HITACHI
Inspire the Next

Hitachi OSGi Series
SuperJ™ Tools User's Guide

80140-CST-000-02
January 2013

Hitachi Communication Technologies America, Inc.

Copyright Notice

©2009-2013 Hitachi Solutions, Ltd., and Hitachi Communication Technologies America, Inc. (Hitachi-CTA), All Rights Reserved. Printed in the United States of America.

This edition (80140-CST-000-02 for product Version 3.3.2, January 2013) replaces previously dated versions of this document. Although this document was believed to be correct at the time of publication, Hitachi Communication Technologies, Ltd., Hitachi Communication Technologies America, Inc., and their affiliates assume no responsibility for errors or omissions. The contents of this manual and the specifications of this product are subject to change without notice. Hitachi-CTA reserves the right to make changes without notice to the specifications and materials contained herein and shall not be responsible for any damages (including consequential) caused by reliance on the materials presented, including, but not limited to typographical and other errors relating to the publication. Technical bulletins may be issued at any time to further clarify product installation, maintenance, use, and administration. In the event of any differences, the information in more recently issued technical bulletin supersedes correlative information in associated product documentation.

For additional information about the products in this document, please contact:

Hitachi Communication Technologies America, Inc.

3617 Parkway Lane
Norcross, GA 30092
Tel (770) 446-8820 Fax (770) 242-1414

Trademarks

Hitachi® and AMN® are registered trademarks of Hitachi, Ltd. SuperJ™, SuperJ Engine®, and SuperJ Framework™ are trademarks or registered trademarks of Hitachi Solutions, Ltd. in Japan.

Certain software products within the SuperJ™ Suite contain Java® software which is owned by Oracle Corporation. The terms and conditions of its use are governed by your agreement with Hitachi. Oracle and Java are registered trademarks of Oracle and/or its affiliates.

All other trademarks are the properties of their respective owners.

SuperJ™ Tools User's Guide

80140-CST-000-02
January 2013

Revision History

The following table identifies the history of revisions to this document, the *SuperJ™ Tools User's Guide* (80140-CST-000-02).

Revision Date	Revision Level	For Product Release	Summary of Changes
December 2009		Version 1.0	HSK First Version
March 2010	00	Version 2.0	Revised for U.S. domestic audience
December 2010	01	Version 3.2	Added new procedures for new features
January 2013	02	Version 3.3.2	Revised for product version 3.3.2. Moved Eclipse installation procedures to Chapter 2. Revised and updated installation procedures and bundle development procedures.

TABLE OF CONTENTS

PREFACE

Please read this Preface to learn important information about using this document and the product(s) that it describes. The Preface contains general information about this document as well as important safety notices.

Contents

Topics covered in this Preface include: **Page**

About this Document

This section provides some general information that you need to know when using this document.

Purpose

This document describes SuperJ Tools (SJT). The document also serves as a configuration and ordering guide.

Intended audience

This document is intended for individuals who will use SJT to develop and deploy bundles.

How this document is organized

The following table summarizes the organization of this document:

Chapter	Description
Preface	Information about this document: its purpose, intended audiences, and organization
1	An introduction to SJT
2	Installation procedures for SJT
3	Procedures and settings for using SJT to develop bundles
4	Procedures and settings for creating deployment packages containing these bundles
5	Procedures for using the CPE Simulator
6	Procedures and settings for creating PAR files to launch deployment packages
Index	An index of the information in this document

Typographical conventions

The following table explains the use of typographical conventions in this document to convey specific information to you:

Type of Information	Convention	Example
Actual data you type in a field	OCR-B 10 BT font	Type `4096`
Generic information you type in a field	Italics	Type *your password*
Substitution variables	Italics inside brackets	Type `attrib` [*filename.hpj*]
Window titles	Italics	The *Setup* window opens.
Menu items, field descriptions, or buttons on a window	The same capitalization and abbreviation that appears on the screen using bold text	Select **Print** on the **File** menu. Click **Apply** in the *Properties* window.

Error or informational message text in a dialog box, LCI message output, or system response	OCR-B 10 BT font	The following message is displayed:
		`Initiating MIB Save`
		
		The system responds:
		`...0016000 SBY`
		`COMMAND END`

Pressing a key on a keyboard	Enclose in angle brackets	Press <Enter>
Pressing a key on a keyboard while holding down another key	List the keys in sequence and separate them by a dash (-)	Press <Ctrl-Alt-Del>
Emphasized words or phrases	Bold italics	Do ***not*** use this feature before you save MIB data.
References to names of books	Title caps, italics	Refer to the *SJDMS System Administrator's Guide.*
References to chapter names or section topics	Same capitalization as the referenced text, in quotation marks	See "Configuration Options" on page 3-2.

Document feedback

At Hitachi, we strive to produce high-quality documentation, and we welcome your feedback. If you have comments or suggestions about our documents, please send us an e-mail. Please include the following information with your feedback:

- Product name and version number
- Document name
- Document date
- Page number
- Brief description of the content, indicating whether you noticed step-by-step instructions that are inaccurate, information that requires clarification or more details, etc.
- Your suggestion(s) for correcting or improving the document

Please send e-mail messages to documentation_mgr@hitachi-cta.com. This e-mail address is only for documentation feedback. If you have a technical question, please contact Technical Support as described in "Technical support" on page xi.

Related documents

In addition to this guide, please refer to the following documents for more information:

Document	Version	Source
OSGi Service Platform Core Specification	Release 4, Version 4.2 June 2009	The OSGi Alliance
OSGi Service Platform Service Compendium	Release 4, Version 4.2 August 2009	The OSGi Alliance
OSGi Service Platform Mobile Specification	Release 4, Version 4.0 July 2006	The OSGi Alliance
J2EE™ Client Provisioning Specification	Version 1.0 (JSR124)	Sun Microsystems, Inc.

A Word about Notices

Our notices delineate possible conditions that could affect operability of software for all of our product lines. This section describes how different types of safety notices are presented in this document.

⚠IMPORTANT

Comply with all danger, warning, and caution statements in this document.

Hazardous conditions

This document includes safety notices that emphasize the potential for these hazardous conditions:

- Warning
- Caution

Warning message

Warning messages indicate that you are in the presence of a risk that can cause catastrophic software failure if the risk is not avoided, which includes:

- Unrecoverable software errors that could cause irretrievable data or complete corruption of data
- Server inoperability
- Necessitating Hitachi-CTA intervention to restore operation

This document uses the following format to indicate **Warning** messages:

⚠WARNING!

Text describing the risk condition.

Caution message

Caution messages indicate the presence of a risk that may cause losses if the risk is not avoided. Property damage includes:

- Software malfunction
- Loss of software or data
- Short-term service interruption

This document uses the following format to indicate **Caution** messages:

> ⚠ CAUTION!
>
> Text describing the risk condition.

Important message

Important messages indicate information that is important to the successful and/or efficient operation of the product, including:

- Settings for the hardware or software to avoid conflict with other devices
- Configuration information to ensure maximum efficiency
- Reminders to verify that vital prerequisite processes have been completed

This document uses the following format to indicate **Important** messages:

> ⚠ IMPORTANT!
>
> Text describing necessary information.

IMPORTANT NOTICES

This section contains important information about successfully and efficiently operating the SJDMS Admin Console and SJT.

⚠ IMPORTANT!

Please read this entire document before you use this product. Successful and efficient use of your SJDMS Admin Console and SJT depends upon careful reading of this document. Be sure to carefully follow all instructions that accompany this product. Pay particular attention to notice statements, and keep this document for future reference.

Technical support

After following the procedures in this document, if the product is still not functioning properly, contact the National Technical Assistance Center (NTAC) at 1-800-944-7185, or via email at ntac@hitachi-cta.com.

1 PRODUCT OVERVIEW

This chapter provides an overview of SuperJ Tools (SJT).

Contents

Topics covered in this document include: **Page**

Note: The first topic begins on the following page.

Introducing SuperJ Tools

SuperJ Tools (SJT) is a tool for developing Java applications for the embedded devices equipped with SuperJ Framework (SJF). With this kit, you can load custom OSGi-compliant Java applications on an embedded device, and run these applications within the Java environment.

SJF combined with this SJT makes it possible for you to remotely control embedded devices including cell phones, car navigation systems and a variety of consumer electronics products. SJT runs in the Eclipse IDE, where these bundles are considered to be plug-ins. Therefore, developing the OSGi bundles implies the use of the Eclipse Plug-in Development Environment (PDE).

Product structure

Bundles developed with SJT can be deployed as services on CPE devices running SJF. This is done by packaging the bundles as package archive (PAR) files, and then using SuperJ™ Deployment Management System (SJDMS) to deploy the PAR files for the services. SuperJ™ Management Agent (SJMA) performs the transactions between the services and the SJDMS.

SuperJ Framework functionality

The SJF provides the following functions for bundles.

- Bundle lifecycle management
- Bundle state management to obtain information from bundles
- Independent running of bundles, even bundles with shared class names.
- Resolution of bundle dependencies
- Service registration function to register functions that can be provided to other bundles.
- Service acquiring function to access functions that other bundles provide

Operating environment

SJF is an OSGi framework that provides a Java runtime environment compliant with OSGi R4.2 specification.

Required software

To successfully use SJT, you must have the following software installed:

IMPORTANT!

Use specified versions when installing SJT. Performance with later versions may vary, producing unexpected results.

Eclipse Indigo option

Software	Version	Installation
Java SE Development Kit (JDK)	1.6 or later	Execute and follow the installation wizard prompts
Eclipse IDE for Java Developers	3.7.2 (Indigo)	See "Installing SJT in Eclipse" on page 2-7 for details.
Eclipse Plug-in Development Environment (PDE)	3.7.2	
Apache Commons Net (Orbit bundle version)[a]	2.0.0 only	Download from the linked list within the latest Recommended Build of Orbit at the Eclipse download site for Orbit, and place the file (such as org.apache.commons.net_2.0.0.v201101241702.jar) in ..\<*EclipseInstallationPath*>\dropins.

a. Only required for using the FTP upload feature to interface with an SJDMS Server.

Eclipse Helios option

Software	Version	Installation
Java SE Development Kit (JDK)	1.6 or later	Execute and follow the installation wizard prompts
Eclipse IDE for Java Developers	3.6.2 (Helios)	See "Installing SJT in Eclipse" on page 2-7 for details.
Eclipse Plug-in Development Environment (PDE)	3.6.2	
Apache Commons Net (Orbit bundle version)[a]	2.0.0 only	Download from the linked list within the latest Recommended Build of Orbit at the Eclipse download site for Orbit, and place the file (such as org.apache.commons.net_2.0.0.v201101241702.jar) in ..\<*EclipseInstallationPath*>\dropins.

a. Only required for using the FTP upload feature to interface with an SJDMS Server.

2 INSTALLING SUPERJ TOOLS

This chapter describes how to install the components of SuperJ Tools (SJT).

Contents

Topics covered in this document include: **Page**

Note: The first topic begins on the following page.

Installing and Configuring Eclipse

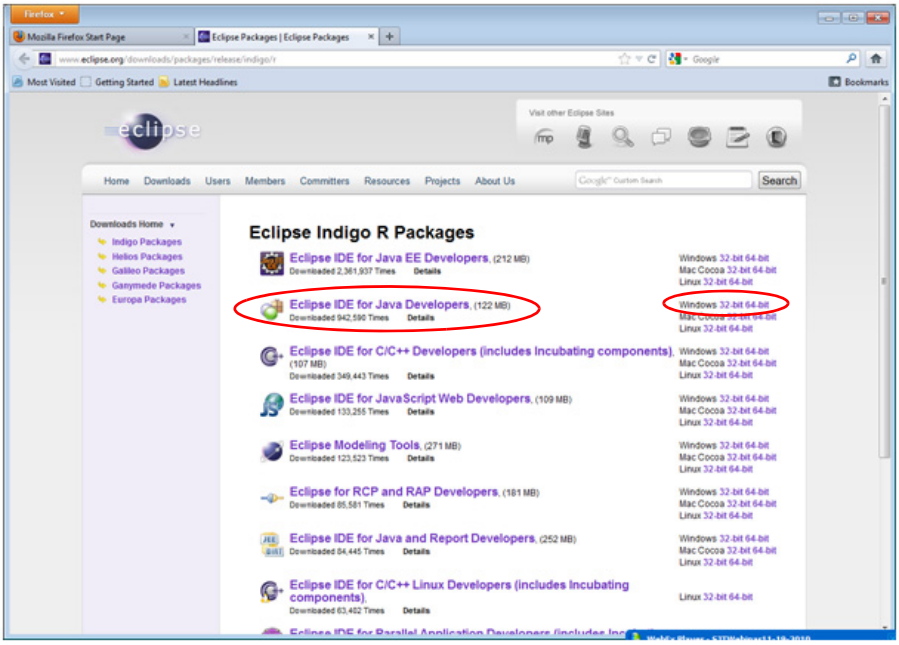
This section provides the procedure to prepare the Eclipse environment.

Installing the Eclipse IDE for Java Developers

Install Eclipse as follows:

1. Access the Eclipse download site (http://www.eclipse.org/downloads/ or the http://archive.eclipse.org/eclipse/downloads).
2. Go to the packages list for the appropriate Eclipse version (see "Required software" on page 1-3 for more information). This procedure assumes that you are downloading Eclipse Indigo.
3. Download the **Eclipse IDE for Java Developers** by selecting the link for your operating system.

   This example shows the download site page for Eclipse Indigo R, noting the Windows 32-bit option for the Eclipse IDE for Java Developers.



4. On the Eclipse downloads page, click the download link.
5. When the download is complete, unzip the downloaded ZIP file to the desired directory. In this document, the root directory (C:) is used.
6. Start Eclipse by navigating to the Eclipse directory and double-clicking on the eclipse.exe file (or selecting the Eclipse start icon from your start menu in Windows).
7. Select a workspace as prompted.
8. Close the Welcome screen.

Configuring Eclipse network proxy connection settings

After you have installed Eclipse, you need to configure Eclipse to make network connections for software updates and downloads.

