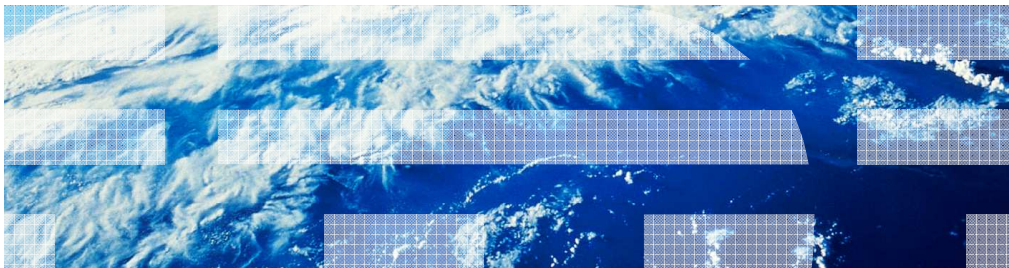


CICS Transaction Server V4.2

Java end-to-end scenario



© 2011 IBM Corporation

CICS Transaction Server Version 4.2 has significantly enhanced its Java support to provide an end-to-end environment where applications can be developed and deployed using the CICS Explorer SDK and managed from CICS Explorer. This scenario takes you through the step by step process from setting up Java support as a system programmer, creating and deploying an application as an application developer, and finally running the application in CICS.

Scenario: Setting up and running Java applications

- **Goals**
 - Run a Java application in CICS
 - Demonstrate different development and management roles
 - End-to-end experience with CICS Explorer SDK
- **Dave the Developer**
 - Create, deploy, and test the application
 - Can troubleshoot application problems
- **Steve the System Programmer**
 - Create the Java environment
 - Install and manage the application
 - Can troubleshoot application dependency problems
- **Setup**
 - Mac OS X 10.6 + Java SE 6 (Update 26) + Eclipse 3.6.2 (Cocoa 64-bit) + CICS Explorer SDK V1.1
 - CICS TS 4.2 CICSplex
 - Java and CICS SM perspectives in separate workbench windows

This module describes an end-to-end scenario of how to set up Java, create and deploy a sample Java application, and run it in CICS. It aims to show the roles involved and how to perform the steps using CICS Explorer and the CICS Explorer SDK. The scenario uses two roles, Dave the Developer and Steve the System Programmer. Dave is responsible for creating, deploying, and testing the Java application. He can troubleshoot any application problems. Steve is responsible for setting up the Java environment in CICS, installing and managing the application. He can troubleshoot any application dependency problems. This scenario uses a workstation with Mac OS X 10.6 with Java Standard Edition Version 6 and Eclipse 3.6.2 with the CICS Explorer SDK Version 1.1 installed. The CICS regions are running CICS TS 4.2 and can be access from the CICS Explorer. The Java and CICS SM perspectives are in separate workbench windows.

Steve the system programmer

1. Configure and install JVM server
2. Create and install (sample) CICS resources
3. Create CICS bundle



Steve the system programmer has to configure and install the JVM server in the CICS region. He has to create and install the required CICS resources. In this scenario, the samples supplied with CICS are used. Finally, Steve has to create a CICS bundle to install the application in the JVM server.

Open the supplied JVM server profile in CICS Explorer

The screenshot displays the CICS Explorer interface in z/OS perspective. The left-hand pane shows a tree view of CICS regions, with the 'JVM Server' section expanded. The center pane shows the file explorer for the path `/u/mqtest/jvmpr (205)`, listing various JVM server profiles such as `DFHJVMCD`, `DFHJVMERR`, `DFHJVMOUT`, `DFHJVMR`, `DFHOSGI`, `DFHOSG1`, `DFHOSG2`, `DFHOSG3`, `DFHOSG4`, `DFHOSG5`, `DFHOSG6`, `DFHOSG7`, `DFHOSG8`, `DFHOSG9`, `DFHOSG10`, `DFHOSG11`, `DFHOSG12`, `DFHOSG13`, `DFHOSG14`, `DFHOSG15`, `DFHOSG16`, `DFHOSG17`, `DFHOSG18`, `DFHOSG19`, `DFHOSG20`, `DFHOSG21`, `DFHOSG22`, `DFHOSG23`, `DFHOSG24`, `DFHOSG25`, `DFHOSG26`, `DFHOSG27`, `DFHOSG28`, `DFHOSG29`, `DFHOSG30`, `DFHOSG31`, `DFHOSG32`, `DFHOSG33`, `DFHOSG34`, `DFHOSG35`, `DFHOSG36`, `DFHOSG37`, `DFHOSG38`, `DFHOSG39`, `DFHOSG40`, `DFHOSG41`, `DFHOSG42`, `DFHOSG43`, `DFHOSG44`, `DFHOSG45`, `DFHOSG46`, `DFHOSG47`, `DFHOSG48`, `DFHOSG49`, `DFHOSG50`, `DFHOSG51`, `DFHOSG52`, `DFHOSG53`, `DFHOSG54`, `DFHOSG55`, `DFHOSG56`, `DFHOSG57`, `DFHOSG58`, `DFHOSG59`, `DFHOSG60`, `DFHOSG61`, `DFHOSG62`, `DFHOSG63`, `DFHOSG64`, `DFHOSG65`, `DFHOSG66`, `DFHOSG67`, `DFHOSG68`, `DFHOSG69`, `DFHOSG70`, `DFHOSG71`, `DFHOSG72`, `DFHOSG73`, `DFHOSG74`, `DFHOSG75`, `DFHOSG76`, `DFHOSG77`, `DFHOSG78`, `DFHOSG79`, `DFHOSG80`, `DFHOSG81`, `DFHOSG82`, `DFHOSG83`, `DFHOSG84`, `DFHOSG85`, `DFHOSG86`, `DFHOSG87`, `DFHOSG88`, `DFHOSG89`, `DFHOSG90`, `DFHOSG91`, `DFHOSG92`, `DFHOSG93`, `DFHOSG94`, `DFHOSG95`, `DFHOSG96`, `DFHOSG97`, `DFHOSG98`, `DFHOSG99`, `DFHOSG100`.

The right-hand pane shows the configuration file for `DFHOSGI`, which includes parameters for `JAVA_HOME`, `current working directory`, and `bundle paths`.

```

# Required parameters
#
# The set of supported CICS options for JVM servers
# differs from those used with JVM pools.
#
# JAVA_HOME specifies the location of the Java directory.
#
#JAVA_HOME=/usr/java/jre/1.6
#
# Set the current working directory. If this environment variable is
# set, it changes the current directory to the specified directory
# before the JVM
# is initialized, and the STDIN, STDOUT and STDERR streams are
# allocated to this directory.
#
# If you do not specify this option, the current working directory is
# left unchanged and the STDIN, STDOUT and STDERR streams are allocated
# to the /tmp directory.
#
#WORK_DIR=
#
# Specify any directories that contain DLLs required at run time. For
# example, to use the IBM DB2 Server for iSeries and SQLI, add the
# directory containing the native DLLs to the LIBPATH SUFFIX option.
# See the "Application Programming Guide and Reference for Java" relevant
# to the level of DB2 being used.
#
#LIBPATH_SUFFIX=
#
#-----
#
# JVM server specific parameters
#-----
#
# Use the DSGL_BUNDLES option to specify a list of additional
# bundles that are installed and activated in the DSGL framework
# when the JVM is initialized.
# The list of bundles must be comma separated.
#
#DSGL_BUNDLES=/u/example/pathToBundleDirectory/EL.jar,
/u/example/pathToBundleDirectory/BD.jar

```

In the CICS Explorer, open the z/OS perspective to edit the supplied JVM server profile DFHOSGI for the appropriate CICS region.

