/>
   <fileset dir="/usr/lpp/db2v12/jdbc/lib" includes="libdb2jcct2zos4_64.so"/>
</library>
```

For DataSource additionally specify

```
<dataSource id="db2Type2" jndiName="jdbc/db2Type2" transactional="false">
   <jdbcDriver libraryRef="db2Type2Driver"/>
   <properties.db2.jcc driverType="2"/>
   <connectionManager agedTimeout="0"/>
</dataSource>
```

- defines the JNDI name that is used by your application when establishing a connection to that data source
- transactional="false" is required to allow CICS to manage the transactions
- driverType="2" is required to use the type 2 connectivity to Db2
- agedTimeout="0" is required to disable Liberty connection pooling
  - Liberty connection pooling is not required as the DB2CONN resource provides Db2 connection pooling

# Liberty CICS JDBC feature: local Db2: type 2 driver: in server.xml

Add the cicsts:jdbc-1.0 feature

For use with DriverManager or DataSource, specify

```
<library id="db2Library">
    <fileset dir="/usr/lpp/db2v12/jdbc/classes" includes="db2jcc4.jar db2jcc_license_cisuz.jar"/>
    <fileset dir="/usr/lpp/db2v12/jdbc/lib" includes="libdb2jcct2zos4_64.so"/>
</library>
```

Add a cicsts_jdbcDriver element

To enable JDBC type 2 connectivity with the java.sql.DriverManager or javax.sql.DataSource interface

```
<cicsts_jdbcDriver libraryRef="db2Library"/>
```

To access Db2 that uses the data source interface, a cicsts_dataSource element must be specified

But the preferred method for data source access is to use the Liberty JDBC features

```
<cicsts_dataSource jndiName="jdbc/cicsDb2DataSource"/>
    <properties.db2.jcc currentSchema="DB2USER" fullyMaterializeLobData="true" />
</cicsts_dataSource>
```

- defines the JNDI name that is used by your application when establishing a connection to that data source

# Liberty: local Db2: type 2 driver: in server.xml: Auto-Configure

Create a default CICS Db2 JDBC type 2 driver data source

Add to the JVM profile

-Dcom.ibm.cics.jvmserver.wlp.autoconfigure=true
-Dcom.ibm.cics.jvmserver.wlp.jdbc.driver.location=/usr/lpp/db2v12/jdbc

Install a JDBC feature

```
<featureManager>
    <feature>jdbc-4.2</feature>
</featureManager>
```

A default Liberty Db2 JDBC type 2 driver data source is added to the Liberty server configuration file
An auto configure for the CICS JDBC can be used (cicsts:jdbc-1.0)

```
<dataSource id="defaultCICSDataSource" jndiName="jdbc/defaultCICSDataSource" transactional="false">
    <jdbcDriver libraryRef="defaultCICSDb2Library"/>
    <properties.db2.jcc driverType="2"/>
    <connectionManager agedTimeout="0"/>
</dataSource>

<library id="defaultCICSDb2Library">
    <fileset dir="/usr/lpp/db2v12/jdbc/classes" includes="db2jcc4.jar db2jcc_license_cisuz.jar"/>
    <fileset dir="/usr/lpp/db2v12/jdbc/lib" includes="libdb2jcct2zos4_64.so"/>
</library>
```

# Java Code Example (DriverManager)

DriverManager example:

```
Connection connection = null;
try {
   connection =
    DriverManager.getConnection("jdbc:default:connection");
} catch (SQLException e) {
   e.printStackTrace();
}

if (connection != null) {
   try {
       // Do JDBC here
   :
```

# Java Code Example (DataSource)

DataSource example:

```
Connection connection = null;
try {
   InitialContext context = new InitialContext();
   DataSource dataSource =
      (DataSource) context.lookup("jdbc/defaultCICSDataSource");
   connection = dataSource.getConnection();
} catch (NamingException e) {
   e.printStackTrace();
} catch (SQLException e) {
   e.printStackTrace();
}

if (connection != null) {
   try {
       // Do JDBC here
    :
```

## Liberty: remote Db2: type 4 driver

The Liberty Db2 DataSource JDBC type 4 driver does not use the CICS Db2 connection resource

- if APARs PI18798 and PI18799 are NOT applied
  - need to add SDSNLOAD and SDSNLOD2 to the CICS STEPLIB

# Liberty: remote Db2: type 4 driver: in server.xml

Add the jdbc-4.x feature

For use with DataSource, specify

```
<server>
    <featureManager>
        <feature>jdbc-4.2</feature>
    </featureManager>

    <library id="db2Lib">
        <fileset dir="/usr/lpp/db2v12/jdbc/classes" includes="db2jcc4.jar
            db2jcc_license_cisuz.jar" />
    </library>

    <dataSource jndiName="jdbc/defaultCICSDataSource">
        <jdbcDriver libraryRef="db2Lib"/>
        <properties.db2.jcc driverType="4"
            serverName="example.db2.ibm.com"
            portNumber="41100"
            databaseName="DSNV11P2"
            user="DBUSER"
            password="{xor}Lz4sLCgwLTs="/>
    </dataSource>
</server>
```

It is recommended that you encode the password
- use the securityUtility tool with the encode option
- http://www.ibm.com/support/knowledgecenter/SS7K4U_8.5.5/com.ibm.websphere.wlp.zseries.doc/ae/rwlp_command_securityutil.html

# JTA: UserTransaction

```
InitialContext ctx = new InitialContext();

UserTransaction tran =
(UserTransaction)ctx.lookup("java:comp/UserTransaction");

DataSource ds = (DataSource)ctx.lookup("jdbc/defaultCICSDataSource");
Connection con = ds.getConnection();