If your network environment connects with the Internet through a proxy server, configure Eclipse as follows:

1. Modify your eclipse.ini file as follows:
   a. Use a text editor to open eclipse.ini.
   b. Set the size of the permanent area memory for the Java VM designated when booting Eclipse to greater than 128MB. For example, add the following lines to eclipse.ini.
   ```
   -Xms40m
   -Xmx512m
   --launcher.XXMaxPermSize
   256m
   ```
   c. Add the following line to the eclipse.ini:
   ```
   -Dorg.eclipse.ecf.provider.filetransfer.excludeContributors=org.eclipse.ecf.provider.filetransfer.httpclient
   ```
   Note: This command says that Eclipse can access the web via the *.pac files of the HTTP clients (i.e., your internet browser).
   d. Save and exit eclipse.ini.
2. Ensure that the settings for your default system HTTP client (browser) are properly configured.
3. From the main menu of Eclipse, choose **Window > Preferences**. The *Preferences* window opens.
4. From the preferences tree view, choose **General > Network Connections**. The *Network Connection* panel displays.

continued...▶

Configuring Eclipse network proxy connection settings, continued

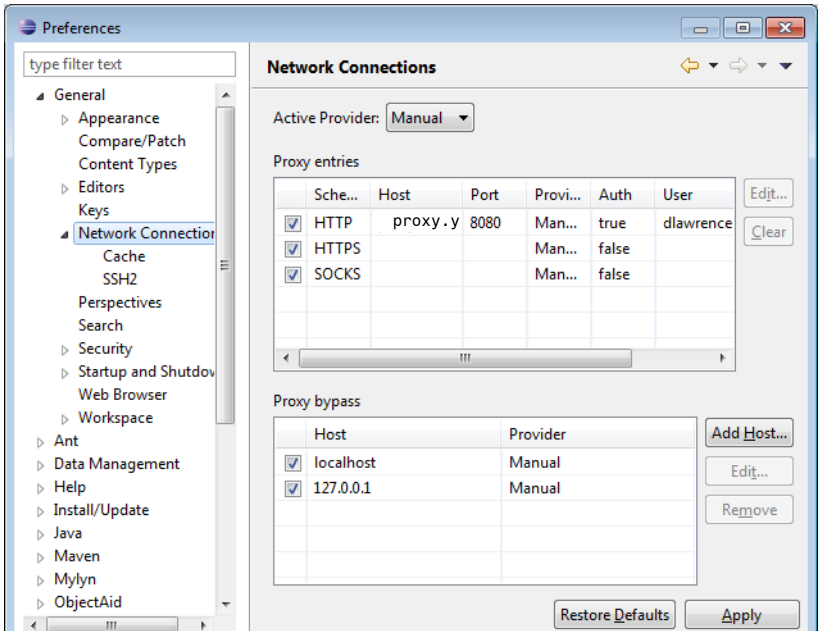
5. If the current proxy settings in the *Network Connections* panel are not correct, make the proxy settings within the Eclipse interface as follows:

   a. From the **Active Provider** list, choose **Manual**. This selects the HTTP, HTTPS, and SOCKS entries in the Proxy entries list.

   b. Select the **HTTP** entry in the **Proxy entries** list, and then click the **Edit** button next to that list. Choose the network settings suitable to your network environment.

   c. In the Edit Proxy Entry dialog box,
   - enter the proxy server address in the **Host** field, and enter the port number in the **Port** field.
   - Select **Requires Authentication,** and enter your network **User** and **Password**.
   - click **OK**.

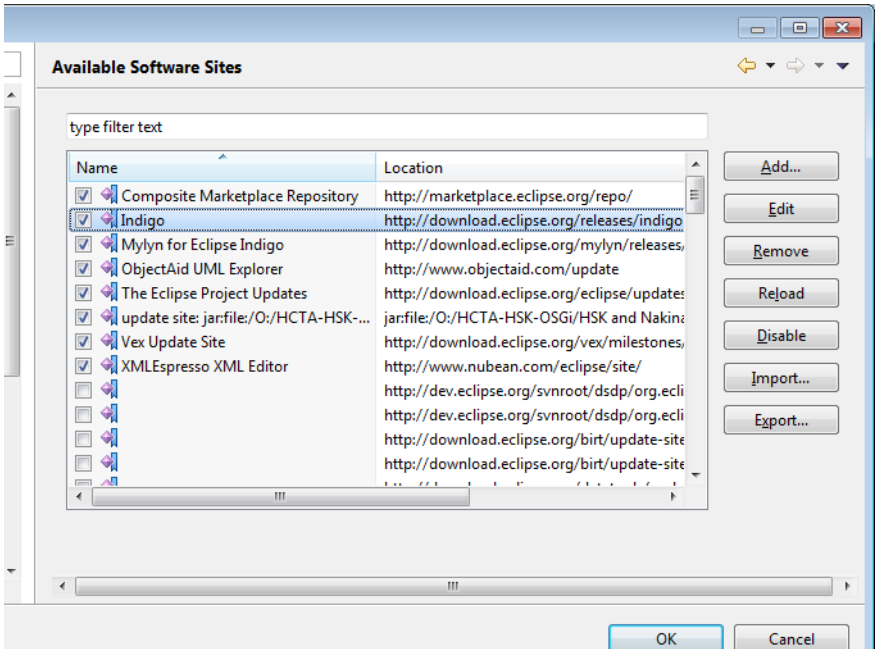
   d. Click **Apply**, and then click **OK**. The Proxy entries list on the **Network Connections** panel shows the proxy settings you made.



Verifying Eclipse communications

Perform the following procedure to ensure that Eclipse can communicate with the Eclipse software download site(s):

1. From the main menu of Eclipse, choose **Window > Preferences**. The *Preferences* window opens.
2. From the category tree view, choose **Install/Update > Available Software Sites**. The *Available Software Sites* panel opens.
3. Select **Indigo** (or **Helios** if you installed Helios) from the **Name** column, and then click **Reload**. Eclipse checks the connection to the selected site.



Note: This step may take several minutes to complete, but do not cancel.

- If the test completes successfully,:

  - Click **OK.**
  - Go to the next procedure.

- If this test is unsuccessful, go to “Configuring Eclipse network proxy connection settings” on page 2-3, and then return to this procedure.
- If you have already completed “Configuring Eclipse network proxy connection settings” on page 2-3 and still cannot successfully complete this procedure, go to “Troubleshooting communication problems” on page 2-6.

Troubleshooting communication problems

If the proxy settings in Eclipse are correct for your network environment but problems persist with Eclipse connecting to an Available Software Site when you test the connection, do the following:

1. Return to the *Network Connections* panel of the *Preferences* window.
2. Clear the settings for HTTPS and SOCKS connections.
3. Click **Apply**.
4. Perform "Verifying Eclipse communications" on page 2-5.
5. If you still cannot successfully verify Eclipse communications, verify that port 80 is available for HTTP traffic. If you need to change the port used for HTTP in Eclipse, perform "Changing the HTTP service port for the targeted platform" on page 3-20.
6. If you still cannot successfully verify Eclipse communications, open your default browser and set it to automatically detect LAN settings. For Internet Explorer, for example, select **Tools > Internet Options**. On the **Connections** tab click **LAN Settings**, and then verify that the only selection is **Automatically detect settings.** Click **OK** twice, and then close the browser.
7. If you still cannot successfully verify Eclipse communications after performing this procedure, contact your System Administrator.

Installing SJT in Eclipse

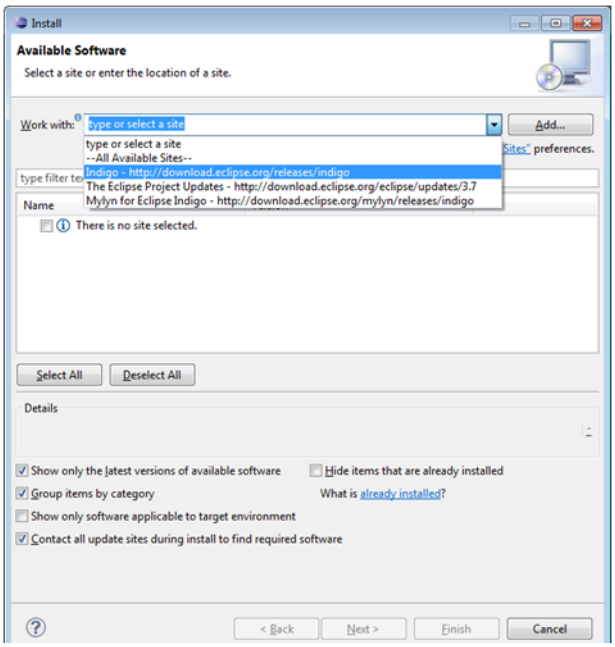
This section guides you through the process of installing the required version of the Eclipse PDE and then installing SJT.

Installing the Eclipse Plug-in Development Environment

To install or update the Eclipse Plug-in Development Environment for SJT, follow these instructions:

Note: If your network uses a proxy server, before proceeding, be sure you have completed the steps described in "Configuring Eclipse network proxy connection settings" on page 2-3.

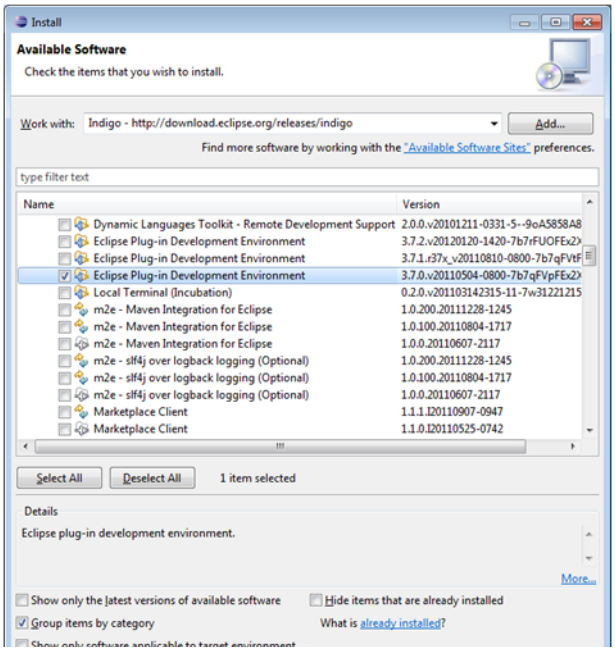
1. Start Eclipse if it is not already running.
   a. If the Workspace Launcher opens, select the desired workspace.
   b. The workbench window opens.
2. On the main menu, choose **Help > Install New Software**. The *Install* window opens.
3. In the Work with list, select the site for your Eclipse version. For example for Eclipse Indigo, select Indigo - http://download.eclipse.org/releases/indigo



continued...

Installing the Eclipse Plug-in Development Environment, continued

4. Clear the **Show only the latest versions of available software** check box
5. In the software list treeview, expand the **General Purpose Tools** node.
6. Scroll down to and select the checkbox for **Eclipse Plug-in Development Environment 3.7.0**.

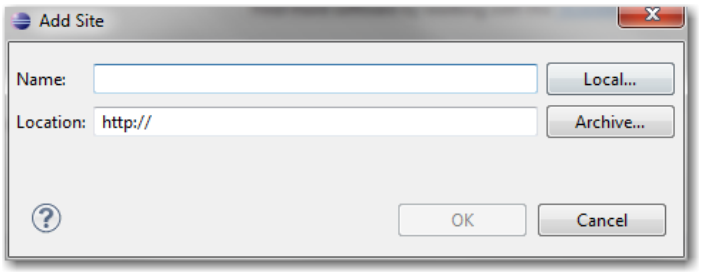


7. Click **Next**. Dependencies are resolved and the Install Details window appears.
8. Click **Next**.
9. In the Review Licenses window, review the license agreement, select **I accept the terms of the license agreement**, and then click the **Finish** button. The software is installed.
10. At the prompt, click **Restart Now** to restart Eclipse.

Installing SJT in Eclipse

After installing the Eclipse PDE, install SJT. Use this procedure to install or update SJT:

1. Choose **Help > Install New Software** from the main menu. The *Install* window opens.
2. Click **Add** to open the *Add Site* window.

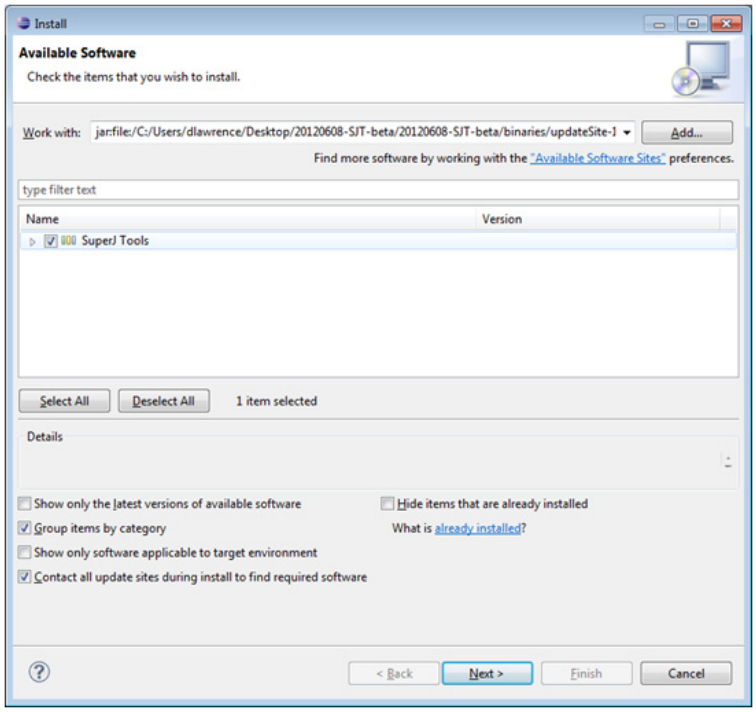


3. Click the **Local** button.
4. On your local or network file system, navigate to and select the SJT upload ZIP file, and then click **Open**.
5. In the *Add Repository* dialog box, click **OK**.

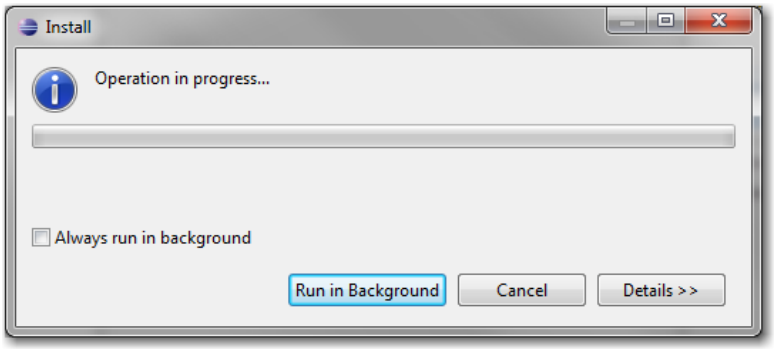
   In the Install window, the **SuperJ Tools** software entry appears.

Installing SJT in Eclipse, continued

6. In the **Name** column, select the check box beside **Super J Tools**, and then click **Next**. The *Install* window is updated showing SuperJ Tools in the list of items to be installed.
7. Select the checkbox for **SuperJ Tools**.



8. Click **Next**.
9. Click **Next**.
10. In the *Review Licenses* window, review the license agreement, and then click **I accept the terms of the license agreement**, and then click **Finish**. Installation begins.