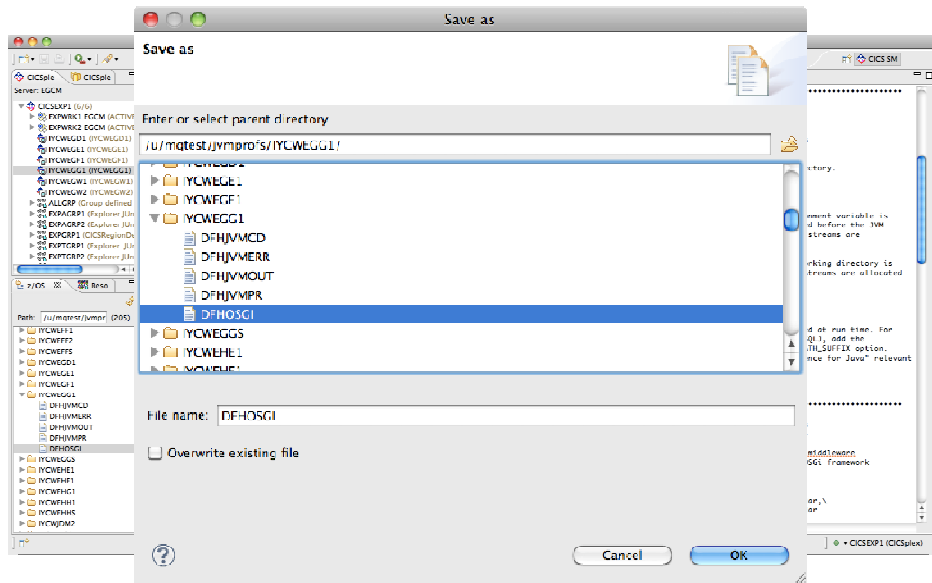
Edit supplied JVM server profile in CICS Explorer

```

#
# JAVA_HOME specifies the location of the Java directory.
#
#JAVA_HOME=/usr/lib/java/J6.0/
#JAVA_HOME=/java/java601_bit64_GA/J6.0.1_64
#
# Set the current working directory. If this environment variable is
# set, a change to the specified directory is issued before the JVM
# is initialized, and the STDIN, STDOUT and STDERR streams are
# allocated to this directory.
#
# If you do not specify this option, the current working directory is
# left unchanged and the STDIN, STDOUT and STDERR streams are allocated
# to the /tmp directory.
#
#WORK_DIR=.
WORK_DIR=/u/webster/CICSEXP1
  
```

The profile contains the options to start the JVM server. The Java home option specifies the location of Java. The value is set during the installation of CICS. You can optionally change the working directory of the JVM server if required.

Save as a new profile



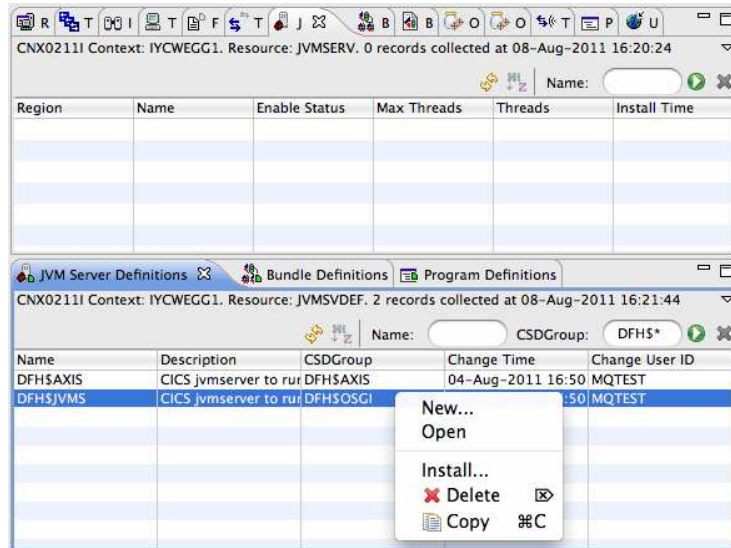
Select File -> Save as from the menu to save a copy of the profile as a new file.

Find the JVM profile directory

Property	Value
Auto Install	
Auto Install Consoles	AUTO
Basic	
CICS SVC	240
CPSM Connection	LMAS
GM Text	'WELCOME TO EXPLORER LMAS CICS 670, REGION IVCWEGG1'
SIT Suffix	PE
SRBSVC	241
Start	INITIAL
SYSIDNT	EGG1
USS Home	/cics/cics670
CSD	
Auto Install Group Lists	(C42E67)
CSD Concurrent Requests	6
CSD Disposition	SHR
CSD Forward Recovery Log	1
CSD Journal ID	1
CSD Recovery	ALL
CSD RLS	YES
Dump	
System Dump Max	5
File Control	
RLS	YES
Java	
JVM Profile Directory	/u/mqtest/jvmprofs/IVCWEGG1
Network	
Application ID	IVCWEGG1
Intersystem Communication	YES
IRC Start	YES
TCP/IP	YES

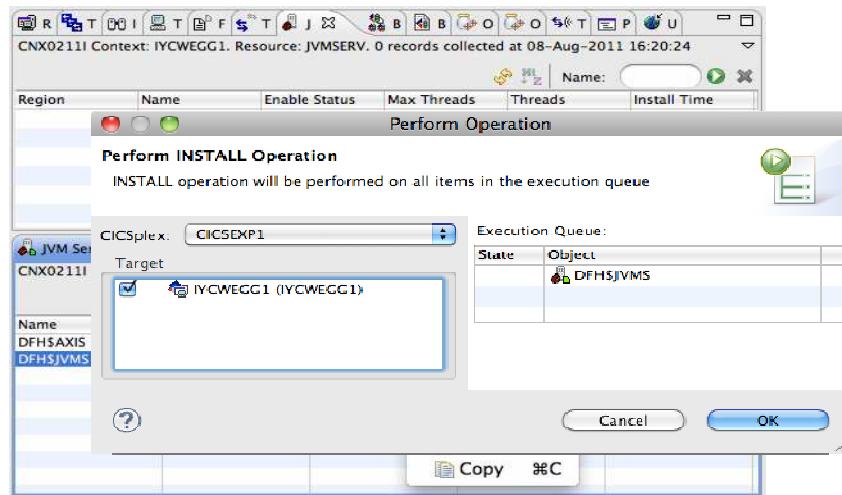
Move the JVM profile into the JVM profile directory. Use the Show SIT parameters menu option in CICS Explorer to find the JVM profile directory for the CICS region.

Install the JVMSERVER resource



In the CICS SM perspective, create a JVMSERVER resource definition and install it. In this scenario, the sample resource definition DFH\$JVMS is used. This resource is already configured to point to the sample profile DFHOSGI. Right-click the definition to install it.

Select the target region



Select the target region where you want to install the resource.

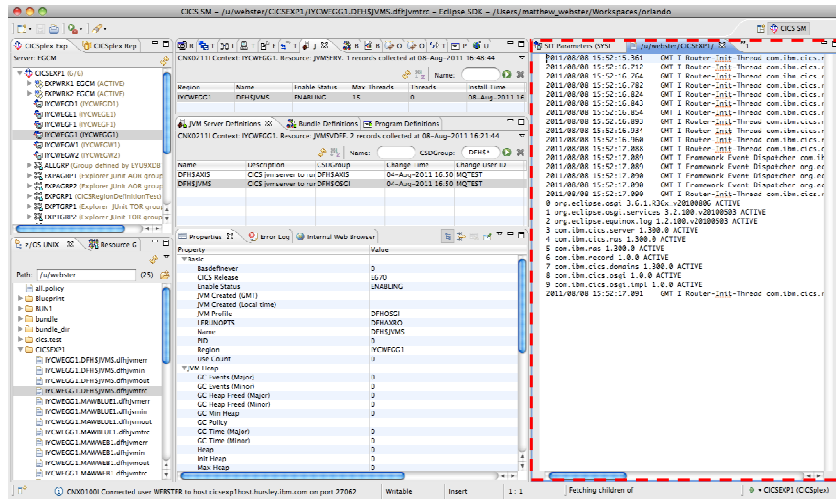
JVM server DFH\$JVMS is enabled

CNX0211I Context: IYCWEGG1. Resource: JVMSEV. 1 records collected at 08-Aug-2011 16:48:44

Region	Name	Enable Status	Max Threads	Threads	Install Time
IYCWEGG1	DFH\$JVMS	✓ ENABLED	15	0	08-Aug-2011 16

You can check the JVM server is enabled by going to the Operations->Java->JVM Servers view. Additional information is also available in the logs for the JVM server.