// Start the User Transaction
tran.begin();
// Perform updates to CICS resources via JCICS API
// Access remote database ←
if (allOk) {
  // Commit updates on both systems
  tran.commit();
} else {
  // Backout updates on both systems
  tran.rollback();
}
```

# Differences between JDBC and SQLJ

JDBC
- Dynamic SQL – all SQL statements in JDBC are assembled at runtime
- Can improve performance by preparing the statement first
  - Improves performance if statement is used multiple times

SQLJ
- Static SQL – allows you to embed static SQL statements into Java programs
- Statements start with #sql
- Contains a serialized profile containing information about static SQL statements

Do the benefits of SQLJ remain?
- Pre-optimise dynamic SQL
- Gap has closed considerably
- Time investment to set up SQLJ vs. performance benefits?

# SQLJ application programming

In general, it can provide better performance because of its ability to use static SQL  (your result may vary)

Example code on page 292 of SG24-8038 – CICS and the JVM Server – Developing and Deploying Java Applications

Can

- Run the SQLJ command from USS command line to translate and compile the source code
- The SQLJ command generates Java source program, optionally it can compile the Java source program, and produce a serialized profile
- Run the db2sqljcustomize command from USS command line to customize the serialized profiles and bind Db2 packages

# Local JDBC and SQLJ in the CICS Db2 environment

In the CICS environment

- The Db2 JDBC driver is link-edited with the CICS Db2 language interface DSNCLI
- The driver converts the JDBC or SQLJ requests into their EXEC SQL equivalents
- Converted requests from the Db2-supplied JDBC driver flow into the CICS Db2 attachment facility in exactly the same way as EXEC SQL requests from any CICS program (e.g. COBOL)
- There is no operational difference between Java program for CICS Db2 and other programs for CICS Db2
- All customization and tuning options available using CICS RDO apply to Java programs for CICS Db2

Details on how to code and build Java applications that use JDBC and SQLJ are in Db2 for z/OS: Programming for Java (for your version of Db2).

# Java Samples

Supplied with the CICS Explorer and github

These samples help you get started with developing Java applications for CICS

CICS also supplies some sample groups and resource definitions

JCICS Examples

- packaged as a set of OSGi bundles
- you can import into a plug-in project to view the Java source code

Servlet Examples

- demonstrate best practice
- show how to use the JCICS API to develop web applications that can interact with CICS and Db2

# CICS Java Database Connectivity (JDBC)

This example includes

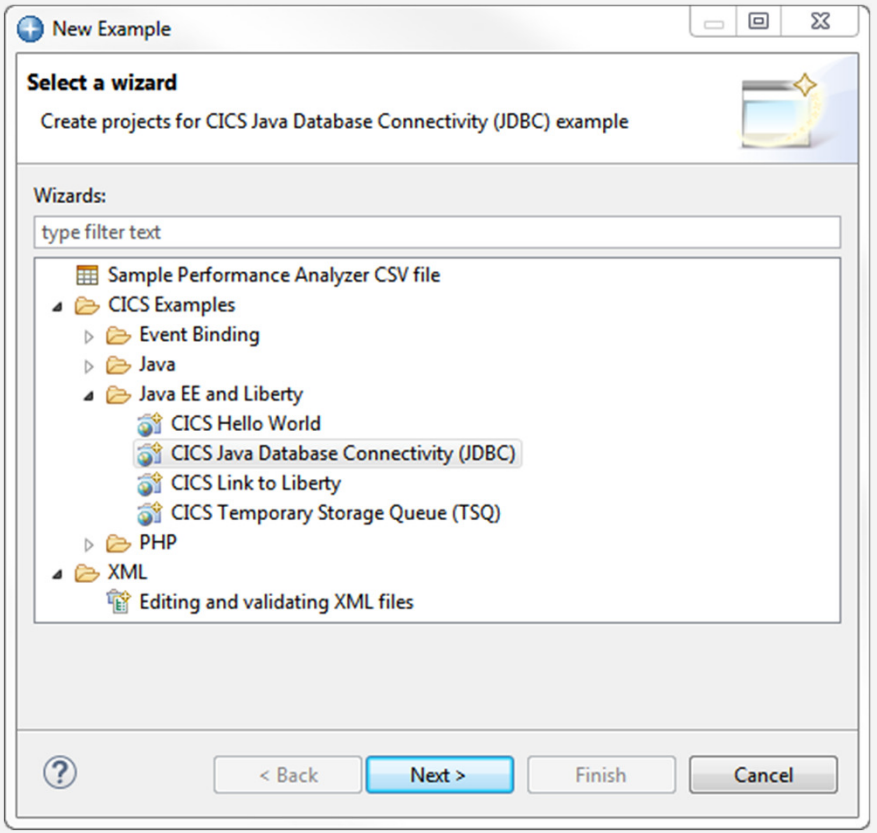
- an OSGi application project
- an OSGi bundle project
- a web-enabled OSGi bundle project
- a CICS bundle project

Provides a web front end to an existing JDBC sample

Demonstrates how to access Db2 data from a Java application in a Liberty JVM server using the CICS JDBC feature

# Creating the Servlet Examples

- Open the Java perspective
- Set the target platform
- In the Eclipse menu bar, click **File > New > Example**
- In the **CICS Examples > Java EE and Liberty** folder, select the example that you would like to use
- Click **Next (x2)**
  - The names of all the projects that are created for the selected example are listed
- Click **Finish**
  - The wizard creates the appropriate projects for the example you selected




https://www.ibm.com/docs/en/cics-ts/5.6?topic=examples-deploying-servlet

# Summary

**CICS and Db2 for Java**

**The Db2 API was not covered.  See:**

- SG24-8038 – CICS and the JVM Server – Developing and Deploying
- Various Db2 manuals
- The Internet

- **Note:** look at SG24-8038 or the Db2 manuals for examples of coding JDBC or SQLJ