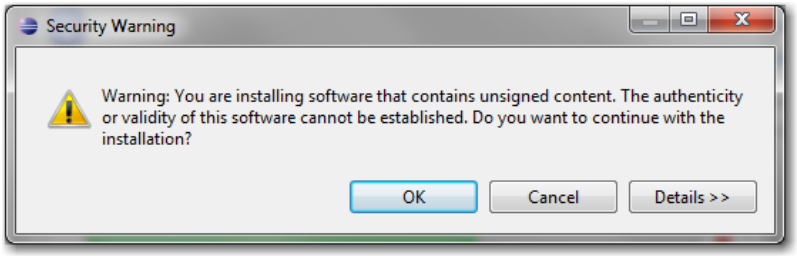


Note: If you want the installation to run in the background, click **Always run in background**.

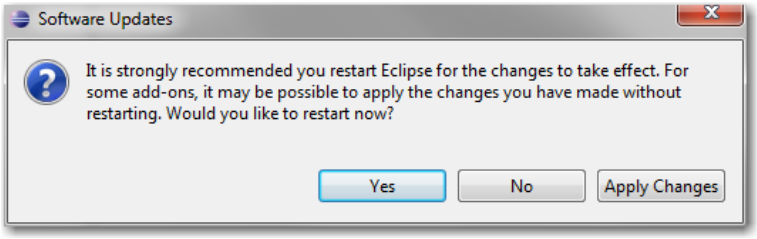
continued...

Installing SJT in Eclipse, continued

11. If a *Security Warning* window opens, click **OK**.



12. When installation completes, click **Yes** to restart Eclipse.

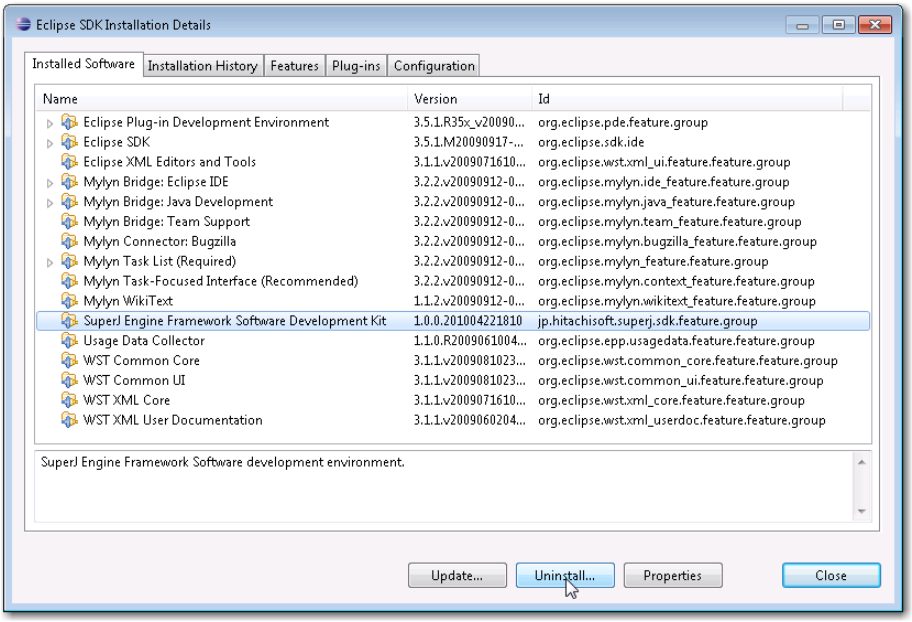


13. If you are updating SJT, the target platform was already set to SuperJ Framework, so perform “Reloading the SuperJ Framework target platform” on page 3-17.

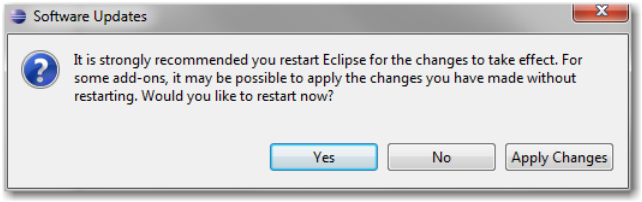
Uninstalling SJT

SJT can be uninstalled with the same procedure for uninstalling a generic Eclipse Plug-in. Details are as follows:

1. Choose **Help > About Eclipse** from the main menu to open the *About Eclipse SDK* window.
2. Click **Installation Details** to open the *Eclipse Installation Details* window.
3. On the **Installed Software** tab, select the previous installation of the SuperJ product from the **Name** list, and then click **Uninstall**.



4. Follow the instructions in the uninstall wizard.
5. If a *Security Warning* displays, click **OK**.
6. Restart Eclipse by clicking **Yes** in the *Software Updates* window.



7. If you are installing an updated SJT, go to Step 13 of "Installing SJT in Eclipse" on page 2-9, and perform the remainder of that procedure.

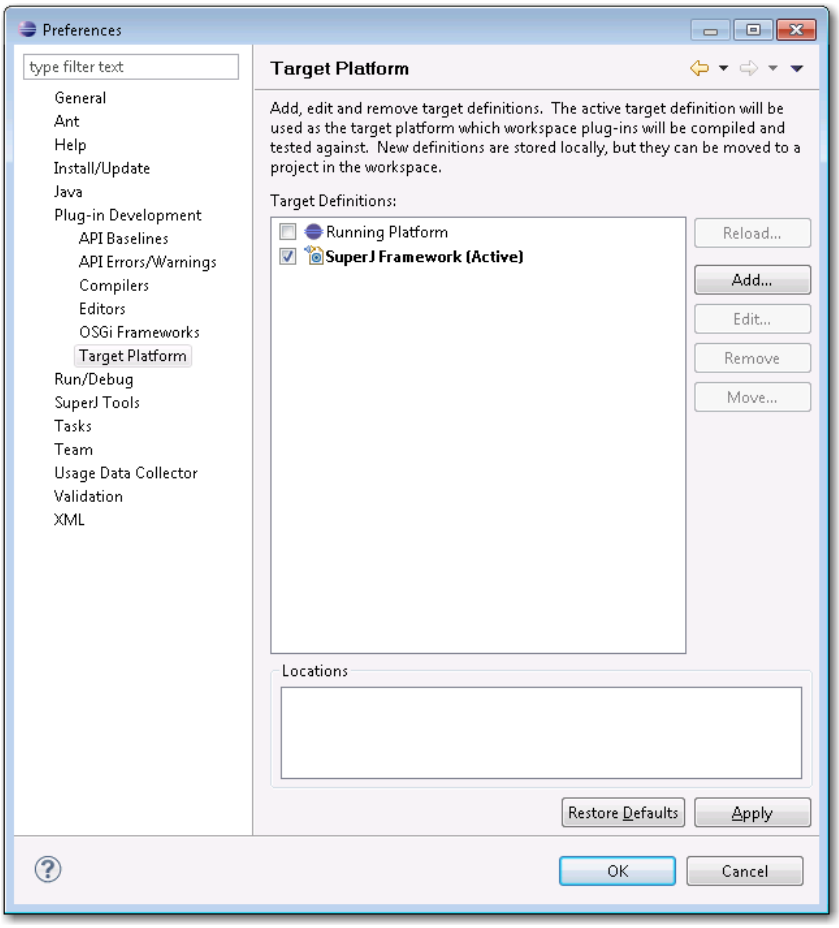
Troubleshooting Upgrade Issues

This section provides information for troubleshooting the installation of SJT.

Resetting SJT after upgrade

If you updated SJT and then received error messages about missing target definitions, libraries, or filename properties, perform the following procedure:

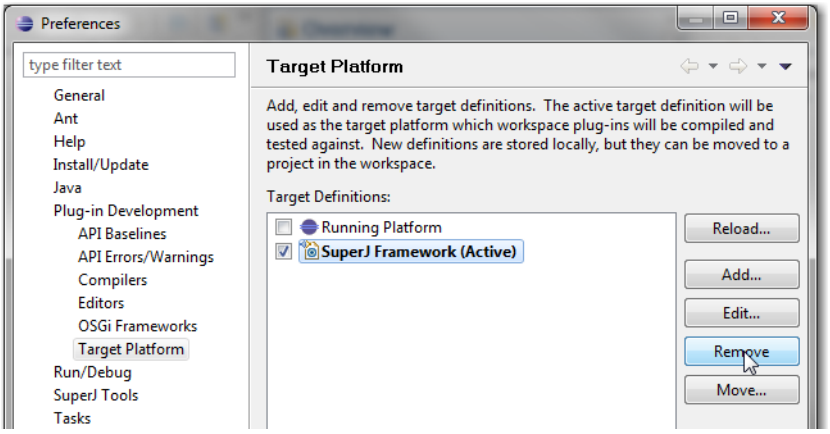
1. Choose **Window > Preferences**, and then in the *Preferences* window select **Plug-in Development > Target Platform**.



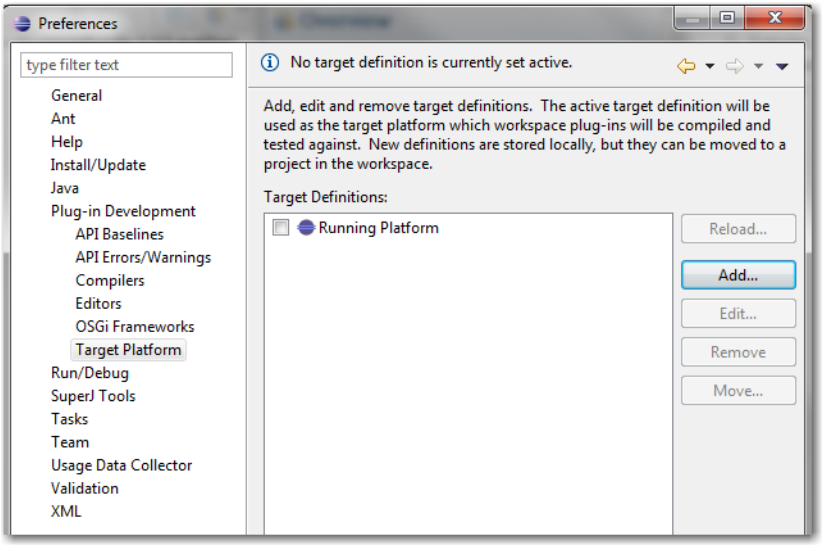
continued...

Resetting SJT after upgrade, continued

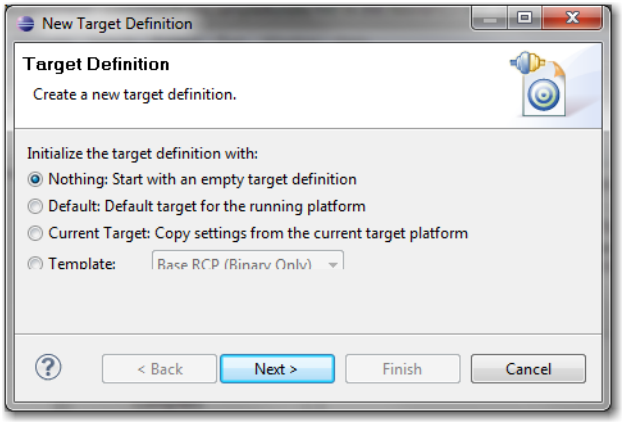
2. If SuperJ Framework is in the list select it, and then click **Remove**.



Otherwise, go to the next step.



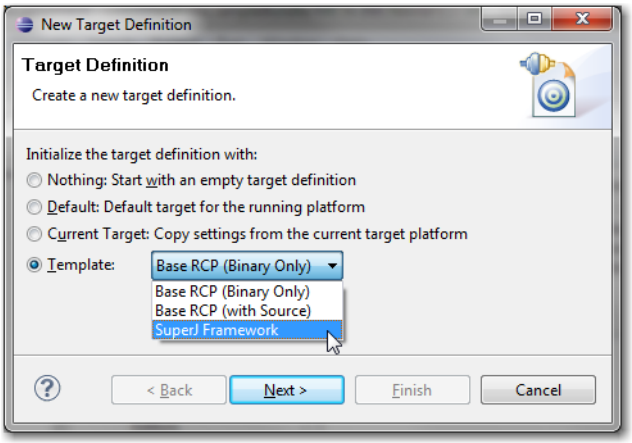
3. Click **Add** to open *New Target Definition*.



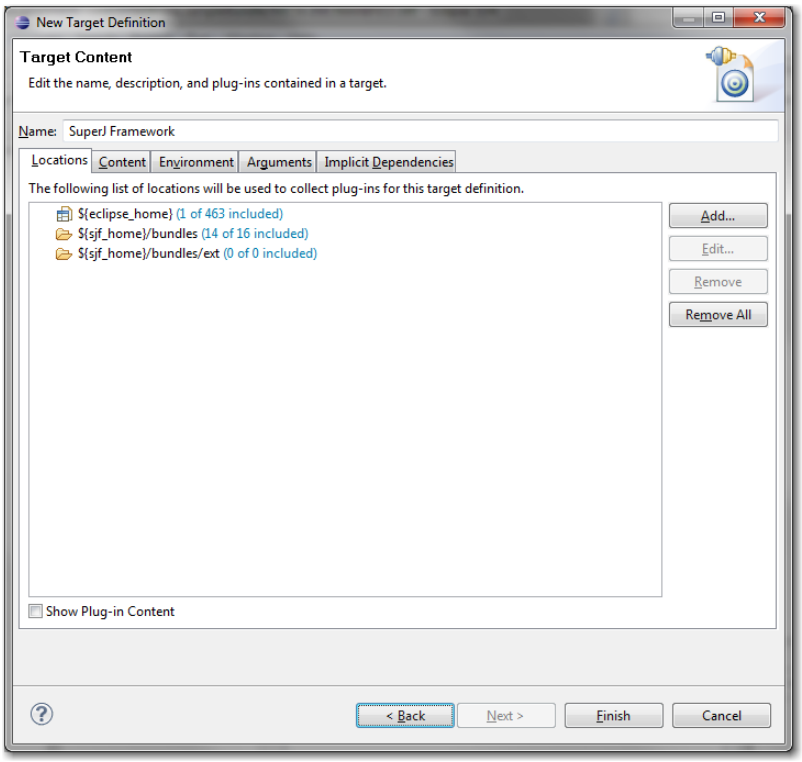
continued...