Check the logs for any problems



The log files are available in the working directory of the JVM server. The working directory is set in the JVM server profile. The JVM writes to standard in, out, and error files in the working directory and writes to a trace file called dfhjvmtrc. These files are prefixed with the APPLID of the CICS region and the name of the JVM server for easy identification.

Contents of the trace file

```

/u/mqtest/jvmprofs/IYCWEGG1/DFHOSGI  SIT Parameters (SYSIN)  /u/webster/CICSEXP1/IYCWEGG1.DFHJVMS.dfhjvmtrc
2011/08/08 15:52:15.361 GMT I Router-Init-Thread com.ibm.cics.router.Router: Starting the OSGi Framework.
2011/08/08 15:52:16.712 GMT I Router-Init-Thread com.ibm.cics.router.Router: Installing OSGi System bundle: /cic
2011/08/08 15:52:16.764 GMT I Router-Init-Thread com.ibm.cics.router.Router: Installing OSGi System bundle: /cic
2011/08/08 15:52:16.782 GMT I Router-Init-Thread com.ibm.cics.router.Router: Installing CICS System bundle: /cic
2011/08/08 15:52:16.824 GMT I Router-Init-Thread com.ibm.cics.router.Router: Installing CICS System bundle: /cic
2011/08/08 15:52:16.843 GMT I Router-Init-Thread com.ibm.cics.router.Router: Installing CICS System bundle: /cic
2011/08/08 15:52:16.854 GMT I Router-Init-Thread com.ibm.cics.router.Router: Installing CICS System bundle: /cic
2011/08/08 15:52:16.893 GMT I Router-Init-Thread com.ibm.cics.router.Router: Installing CICS System bundle: /cic
2011/08/08 15:52:16.934 GMT I Router-Init-Thread com.ibm.cics.router.Router: Installing CICS System bundle: /cic
2011/08/08 15:52:16.960 GMT I Router-Init-Thread com.ibm.cics.router.Router: Installing CICS System bundle: /cic
2011/08/08 15:52:17.088 GMT I Router-Init-Thread com.ibm.cics.osgi.impl [com.ibm.cics.osgi.CICSController, com.i
2011/08/08 15:52:17.089 GMT I Framework Event Dispatcher com.ibm.cics.osgi.impl: BundleEvent STARTED
2011/08/08 15:52:17.089 GMT I Framework Event Dispatcher org.eclipse.osgi: BundleEvent STARTED
2011/08/08 15:52:17.090 GMT I Framework Event Dispatcher org.eclipse.osgi: FrameworkEvent STARTED
2011/08/08 15:52:17.090 GMT I Framework Event Dispatcher org.eclipse.osgi: FrameworkEvent STARTLEVEL CHANGED
2011/08/08 15:52:17.090 GMT I Router-Init-Thread com.ibm.cics.router.Router: Bundles:
0 org.eclipse.osgi 3.6.1.R36x.v20100806 ACTIVE
1 org.eclipse.osgi.services 3.2.100.v20100503 ACTIVE
2 org.eclipse.equinox.log 1.2.100.v20100503 ACTIVE
3 com.ibm.cics.server 1.300.0 ACTIVE
4 com.ibm.cics.ras 1.300.0 ACTIVE
5 com.ibm.ras 1.300.0 ACTIVE
6 com.ibm.record 1.0.0 ACTIVE
7 com.ibm.cics.domains 1.300.0 ACTIVE
8 com.ibm.cics.osgi 1.0.0 ACTIVE
9 com.ibm.cics.osgi.impl 1.0.0 ACTIVE
2011/08/08 15:52:17.091 GMT I Router-Init-Thread com.ibm.cics.router.Router: OSGi Framework started.

```

The trace file provides details about the startup of the JVM server and the initialization of the OSGi framework. In this example, the OSGi bundles that are required by the system are listed as they start.

JVM server information

Property	Value
Basic	
BaseDefNew	0
CICS Release	E570
Enable Status	ENABLING
JVM Created (GMT)	
JVM Created (Local time)	
JVM Profile	DFHOSGI
LERUNOPTS	DFHAXRO
Name	DFH5JVMS
PID	0
Region	IYCWECC1
Use Count	0
JVM Heap	
GC Events (Major)	0
GC Events (Minor)	0
GC Heap Freed (Major)	0
GC Heap Freed (Minor)	0
GC Min Heap	0
GC Policy	
GC Time (Major)	0
GC Time (Minor)	0
Heap	0
Init Heap	0
Max Heap	0
Peak Heap	0
Resource Signature	
Threads	
Max Threads	15
Peak Threads	0
Peak Waiting Threads	0
Sys Thread Peak Wait	0
Sys Thread Usage	8
Sys Thread Waits	0
Sys Thread Wait Time	0
Threads	0
Thread Waits	0
Thread Wait Time	0
Waiting Threads	0
Waiting Threads (Sys Thread)	0

The enabled JVMSERVER resource has lots of useful information about the underlying JVM, including garbage collection information and thread usage. When the JVMSERVER resource is enabled, the JVM server is ready for applications.

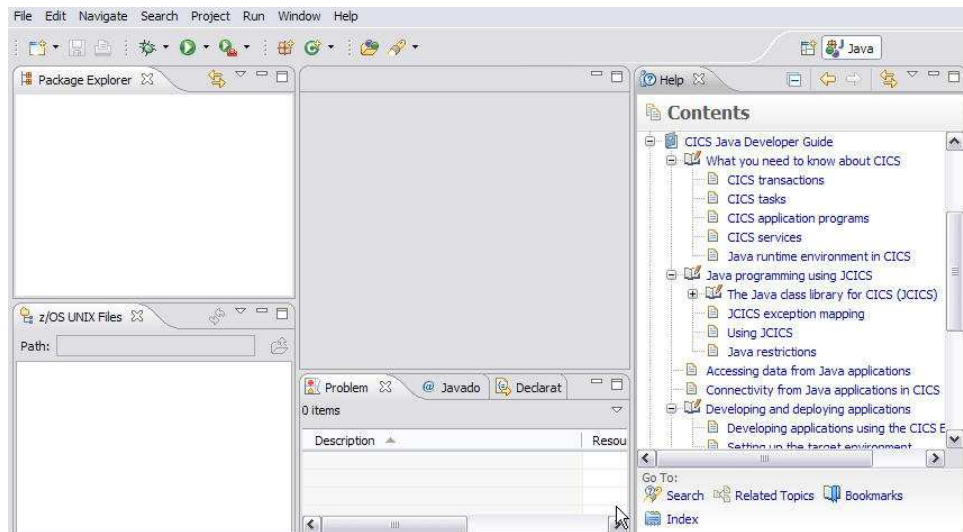
Dave the application developer

1. Consult CICS Java Developer Guide
2. Set Target Platform
3. Create application
4. Deploy application as CICS bundle



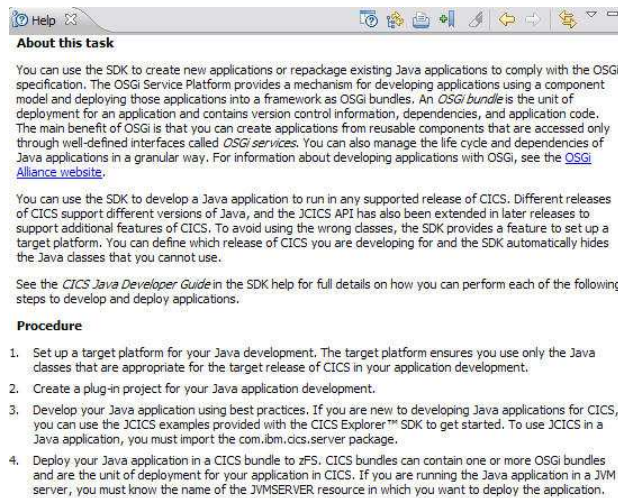
Dave the application developer has four main tasks. The first is to consult the CICS Java Developer Guide in the CICS Explorer SDK help. This guide contains information about developing Java applications for CICS. Dave then has to configure the SDK to provide a target platform. The target platform defines what release of CICS the application is going to run in. Setting a target platform ensures that Dave doesn't use any JCICS API that is unavailable in that release of CICS. Dave can then develop the application and deploy the application as a CICS bundle to test it.

Starting point in the CICS Explorer SDK



The starting point in the CICS Explorer SDK is an empty workspace with no projects. The help with the CICS Java Developer Guide is also shown in the screen capture and includes the JCICS Javadoc information.

Developing applications using the CICS Explorer SDK



About this task

You can use the SDK to create new applications or repackage existing Java applications to comply with the OSGi specification. The OSGi Service Platform provides a mechanism for developing applications using a component model and deploying those applications into a framework as OSGi bundles. An *OSGi bundle* is the unit of deployment for an application and contains version control information, dependencies, and application code. The main benefit of OSGi is that you can create applications from reusable components that are accessed only through well-defined interfaces called *OSGi services*. You can also manage the life cycle and dependencies of Java applications in a granular way. For information about developing applications with OSGi, see the [OSGi Alliance website](#).

You can use the SDK to develop a Java application to run in any supported release of CICS. Different releases of CICS support different versions of Java, and the JCICS API has also been extended in later releases to support additional features of CICS. To avoid using the wrong classes, the SDK provides a feature to set up a target platform. You can define which release of CICS you are developing for and the SDK automatically hides the Java classes that you cannot use.

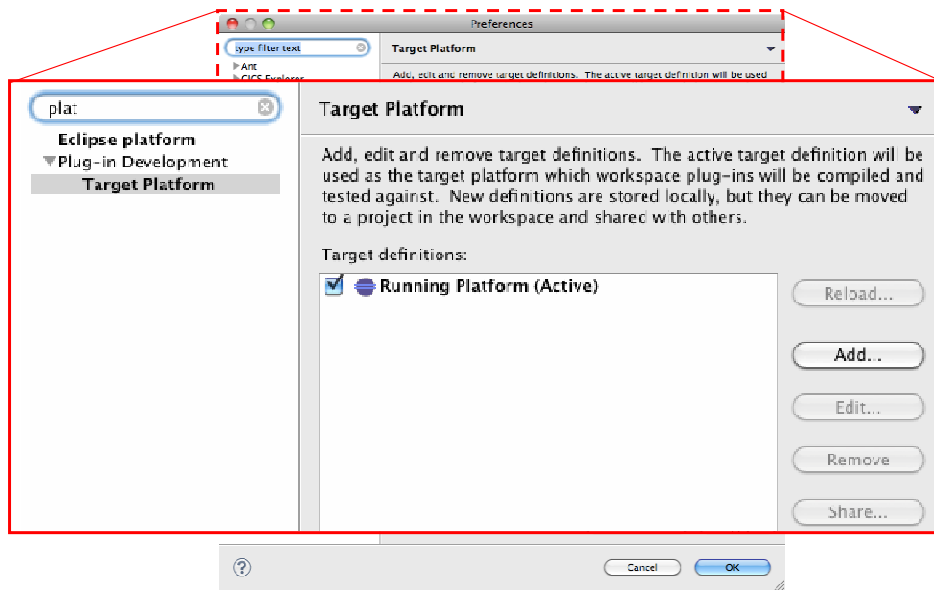
See the *CICS Java Developer Guide* in the SDK help for full details on how you can perform each of the following steps to develop and deploy applications.

Procedure

1. Set up a target platform for your Java development. The target platform ensures you use only the Java classes that are appropriate for the target release of CICS in your application development.
2. Create a plug-in project for your Java application development.
3. Develop your Java application using best practices. If you are new to developing Java applications for CICS, you can use the JCICS examples provided with the CICS Explorer™ SDK to get started. To use JCICS in a Java application, you must import the `com.ibm.cics.server` package.
4. Deploy your Java application in a CICS bundle to zFS. CICS bundles can contain one or more OSGi bundles and are the unit of deployment for your application in CICS. If you are running the Java application in a JVM server, you must know the name of the JVMSERVER resource in which you want to deploy the application.

The help has full details about how to develop Java applications for CICS in the CICS Explorer SDK. The procedure has four steps, which this scenario will demonstrate.

Set target platform



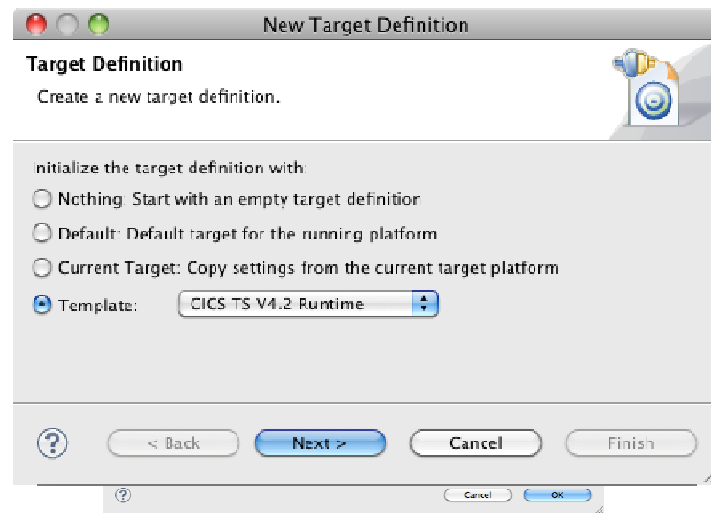
17

Java end-to-end scenario

© 2011 IBM Corporation

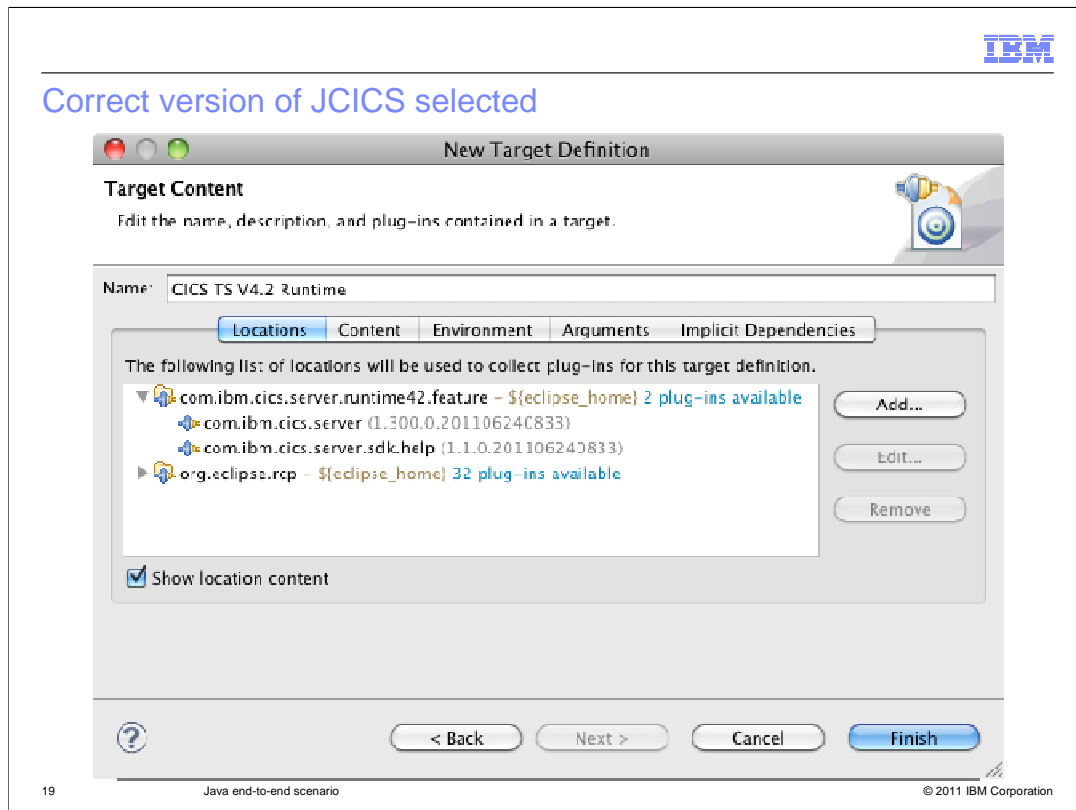
Open the preferences in the Eclipse IDE and open the Plug-in Development section to select the target platform. Click Add to create a new target platform.