Resetting SJT after upgrade, continued

4. Choose the **Template** option, and then select **SuperJ Framework** from the list.



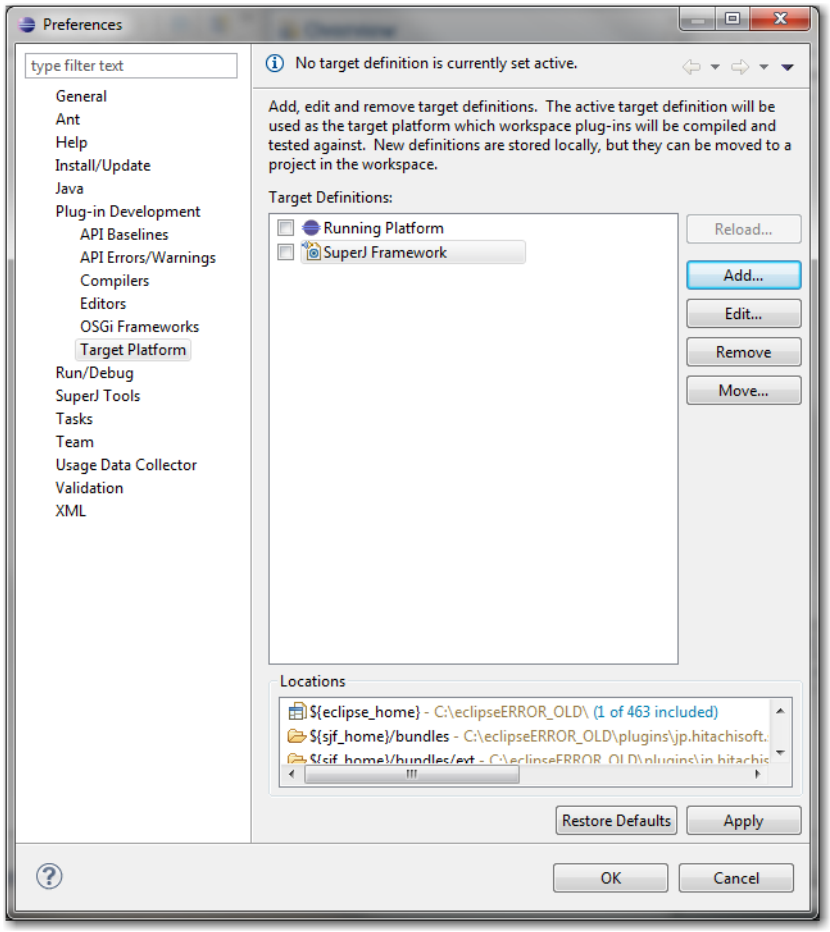
5. Click **Next.**



continued...

Resetting SJT after upgrade, continued

6. Click **Finish.** The *Preferences* window shows the **SuperJ Framework** option in the list of **Target Definitions**.

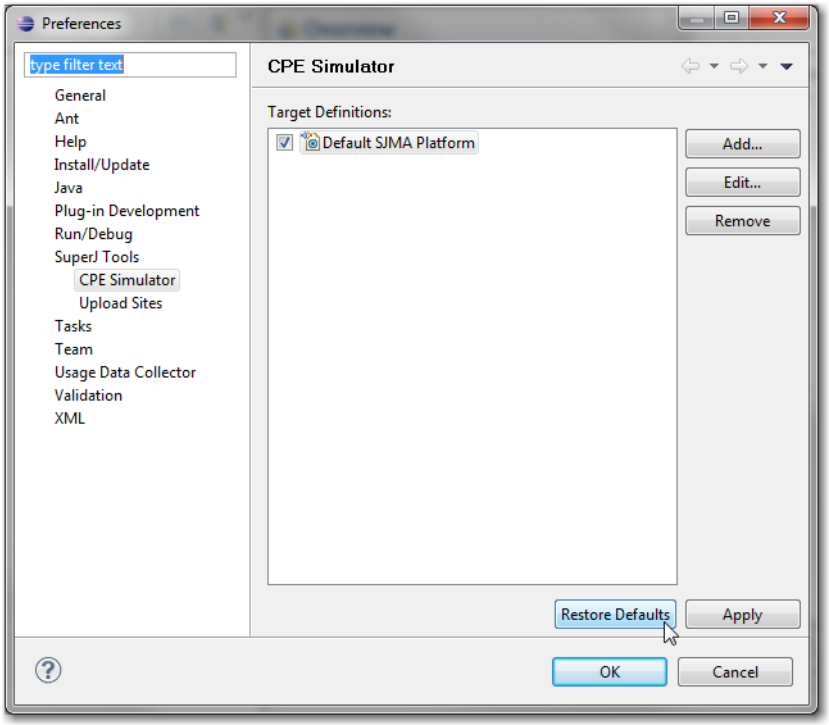


7. Select **SuperJ Framework**, and then click **Apply**.

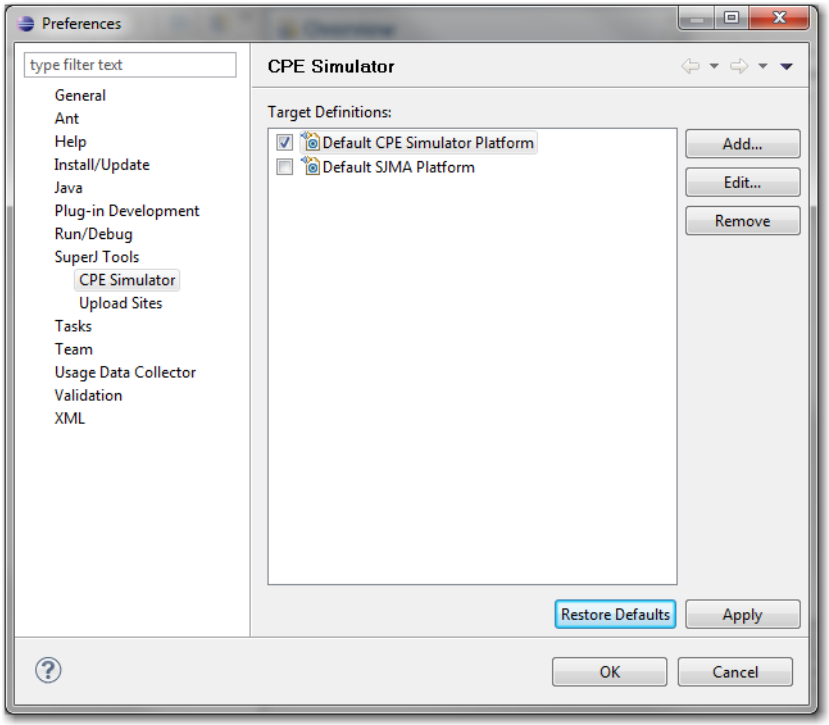
continued...

Resetting SJT after upgrade, continued

8. Select **SuperJ Tools > CPE Simulator** in from the Preferences window.



9. Click **Restore Defaults**.

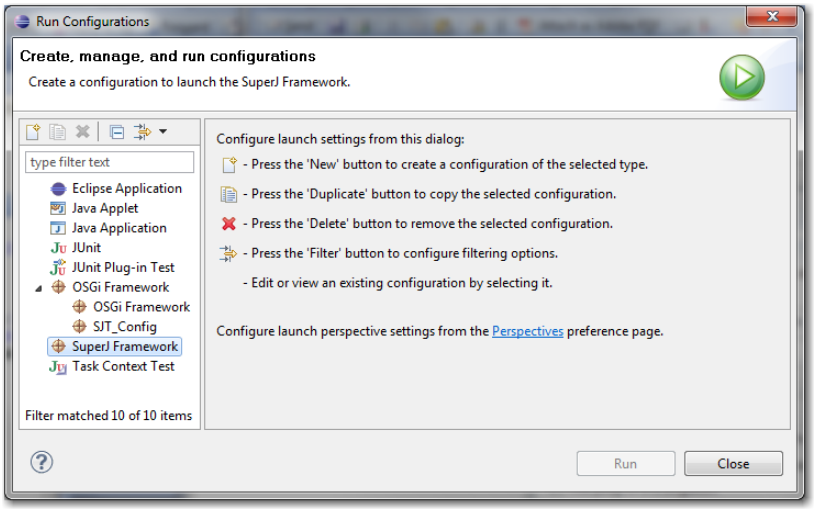


10. Click **OK** to close the *Preferences* window.

continued...▶

Resetting SJT after upgrade, continued

11. Choose **Run >Run Configurations** to open the *Run Configurations* window.



12. Delete all the configurations listed under **SuperJ Framework** in the tree panel on the left, and then click **Close**.
13. If you still receive an error message, perform this procedure a second time.
14. If you have performed this procedure twice and still receive an error message, call the Hitachi National Technical Assistance Center (NTAC) at 1-800-944-7185.

3 DEVELOPING A BUNDLE

This chapter describes how to develop an application bundle for the SJT.

Contents

Topics covered in this document include: **Page**

Note: The first topic begins on the following page.

Using SJT to Develop New Bundles

You can create OSGi bundles that run on the SJF using the Eclipse Plug-in Project. This section discusses how to make a new bundle project in the SJT and how to export it so that it is available to deployment packages.

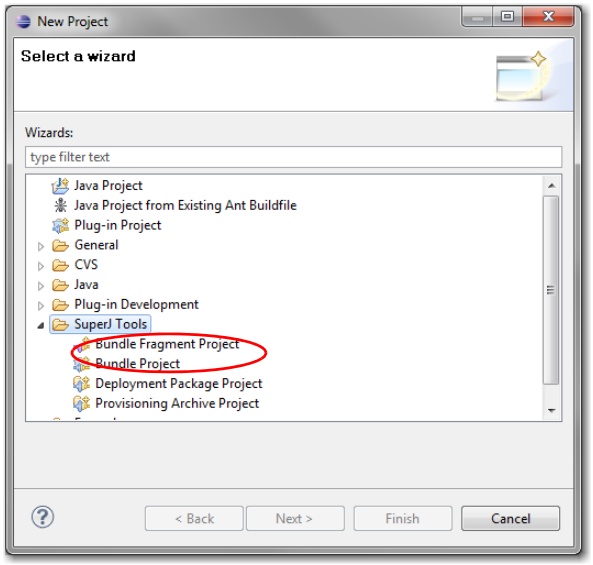
Example bundles are available from Hitachi; contact your Hitachi representative for more information.

Creating a Bundle Development Project

The procedure to create a bundle (plug-in) project is as follows:

1. Select **File > New > Project** from the main menu to open the *New Project* window.
2. Under **Super J Tools,** select either **Bundle Project** or **Bundle Fragment Project** as desired, and then click **Next**.

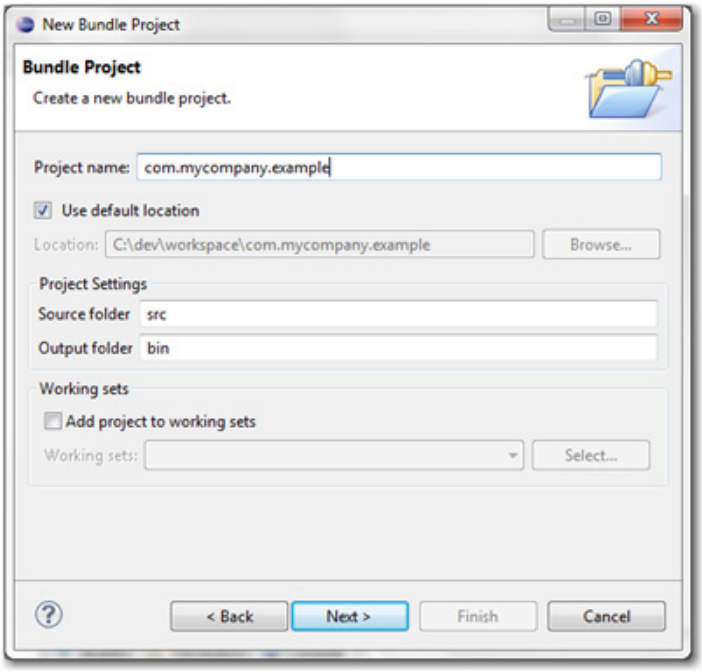
   The remainder of this procedure describes the creation of a bundle project, but the same steps apply if you are developing a bundle fragment.



continued...

Creating a Bundle Development Project, continued

3. In the *New Bundle Project* dialog box, do the following:
   a. Type a unique name based in the **Project name** field following reverse domain naming convention. This populates the bundle's Symbolic Name.
   b. Make any desired changes in the **Project Settings** group.
   c. If you want the bundle to be assigned only to a working set, select this option and choose from the list of working sets.

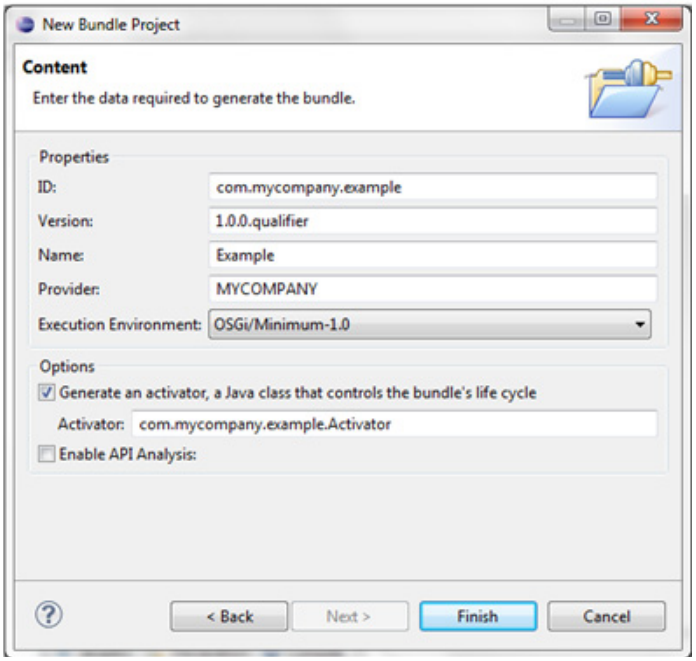


4. Click **Next**.

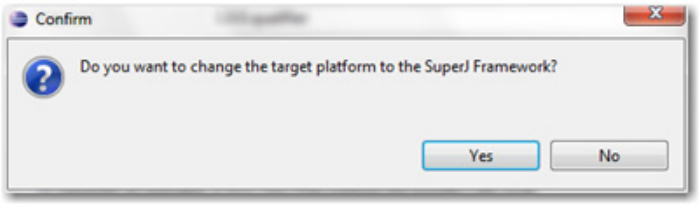
continued...

Creating a Bundle Development Project, continued

5. Review the fields in the **Properties** area of the *New Bundle Project: Content* window to ensure that they are appropriate, and make changes if necessary:
   a. **ID** field: populated by the **Project name** field in the preceding step.
   b. **Version** field: the contents of this field are appended to the contents of the ID field to create a unique Resource ID to identify the bundle in the framework.
   c. **Name** field: populated from the project name entered previously.
   d. **Provider** field: optional field to identify the developer of the bundle
   e. The **Execution Environment** is the minimum set of Java APIs required for the bundle to run, subject of the OSGi Specification. Set by default to OSGi/Minimum 1.0.
6. Click **Finish**.



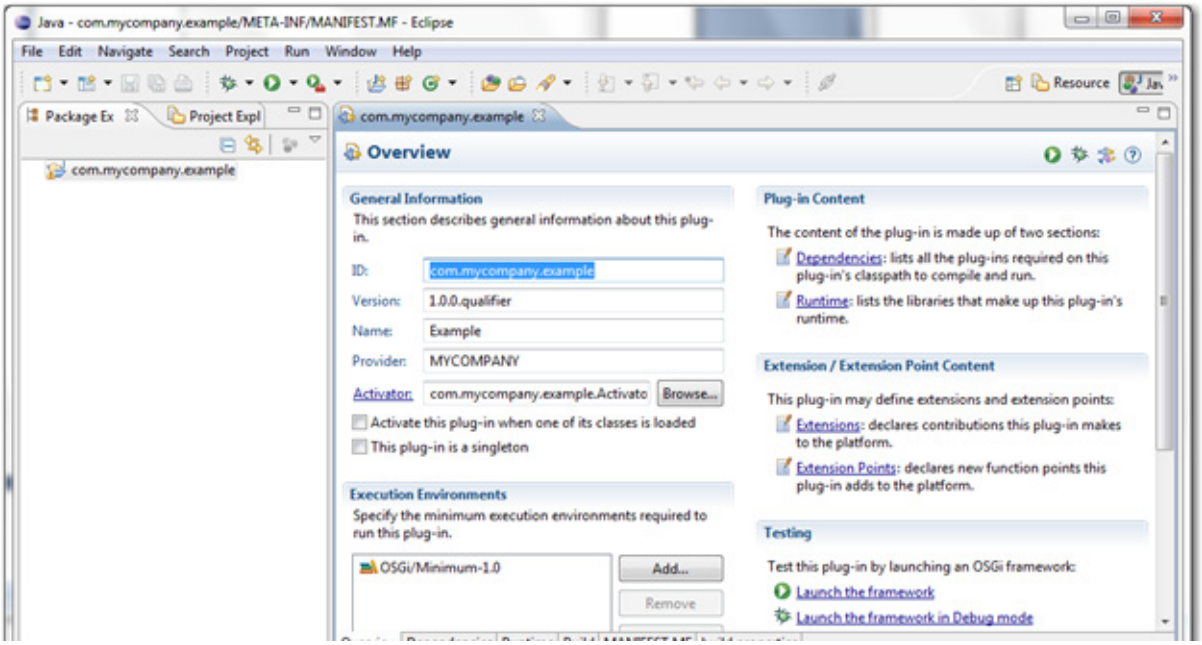
7. If you are prompted to change the target platform to the SuperJ Framework, click **Yes**.



continued...

Creating a Bundle Development Project, continued

The bundle window opens. This is the window from which to start the next procedure.



The bundle window contains the following tabs:

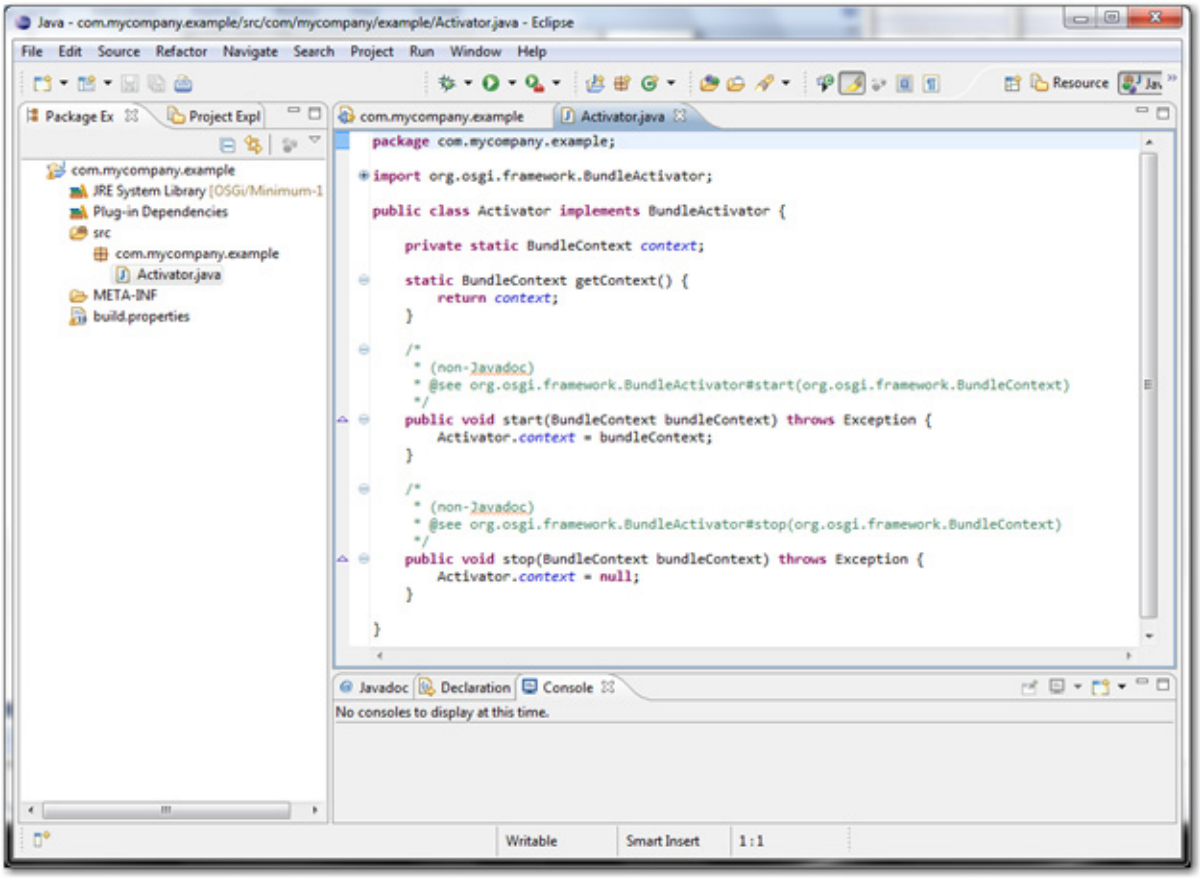
- Overview
- Dependencies
- Runtime
- Build
- MANIFEST.MF
- build.properties

Note: Both the Extensions tab and the Extension Points tabs are only used to develop an Eclipse Plug-in and should not be used to develop the an OSGi bundle.

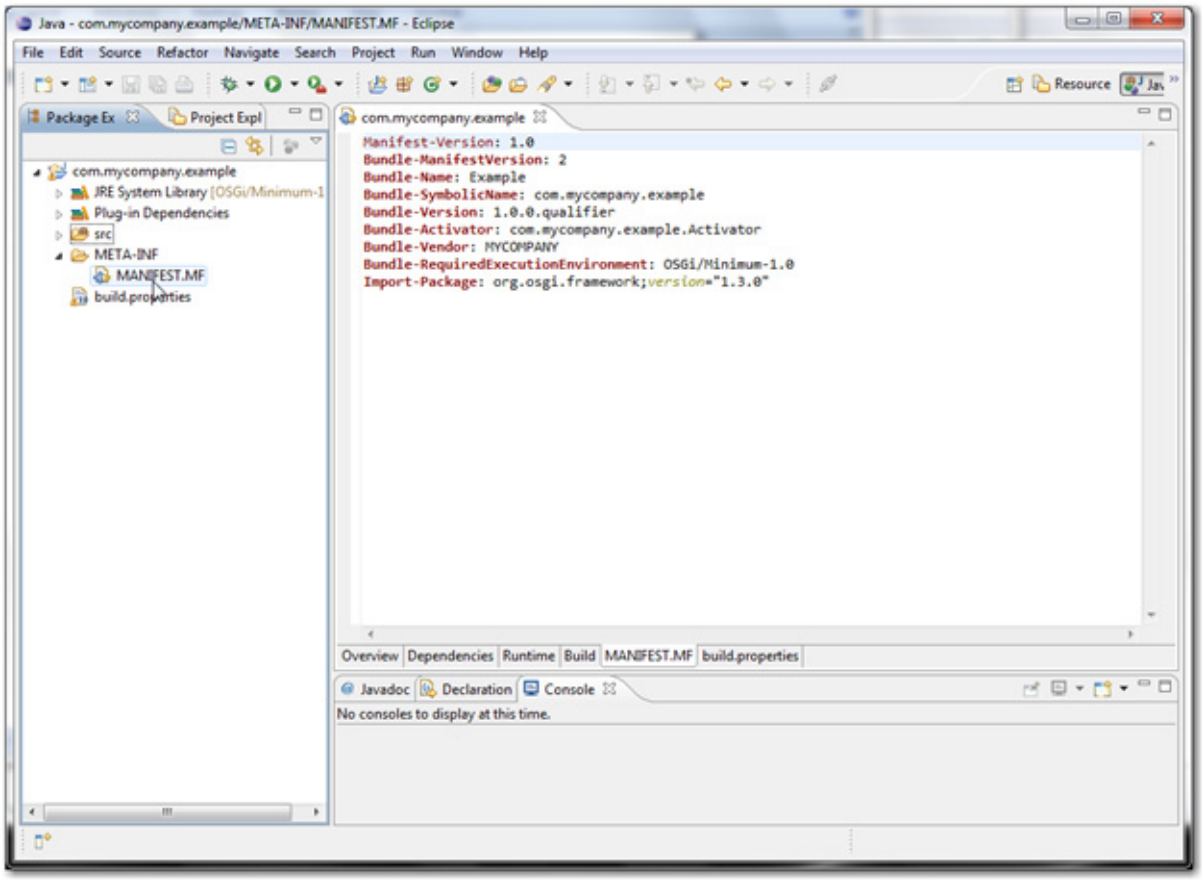
For more information on the tabs in the bundle window, see Eclipse Help.

8. To view or edit the MANIFEST.MF file, you can click the MANIFEST.MF tab. You can also access the manifest through the Package Explorer.

9. To access the source Java within the bundle, you can use the Package Explorer to expand the `src` node and the project package node. You will then see the source `*.java` files for the bundle. Double-click any of these to open the file in the editor.



10. You can also access the bundle manifest under the META-INF node in Package Explorer.



Exporting a bundle project

When you are ready to export the bundle project, follow these steps:

1. In the Package Explorer, right-click on the project, and then select **Export** on the context menu. The *Export* window opens with Select page.
2. In the destinations treeview, expand the **SuperJ Tools** node and select **Deployable bundles**.
3. Click **Next**. The *Export* dialog, *Deployable plug-ins and fragments* page appears.
4. Under **Available plug-ins and fragments**, select the check box(es) for the bundle(s) you want to export.
5. If you are using working sets, click **Working Sets** and select the appropriate working set.
6. Complete the **Destination** tab by choosing to the export target as either:
   - The destination **Directory** where you want the bundle to be exported,
   - The location for the bundle to be exported to an **Archive file** (ZIP), or
   - The location of the desired repository by choosing **Install into host. Repository**,
7. Choose a path for the selected type of JAR file for this bundle by choosing a **Directory** from the list, or by clicking **Browse** to locate the desired folder.

   Note: The directory you specify must contain the site.xml file. For example, if the directory is update, it structure would be as follows:

   ```
   update/
           features/
           plugins/
           site.xml
   ```

8. Make any desired changes on the **Options** tab.
   - If you want to export source code with an executable bundle, select the **Export source > Include source in exported plug-ins** option.
9. If you want to export this bundle with JAR signing, and you have all the necessary information for a keystore, complete the JAR Signing tab.

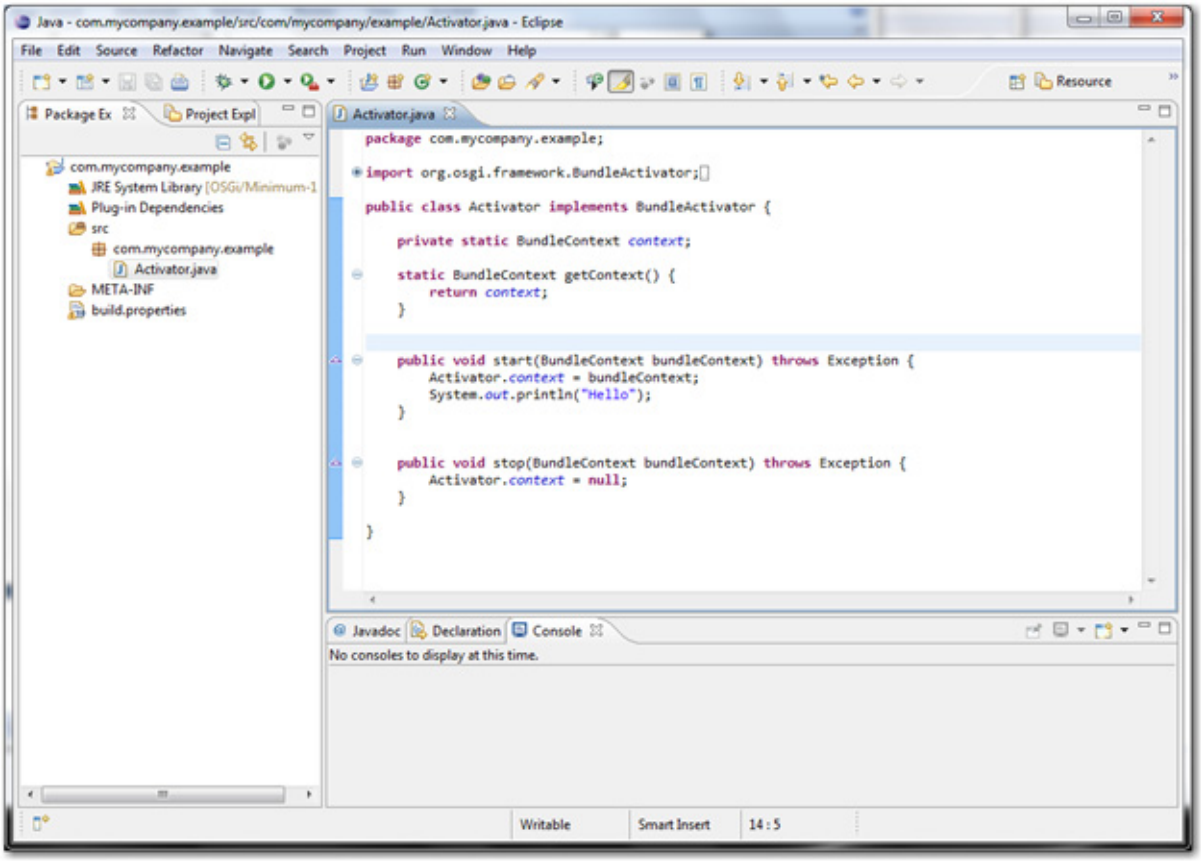
   Note: Do not use JAR Signing unless you have complete information for authentication of the digital signature.