Create a new target platform



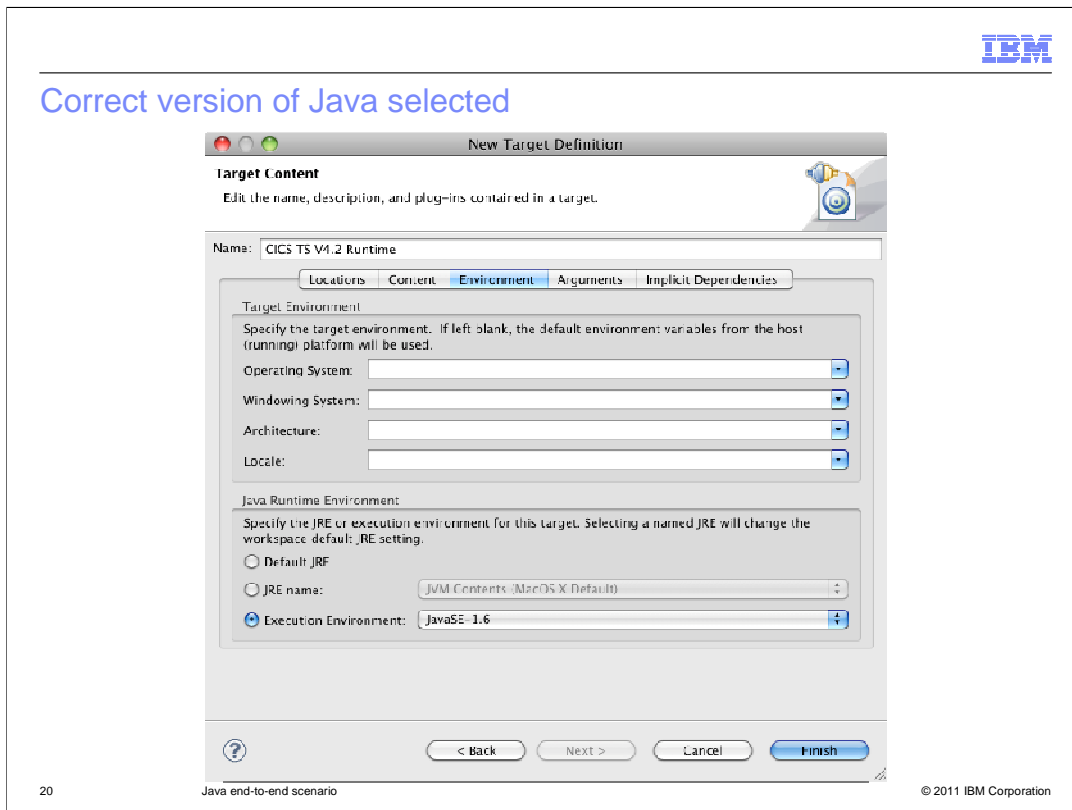
Select the appropriate release of CICS from the list in the templates.

Correct version of JCICS selected



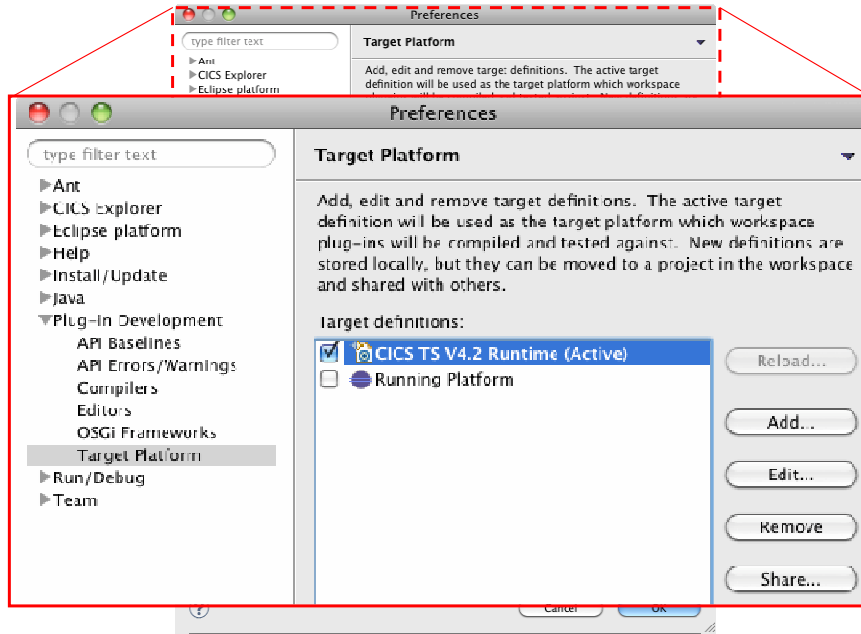
The template specifies the correct version of JCICS for CICS TS 4.2.

Correct version of Java selected



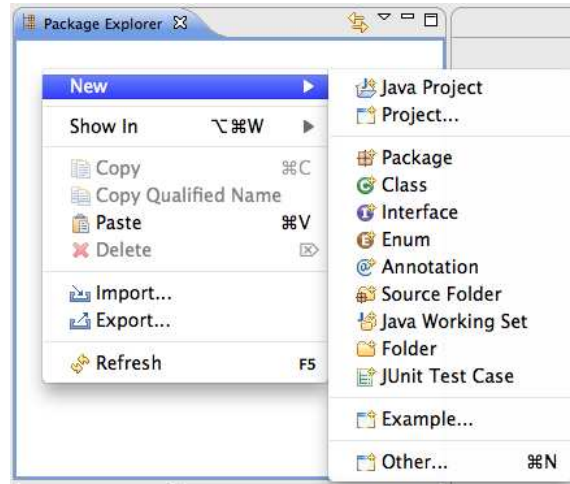
The environment contains the correct version of Java.

Select the new target platform



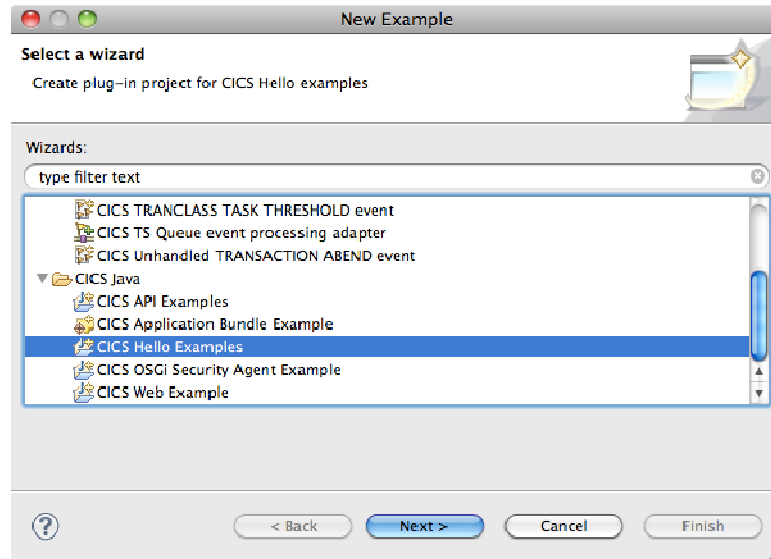
Select the newly-created target platform.

Add the examples to the workspace



Add the Hello world, JCICS, and web examples to the workspace by selecting New -> Example.