10. Click **Finish**. The *Export Plug-in* window shows the progress of the export operation. If desired, click **Run in Background**.

Using the SuperJ Framework Simulator

To check the operation of the OSGi bundles in your bundle development project, you can use the SuperJ Framework Simulator as described in this procedure as part of testing to debug and/or run configurations of bundles.

1. In an OSGi bundle project, such as the one created in the preceding procedure, use the **Package Explorer** tab to navigate to and double-click on Activator.java to edit it.
2. Insert a simple command line, such as a System.out.println().

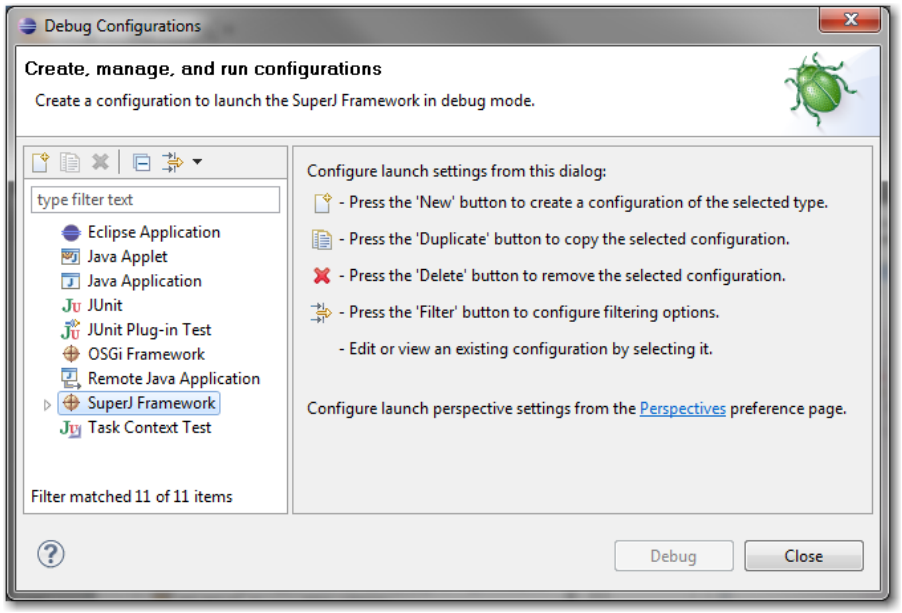


3. Click **Save**.
4. If there are any other processes running in Eclipse, you must terminate them before the SuperJ Framework Simulator can run.

continued...▶

Using the SuperJ Framework Simulator, continued

5. From the main menu, select **Run > Debug Configurations,** or **Run Configurations.** This opens the correlating *Debug Configurations or Run Configurations* window:



Note: For simplicity, the remainder of this procedure refers only to the *Run Configurations* window, but the same instructions also apply to the *Debug Configurations* window.

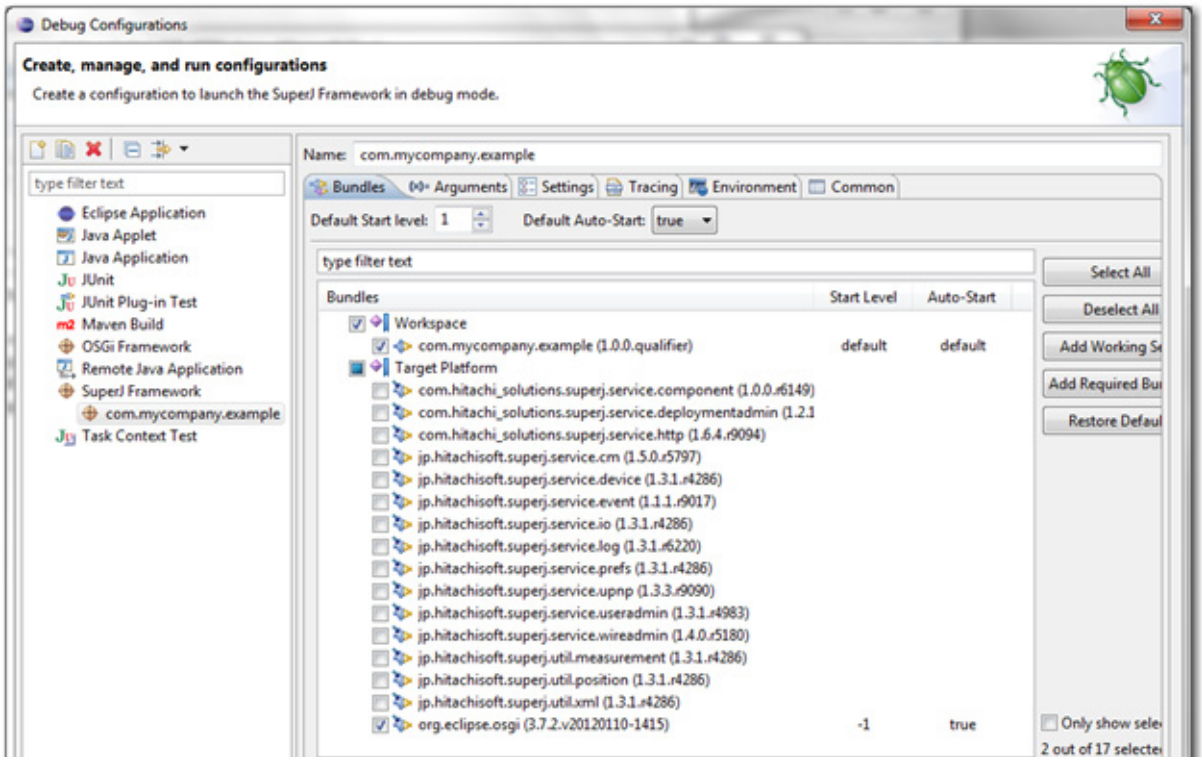
The SuperJ Framework Simulator does not launch from the run or debug links or buttons in the Overview panel of the bundle editor.

continued...▶

Using the SuperJ Framework Simulator, continued

6. From the tree view, expand the **SuperJ Framework** node, and then do one of the following:
   - Select an existing SuperJ Framework configuration, or
   - If the **SuperJ Framework** node is empty, right-click on the **SuperJ Framework** node, select **New** from the context menu to add a new SuperJ Framework configuration, and then type a **Name** for it.
7. On the **Bundles** tab, in Bundles list
   - under **Workspace**, select the bundle(s) that you want to run.
   - under **Target Platform**, ensure that `org.eclipse.osgi` is selected, and select any required service bundles.

   Note: By default, all bundles that are currently open in the Workspace are selected in [*Run/Debug*] *Configurations* windows.



The following services are included with a SuperJ Framework:

- Package Admin Service
- Start Level Service
- Permission Admin Service
- Conditional Permission Admin Service
- URL Handlers Service
- Service Tracker
- Service Hooks

You can run any of the following OSGi standard services by selecting the correlating Bundle in from the **Target Platform** list, as identified in Table 3-1 on the following page.

continued...▶

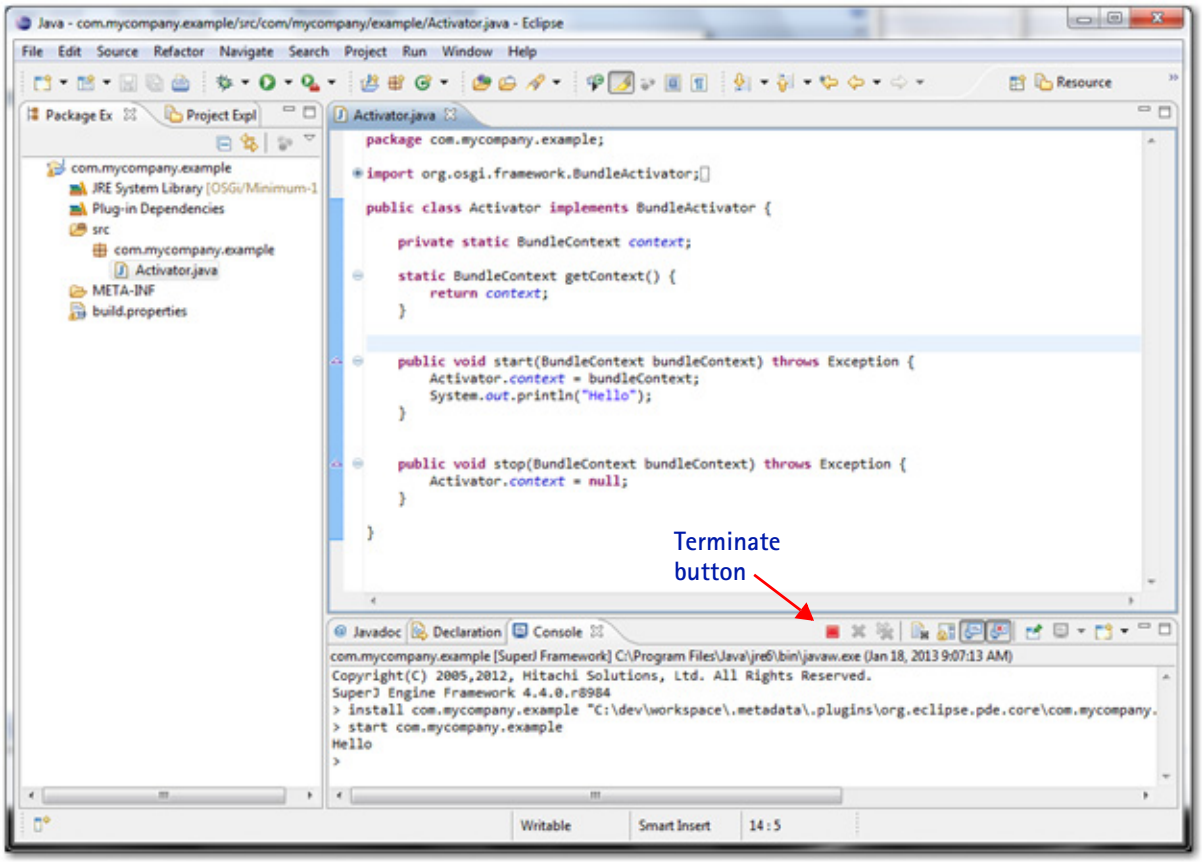
Using the SuperJ Framework Simulator, continued

Table 3-1: SJT standard services from the SuperJ Framework

Name	Provides	Bundle
XML Service	XML parser	jp.hitachisoft.superj.util.xml
		jp.hitachisoft.superj.util.xml-sax
Measurement Service	Measurements in different units	jp.hitachisoft.superj.util.measurement
Position Service	Operates position data	jp.hitachisoft.superj.util.position
Wire Admin Service	Connections for service(s)	jp.hitachisoft.superj.service.wireadmin
Log Service	Functions to output and retrieve bundle logs	jp.hitachisoft.superj.service.log
Configuration Admin Service	Configuration of services	jp.hitachisoft.superj.service.cm
Device Access Service	Access control for a device	jp.hitachisoft.superj.service.device
IO Connector Service	Expansion of generic connection framework	jp.hitachisoft.superj.service.io
Preferences Service	Controls for bundle-specific setting items	jp.hitachisoft.superj.service.prefs
User Admin Service	User authentication data control	jp.hitachisoft.superj.service.useradmin
UPnP Device Service	UPnP device control service	jp.hitachisoft.superj.service.upnp
Event Admin Service	Delivery of events between the bundles	jp.hitachisoft.superj.service.event
Declarative Services	Provides a publish/find/bind model for services; registers a service and handles service dependencies.	com.hitachi_solutions.superj.service.component
HTTP Service	Servlet container service to run a program on the web server and the server side	com.hitachi_solutions.superj.service.http
Deployment Admin	Installation / update / uninstall of multiple bundles at the same time	com.hitachi_solutions.superj.service.deployment admin

Note: The framework data directory is removed when the simulator boots regardless of options selected on the **Settings** tab in the [*Run/Debug*] *Configurations* windows.

8. Click **Run**. The SuperJ Framework Simulator and bundles start. Results display on the **Console** tab in the bottom pane of the project window:



Type `help` in the **Console** tab to get a list of available commands.

9. When you are finished, or before restarting the SuperJ Framework Simulator, click **Terminate**.

Working with Existing non-SJT Bundles

If you have a JAR bundle that was developed to run in a different environment, you can make it OSGi-compatible as described here.

Referencing external bundles in the targeted platform

You can add external bundles to the target platform of Eclipse as follows:

1. Copy the JAR files for the desired bundles to the following directory: <*EclipseInstallPath*>`\plugins\jp.hitachisoft.superj.sdk.lib_<version>\sjefbin\bundles\ext`
2. Perform all steps of "Reloading the SuperJ Framework target platform" on page 3-17.

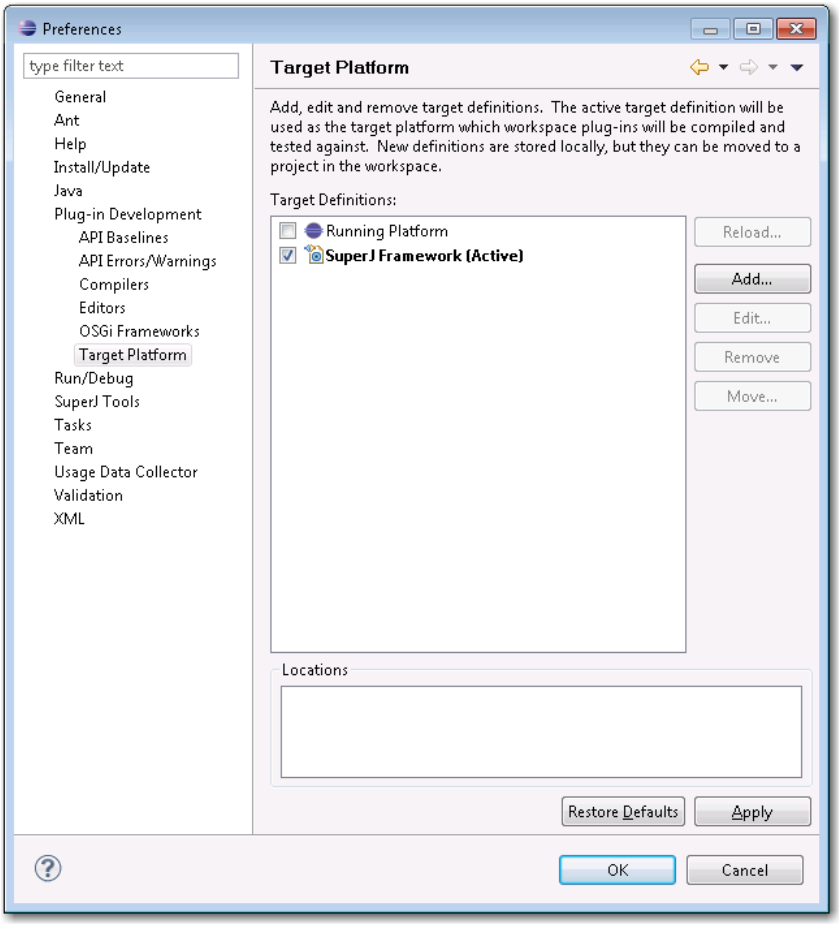
Setting SuperJ Framework as the Target OSGi Platform

This section describes how to select SuperJ Framework as the target platform for OSGi frameworks in Eclipse, and how to make sure it remains loaded as the target platform.