Add OSGi bundle projects



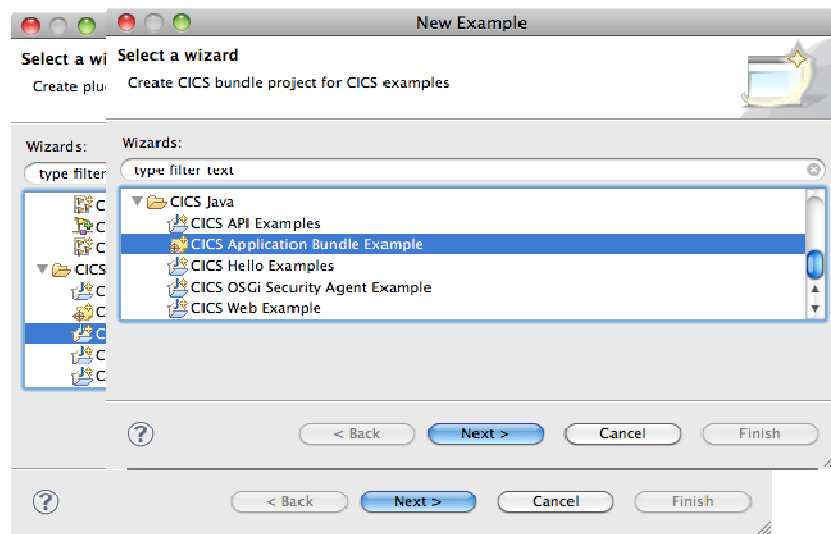
23

Java end-to-end scenario

© 2011 IBM Corporation

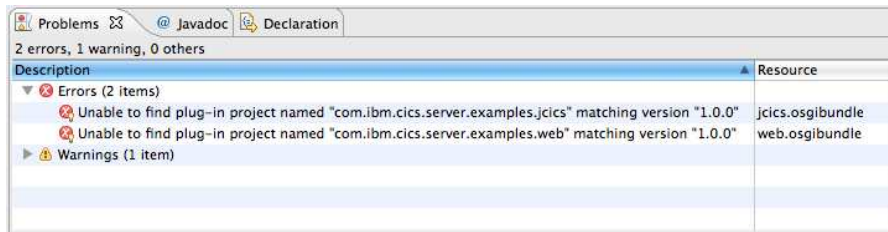
In the CICS Java section, select CICS Hello Examples from the list to import the Hello World OSGi bundle into the Package Explorer. You can also import example OSGi bundles for the JCICS API and web. The security agent example is available if you want to set up a Java security manager.

Add CICS bundle project



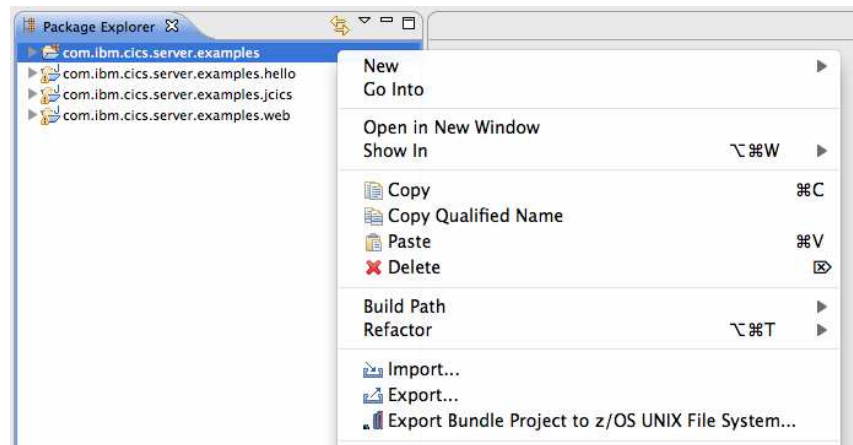
Select the CICS Application Bundle Example to create a bundle project in the Package Explorer view. The CICS bundle project is the unit of deployment to deploy applications in CICS.

Errors in the Problems view



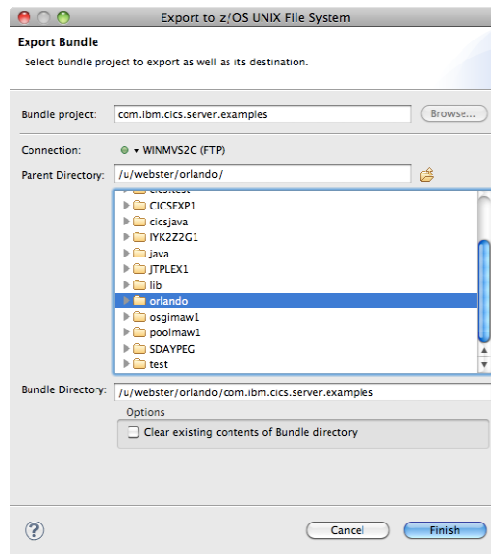
If there are any problems with the projects, these are reported in the Problems view. Here the JCICS API and web example OSGi bundles have not been imported into the workspace.

Deploy application as a CICS bundle



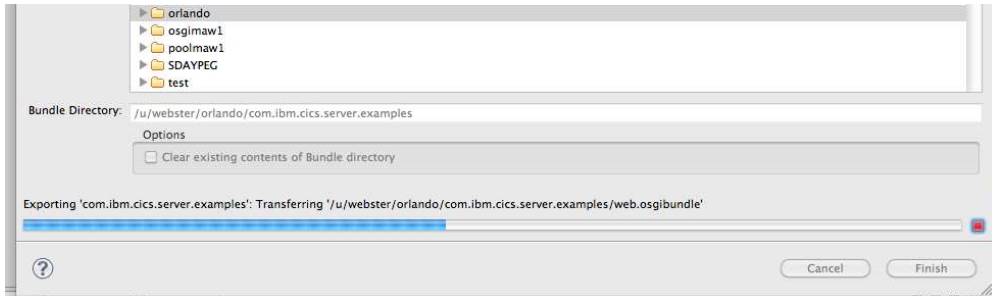
Right-click the CICS bundle project to export it to a suitable directory in zFS.

Select CICS bundles directory



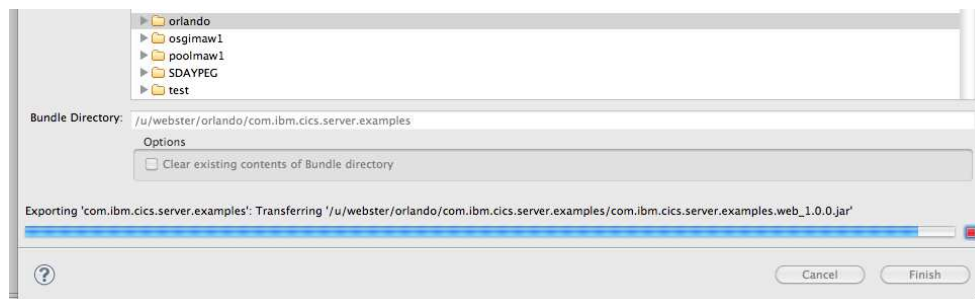
Select an appropriate directory to deploy the CICS bundle.

Transfer metadata files



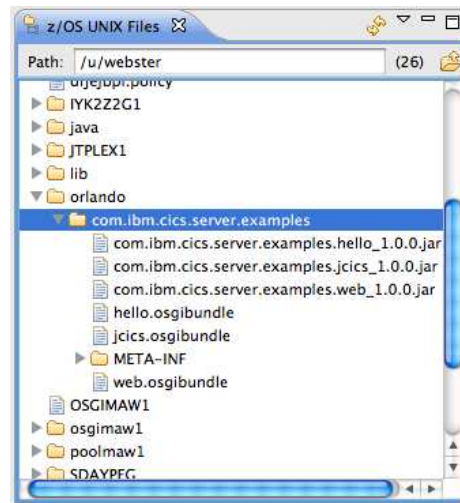
The CICS bundle is deployed to zFS, including the metadata files that describe the OSGi bundles.

Builds, locally exports, and transfers OSGi bundles



The Explorer SDK also builds the Java, locally exports and transfers the OSGi bundles as JAR files to the CICS bundle directory in zFS.

Handover CICS bundle to Steve



At this point, Dave the Application Developer hands over the deployed CICS bundle to Steve the system programmer.

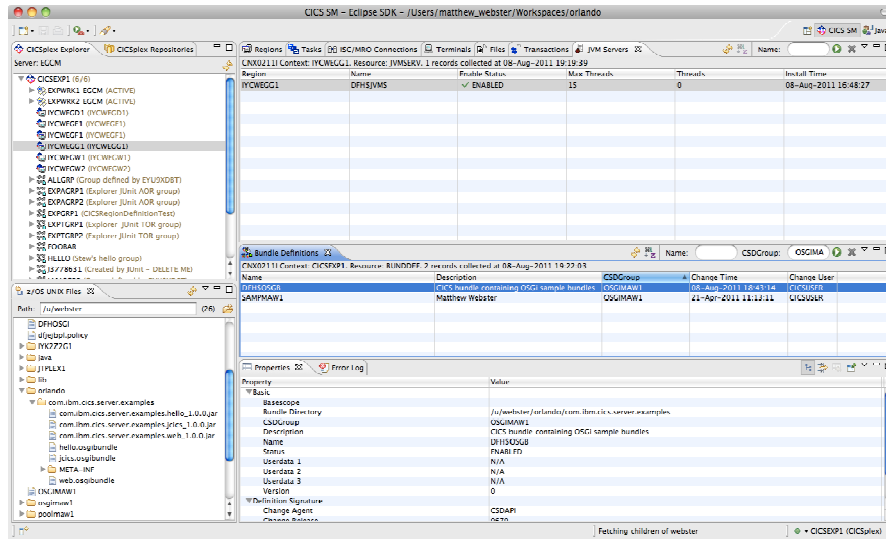
Install the application in the JVM server

1. Install application as CICS bundle
2. Validate OSGi bundles and services
3. Ensure TCP/IP service and URI Map available
4. Run the application




Steve the system programmer has the authority to install the application as a CICS bundle in the CICS region that has a running JVM server. Steve has to validate that the OSGi bundles and services install successfully, that the application is available from the web, and run the application.