Setting the Target Platform

To configure SuperJ Framework as the default OSGi framework for Eclipse, use the following procedure:

1. From the main menu, select **Window > Preferences**. The *Preferences* window opens.
2. From the tree view, select **Plug-in Development > Target Platform**. The **Target Platform** list displays in the right pane. Ensure that the active selection appears as follows:

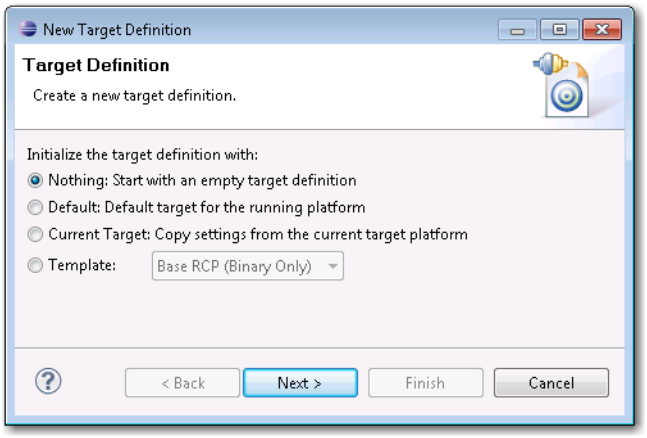


3. If **SuperJ Framework (Active)** is already selected in this window, go to Step 5 on page 3-16.

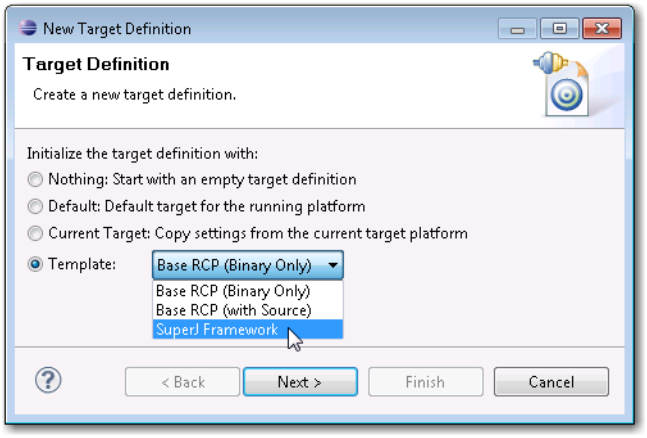
continued...

Setting the Target Platform, continued

4. If **SuperJ Framework** is not available in the list, add it as follows:
   a. Click **Add** to open the *New Target Definition* window.



   b. Select the **Template** option, select **SuperJ Framework**, and then click **Next**.
   c. Click **Finish** to return to the **Target Platform** list.

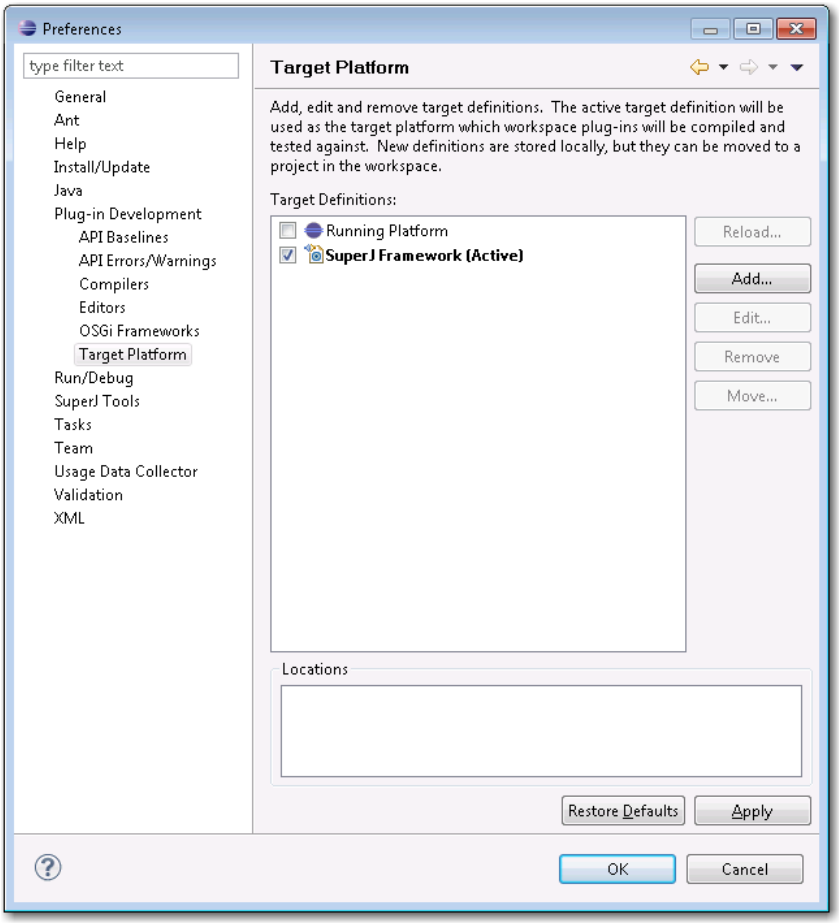


   d. Select the check box for **SuperJ Framework** to make it active.
5. Click **OK** to close the *Preferences* window.

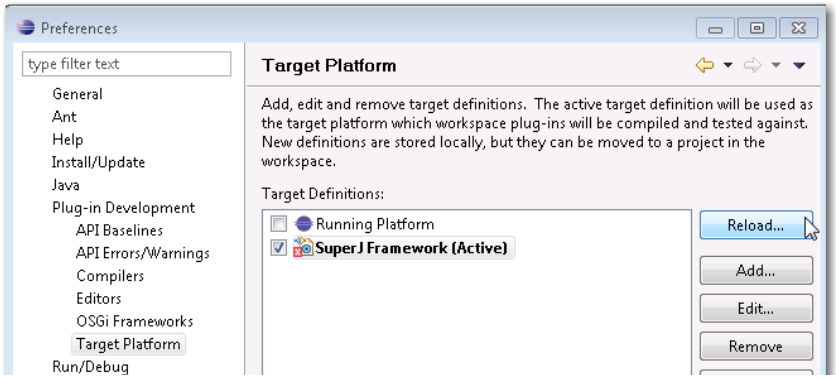
Reloading the SuperJ Framework target platform

Reload the SuperJ Framework as the targeted platform for OSGi frameworks in Eclipse as follows:

1. Choose **Window > Preferences** from main menu.
2. From the tree panel on the left, choose **Plug-in Development > Target Platform**. The **Target Definitions** list displays.



3. Select **SuperJ Framework**, and then click **Reload**.

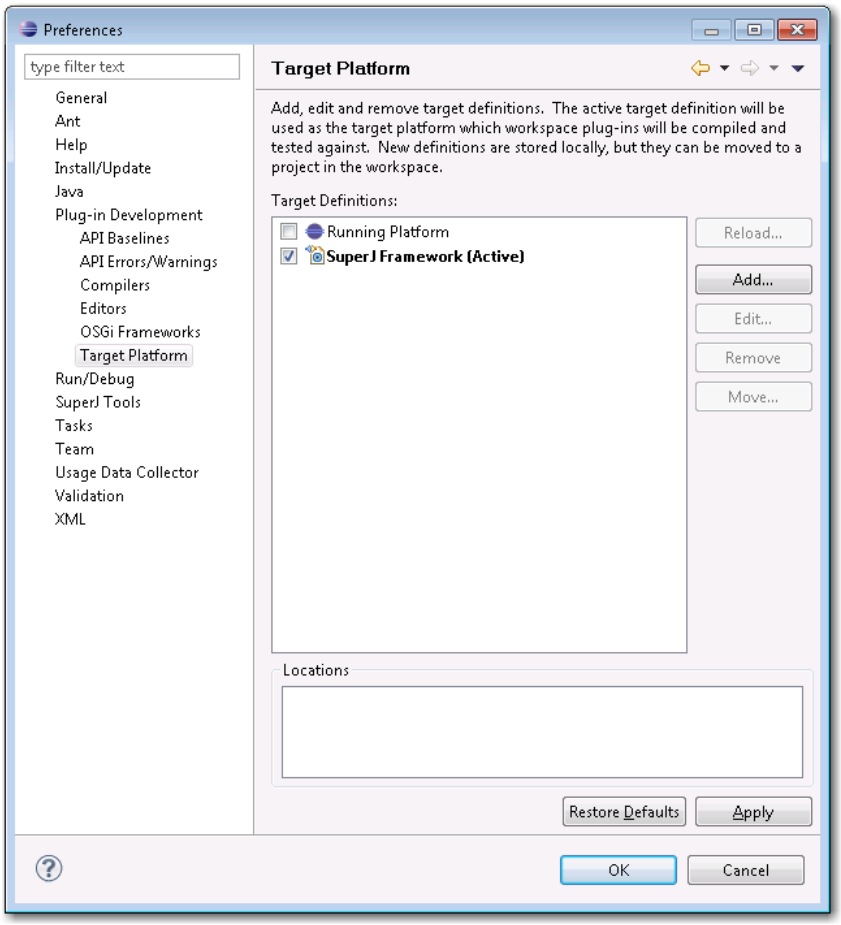


4. Click **OK** to close the *Preferences* window.

Restoring the default target platform

Sometimes when developing and/or debugging a bundle, you may opt to run/debug it in another OSGi framework such as Equinox. Whenever you run the project as an Eclipse Application or OSGi Framework (Equinox), set the target platform back to default as follows:

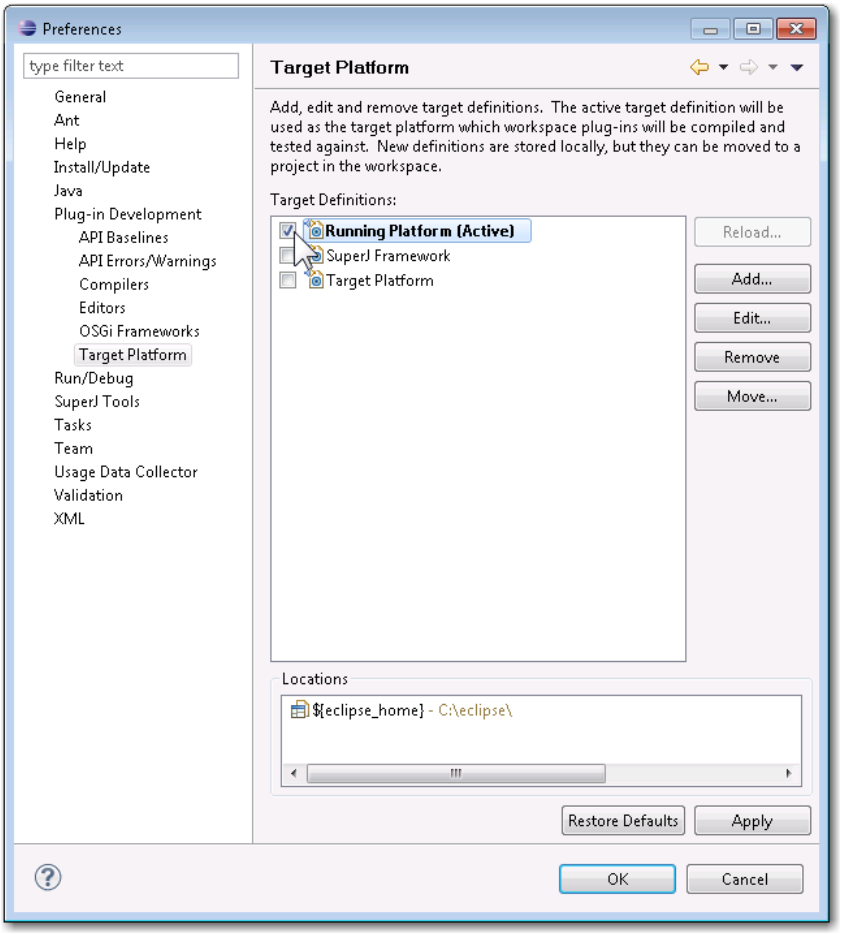
1. Choose **Window > Preferences** from main menu to open the *Preferences* window.
2. From the tree panel on the left, choose **Plug-in Development > Target Platform**. The **Target Definitions** list displays in the *Target Platform* pane.



continued...

Restoring the default target platform, continued

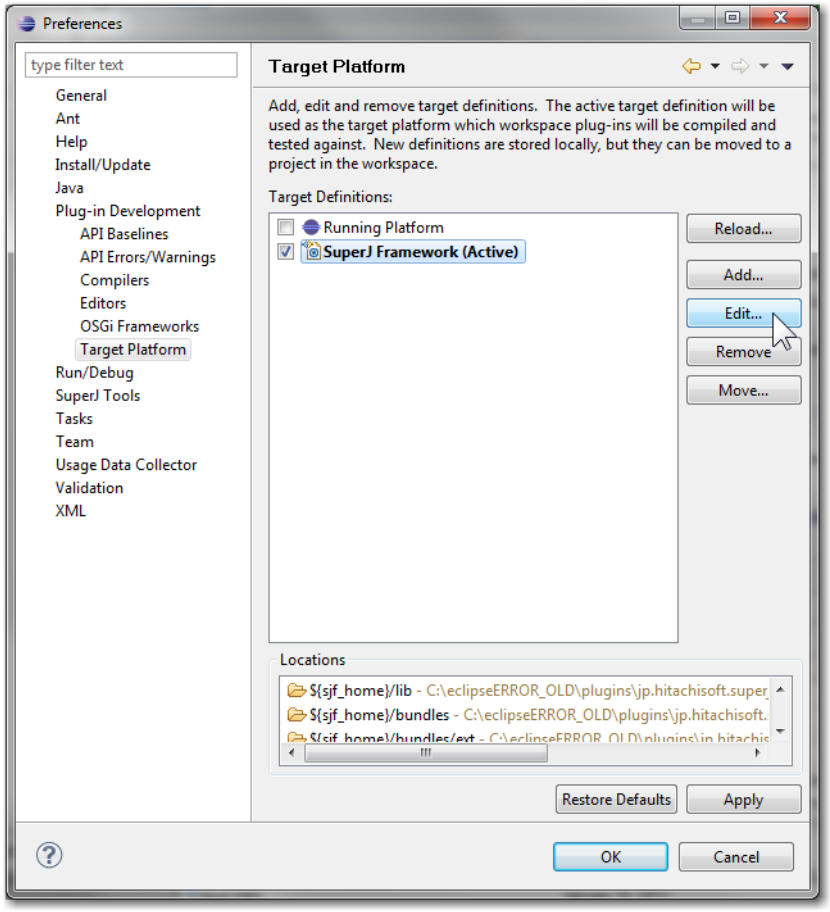
3. Select Running Platform, and then click **OK** to close the *Preferences* window.