Handover CICS bundle from Dave



Steve creates a BUNDLE resource definition in the CICS SM perspective that points to the bundle directory created by Dave.

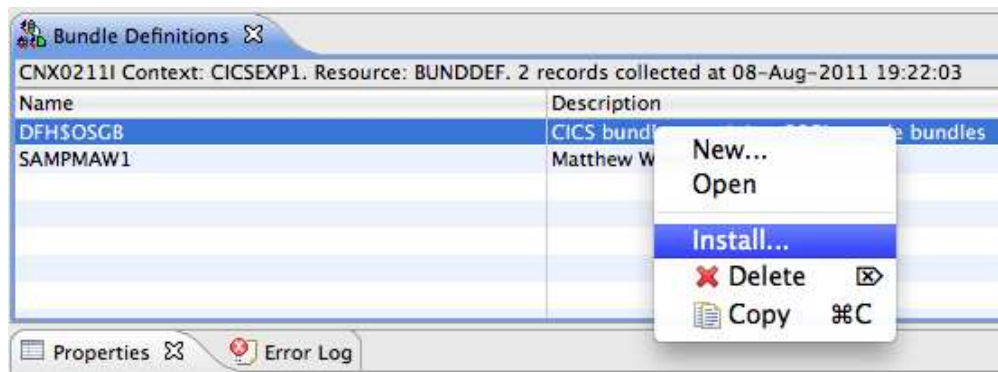
BUNDLE definition properties



Property	Value
▼ Basic	
Basescope	
Bundle Directory	/u/webster/orlando/com.ibm.cics.server.examples
CSDGroup	OSGIMAW1
Description	CICS bundle containing OSGi sample bundles
Name	DFH\$OSGB
Status	ENABLED
Userdata 1	N/A
Userdata 2	N/A
Userdata 3	N/A
Version	0

The BUNDLE definition points to the bundle directory in zFS. In this example, the sample BUNDLE definition DFH\$OSGB is used.

Install the CICS BUNDLE resource



Right-click the definition to install the CICS BUNDLE resource. When you install and enable the CICS BUNDLE resource, CICS creates the OSGi bundles and services in the JVM server.

Check the bundle and bundle parts

The top screenshot shows the 'Bundles' view in the IBM WebSphere Administration Console. The context is 'CNX02111 Context: IYCWEGG1. Resource: BUNDLE. 1 records collected at 08-Aug-2011 20:04:19'. The table below shows the bundle details:

Region	Name	Status	Install Time
IYCWEGG1	DFHSOSGB	✓ ENABLED	08-Aug-2011 19:36:46

The bottom screenshot shows the 'Bundle Parts' view for the same region. The context is 'CNX02111 Context: IYCWEGG1. Resource: BUNDPART. 3 records collected at 08-Aug-2011 20:04:31'. The table below shows the bundle parts:

Region	Bundle	Bundle Part	Enable Status	Meta Data File	Part Class	Part Type
IYCWEGG1	DFHSOSGB	hello	ENABLED	hello.osgibundle	DEFINITION	http://www.ibm.com/xml
IYCWEGG1	DFHSOSGB	jcics	ENABLED	jcics.osgibundle	DEFINITION	http://www.ibm.com/xml
IYCWEGG1	DFHSOSGB	web	ENABLED	web.osgibundle	DEFINITION	http://www.ibm.com/xml

Check that the BUNDLE resource and its bundle parts are enabled. You can check the BUNDLE resources in the Operations -> Bundles view. The bundle parts show the resources that are included in the CICS bundle. Each bundle part is enabled, meaning that CICS was able to successfully create a dynamic resource for each definition in the manifest of the bundle.

OSGi bundle for web example

JVM Server	Symbolic Name	Version	State	Bundle	Bundle Part
DFH\$JVMS	com.ibm.cics.server.examples.hello	1.0.0	✓ ACTIVE	DFH\$OSGB	hello
DFH\$JVMS	com.ibm.cics.server.examples.jcics	1.0.0	✓ ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	com.ibm.cics.server.examples.web	1.0.0	✓ ACTIVE	DFH\$OSGB	web

The OSGi Bundles view shows the OSGi bundles that are running in the JVM server. It includes the name and version information for each bundle, its state in the OSGi framework, and the BUNDLE resource in which it is contained. In this screen capture, all three examples of OSGi bundles are active in the OSGi framework.

OSGi service for web example

The screenshot shows two views from the IBM WebSphere console. The top view is 'OSGi Bundles' for context IYCWEGG1, showing three active bundles. The bottom view is 'OSGi Services' for context OSGISERV, showing nine active services.

JVM Server	Symbolic Name	Version	State	Bundle	Bundle Part
DFH\$JVMS	com.ibm.cics.server.examples.hello	1.0.0	ACTIVE	DFH\$OSGB	hello
DFH\$JVMS	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	com.ibm.cics.server.examples.web	1.0.0	ACTIVE	DFH\$OSGB	web

JVM Server	Service Name	OSGi Bundle	Version	Service Status	Bundle	Bundle Part
DFH\$JVMS	examples.hello.HelloCICSWorld	com.ibm.cics.server.examples.hello	1.0.0	ACTIVE	DFH\$OSGB	hello
DFH\$JVMS	examples.hello.HelloWorld	com.ibm.cics.server.examples.hello	1.0.0	ACTIVE	DFH\$OSGB	hello
DFH\$JVMS	examples.ProgramControl.ClassOne	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	examples.ProgramControl.ClassTwo	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	examples.ProgramControl.ClassThree	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	examples.ProgramControl.ClassFour	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	examples.TDQ.ClassOne	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	examples.TSQ.ClassOne	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	examples.Web_Sample1	com.ibm.cics.server.examples.web	1.0.0	ACTIVE	DFH\$OSGB	web

The OSGi services view shows the services that are registered in the OSGi framework of the JVM server for each OSGi bundle. It is possible for an OSGi bundle to have more than one service. In this screen capture, all the services that are defined for the three example OSGi bundles are active.

CICS PROGRAM for web example

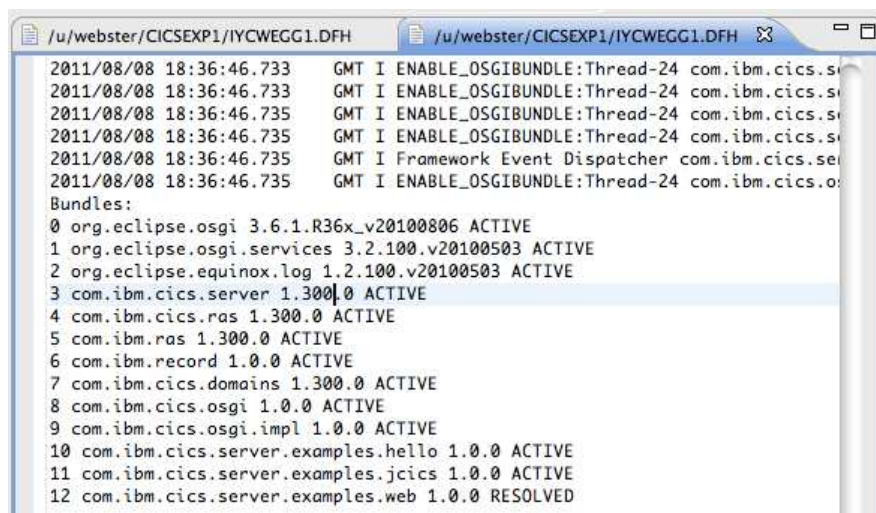
The screenshot shows two windows from the IBM CICS Explorer interface. The top window displays the 'PROGRAM' resource for region IYCWEGG1, and the bottom window displays the 'OSGi Services' for JVM Servers.

Region	Name	Status	Use Count	Concurrent Use Count	Language	JVM Class
IYCWEGG1	DFJ\$JHE1	✓ ENABLED	0	0	JAVA	examples.hello.HelloWorld
IYCWEGG1	DFJ\$JHE2	✓ ENABLED	0	0	JAVA	examples.hello.HelloCICSWorld
IYCWEGG1	DFJ\$JWB1	✓ ENABLED	0	0	JAVA	examples.Web.Sample1

JVM Server	Service Name	OSGi Bundle	Version	Service Status	Bundle	Bundle Part
DFH\$JVMS	examples.hello.HelloCICSWorld	com.ibm.cics.server.examples.hello	1.0.0	ACTIVE	DFH\$OSGB	hello
DFH\$JVMS	examples.hello.HelloWorld	com.ibm.cics.server.examples.hello	1.0.0	ACTIVE	DFH\$OSGB	hello
DFH\$JVMS	examples.ProgramControl.ClassOne	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	examples.ProgramControl.ClassTwo	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	examples.ProgramControl.ClassThree	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	examples.ProgramControl.ClassFour	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	examples.TDQ.ClassOne	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	examples.TSQ.ClassOne	com.ibm.cics.server.examples.jcics	1.0.0	ACTIVE	DFH\$OSGB	jcics
DFH\$JVMS	examples.Web.Sample1	com.ibm.cics.server.examples.web	1.0.0	ACTIVE	DFH\$OSGB	web

To test the web example, the sample program DFJ\$JWB1 has to be installed and enabled. The PROGRAM resource defines the JVM server and the JVM class for the web sample.

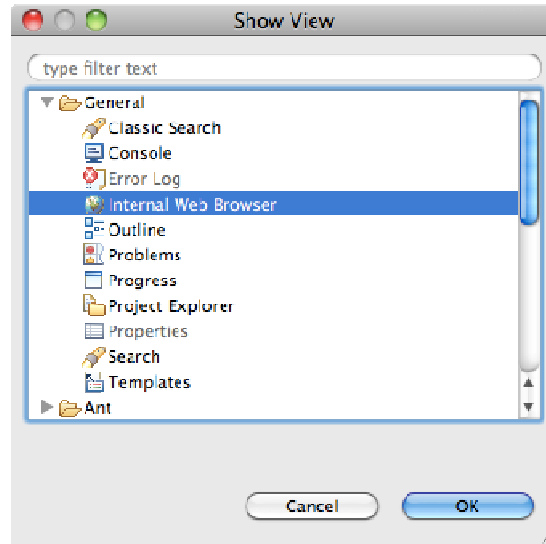
Check the log files for any problems



```
/u/webster/CICSEXP1/IYCWEGG1.DFH /u/webster/CICSEXP1/IYCWEGG1.DFH
2011/08/08 18:36:46.733 GMT I ENABLE_OSGIBUNDLE:Thread-24 com.ibm.cics.s
2011/08/08 18:36:46.733 GMT I ENABLE_OSGIBUNDLE:Thread-24 com.ibm.cics.s
2011/08/08 18:36:46.735 GMT I ENABLE_OSGIBUNDLE:Thread-24 com.ibm.cics.s
2011/08/08 18:36:46.735 GMT I ENABLE_OSGIBUNDLE:Thread-24 com.ibm.cics.s
2011/08/08 18:36:46.735 GMT I Framework Event Dispatcher com.ibm.cics.se
2011/08/08 18:36:46.735 GMT I ENABLE_OSGIBUNDLE:Thread-24 com.ibm.cics.o
Bundles:
0 org.eclipse.osgi 3.6.1.R36x_v20100806 ACTIVE
1 org.eclipse.osgi.services 3.2.100.v20100503 ACTIVE
2 org.eclipse.equinox.log 1.2.100.v20100503 ACTIVE
3 com.ibm.cics.server 1.300.0 ACTIVE
4 com.ibm.cics.ras 1.300.0 ACTIVE
5 com.ibm.ras 1.300.0 ACTIVE
6 com.ibm.record 1.0.0 ACTIVE
7 com.ibm.cics.domains 1.300.0 ACTIVE
8 com.ibm.cics.osgi 1.0.0 ACTIVE
9 com.ibm.cics.osgi.impl 1.0.0 ACTIVE
10 com.ibm.cics.server.examples.hello 1.0.0 ACTIVE
11 com.ibm.cics.server.examples.jcics 1.0.0 ACTIVE
12 com.ibm.cics.server.examples.web 1.0.0 RESOLVED
```

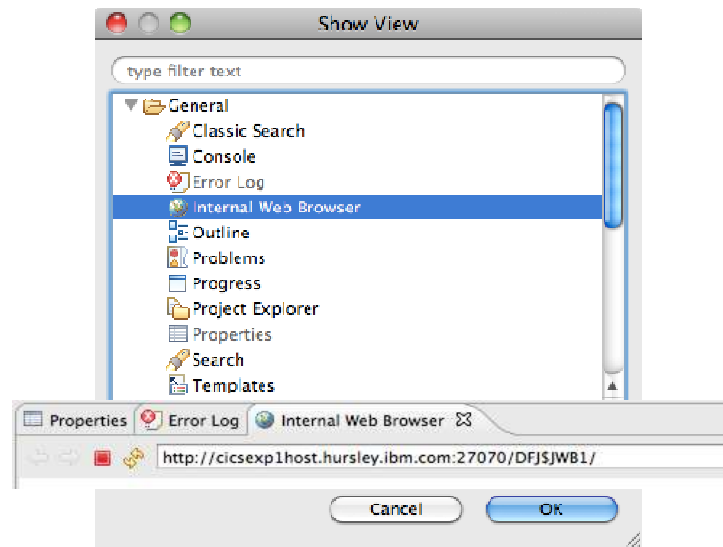
You can check the log for any problems with the OSGi bundles and services. The log now contains the list of OSGi bundles for the application that have been installed by the system programmer.

Open the internal web browser



Open the internal web browser in the CICS Explorer to test the web example.

Enter the URL for the web example



Enter the URL to call the Java program.

Success

The screenshot shows the CICS Explorer interface. On the left, a tree view displays the system structure, including various CICS programs and bundles. The main window is split into two panes. The top pane shows a web browser window displaying the 'Web Sample1' application. The bottom pane displays the 'Inbound Client Request Information' for the current request.

Web Sample1

Inbound Client Request Information:

- Method: GET
- Version: HTTP/1.1
- Path: /DFISJWB1/
- Request Type: IITTPYES
- Query String: null
- HTTP headers:

The right pane shows the details of the HTTP request, including headers such as 'Host', 'User-Agent', 'Accept', 'Accept-Language', 'Accept-Encoding', 'Cookie', 'IBM-OSQO-ACCESS', 'Server', 'Client-Address', 'Client-Address:2', 'Server-Address', 'Client-Host', 'OSI', 'Type-Version', 'Port-Number', 'DeclLen', 'DeclLen:2', and 'Sample1 complete'.



Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send email feedback:

[mailto:iea@us.ibm.com?subject=Feedback about CICSTS42 E2E Java.ppt](mailto:iea@us.ibm.com?subject=Feedback%20about%20CICSTS42%20E2E%20Java.ppt)

This module is also available in PDF format at: [../CICSTS42_E2E_Java.pdf](#)

You can help improve the quality of IBM Education Assistant content by providing feedback.



Trademarks, disclaimer, and copyright information

IBM, the IBM logo, ibm.com, CICS, CICS Explorer, and z/OS 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 other IBM trademarks is available on the web at "[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)" at <http://www.ibm.com/legal/copytrade.shtml>

Java, and all Java-based trademarks and logos are trademarks of Oracle and/or its affiliates.

Other company, product, or service names may be trademarks or service marks of others.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.

© Copyright International Business Machines Corporation 2011. All rights reserved.