Changing the HTTP service port for the targeted platform

The HTTP service provided with the SuperJ Framework runs on port 80 by default. If port 80 is already in use, HTTP cannot work. You can change the port number of HTTP service by editing the system property `org.osgi.service.http.port` for the targeted platform as follows:

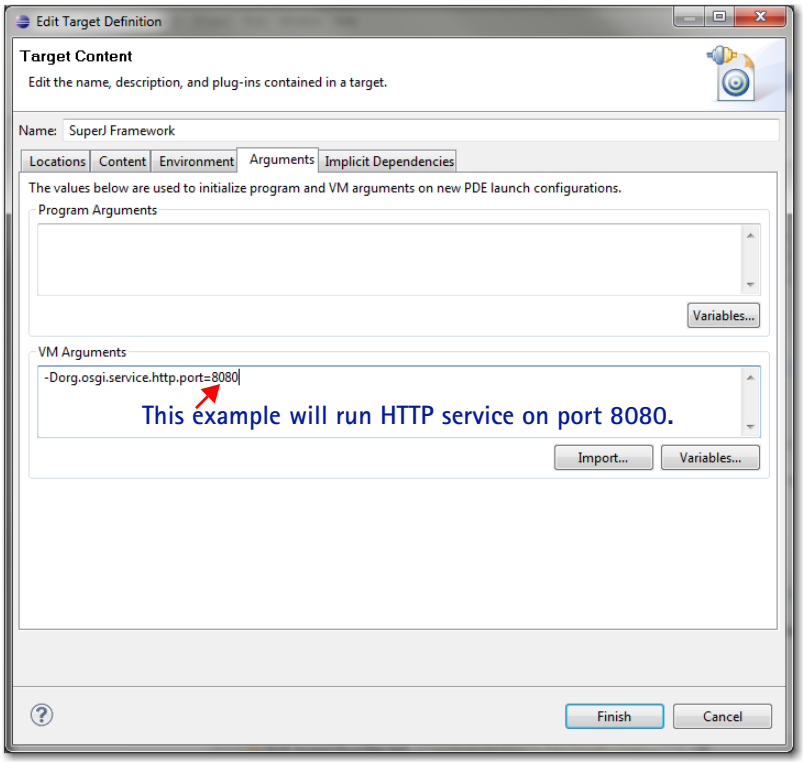
1. Choose **Window > Preferences** from main menu to open the *Preferences* window.
2. From the tree panel on the left, choose **Plug-in Development > Target Platform**. The **Target Definitions** list displays in the *Target Platform* pane.
3. Select **SuperJ Framework**, and then click **Edit**.



continued...

Changing the HTTP service port for the targeted platform, continued

4. Click the **Arguments** tab.
5. Add to **VM arguments**:
   `-Dorg.osgi.service.http.port=<desiredPortNumber>`



6. Click **Finish**.
7. Click **OK** to close the *Preferences* window.

4 WORKING WITH DEPLOYMENT PACKAGES

This chapter describes how to create deployment packages for SJF.

Contents

Topics covered in this document include: **Page**

Note: The first topic begins on the following page.

Deployment Packages and SJT

A deployment package is a group of resources that can be installed, updated, and uninstalled as a unit. You can deploy groups of bundles as services using deployment packages.

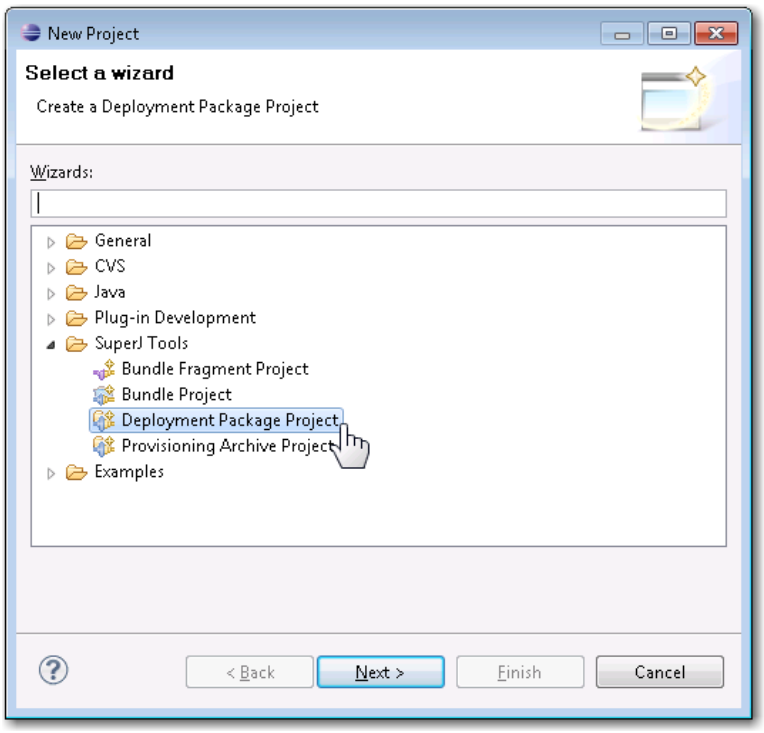
A deployment package consists of two types of installable resources that are described in the Name sections of the MANIFEST.MF and are stored in the JAR file under a path called the Resource ID:

Resource Type	Description
Bundle resource	Subsets of installable resources that are treated differently from the processed resources by the Deployment Admin service.
Non-bundle resources	Non-JAR resources, such as image files for an app icon, readme.txt or other text files, configuration files for the app, etc.

Creating a Deployment Package Project

The first step in creating a deployment package project is to create a deployment package project as described in this procedure.

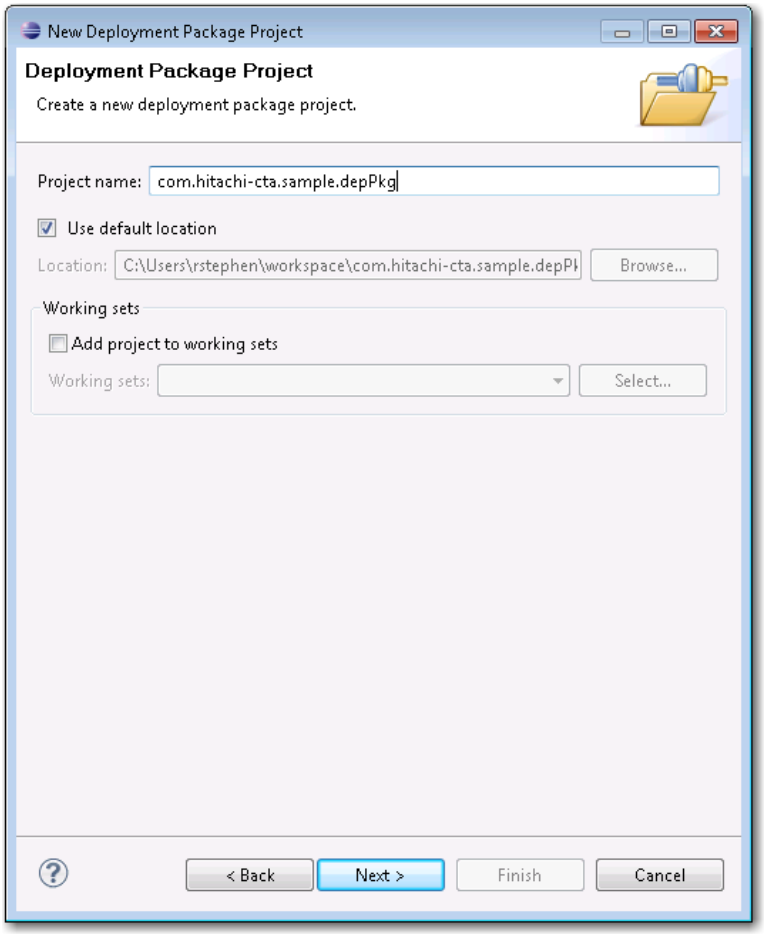
1. From the main menu, select **File > New > Project**. The *New Project* window opens.
2. Under **SuperJ Tools**, select **Deployment Package Project**.



continued...

Deployment Packages and SJT, continued

3. Click **Next**. This opens the **Deployment Package Project** page in the *New Deployment Package Project* window.



4. Do the following:
   a. Type the **Project name** following the reverse-URL naming convention for the project (groupID) and appending the package name (artifactID).
   b. Select a file location for the deployment package by either selecting **Use default location** or deselecting that option and choosing another file location.
   c. If no working sets are defined, do not select to **Add project to working set**. If you want the bundle to be assigned to a Java working set, select this option and choose from the list of working sets.
   d. Click **Next**.

continued...

Deployment Packages and SJT, continued

5. Complete the **Content** panel as described in the following table:

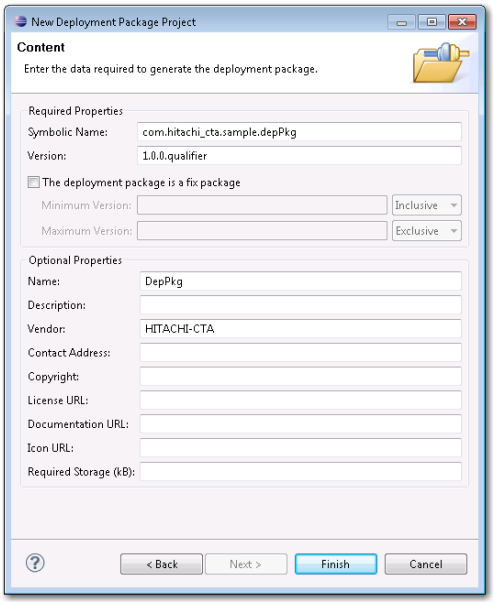


Table 4–1: General information about the deployment package

Field		Description[a]
Required Properties	**Symbolic Name**	Specifies a symbolic name. (e.g. *groupID.artifactID*)
	Version	Specifies a version for the deployment package.
	The deployment package is a fix package • Minimum Version • Maximum version	Select when creating archive of Fix Package, and then: • Type the version requirements targeted for the fix. • Select whether Inclusive or Exclusive of the specified version.
Optional Properties (to populate manifest.mf for this package)	**Name**	Type the name of this deployment package.
	Description	Describe this deployment package.
	Vendor	Identify the vendor of the deployment package.
	Contact Address	Type contact information for the vendor/developer, such as an e-mail address.
	Copyright	Type the copyright statement for this package.
	License URL	Provide a URL for a license file for the this package.
	Documentation URL	Provide a URL to any document for this package.
	Icon URL	Specify a URL to an icon file for this package.
	Required Storage (kB)	Specify the minimum amount of persistent storage required by this deployment package as a product.

a. See the "Deployment Admin Specification" in the *OSGi Service Platform Service Compendium* as described in"Related documents" on page viii for details.

6. Click **Finish**. The deployment package project is ready for adding bundles to it, and the **Overview** tab is populated with this information.

Using the Deployment Package Editor

The deployment package editor opens automatically when you create a project. To open the deployment package editor manually:

1. In the Package Explorer tree view, find the package you want to edit, open its **META-INF** folder, and then double-click on **MANIFEST.MF**.

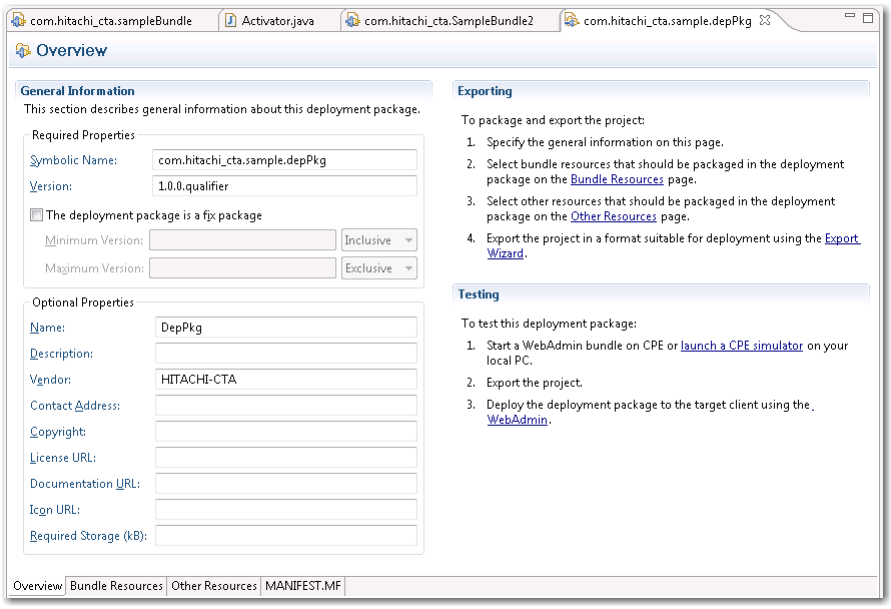


Figure 4-1: Overview tab

The deployment package editor contains the following tabs:

Tab	Content
Overview	Deployment package information and links for exporting and testing the deployment package
Bundle Resources	Fields for viewing or changing the bundles / bundle projects in the deployment package, and for entering information about each bundle/project.
Other Resources	Fields for viewing or changing the non-bundle resources in the deployment package, and for entering information about each resource
MANIFEST.MF	Manifest for this deployment package

IMPORTANT

When editing MANIFEST.MF, do not edit the Name header that was added by default.

Editing deployment package information

You can edit information about the deployment package from the **Overview** tab (see Figure 4-1 on page 4-5). The following settings are available:

Table 4–2: Field description of Overview tab for the deployment package

Field		Description[a]
Required Properties	**Symbolic Name**	Symbolic name. (e.g. *groupID.artifactID*)
	Version	Version for the deployment package.
	The deployment package is a fix package • Minimum Version • Maximum version	If creating archive of Fix Package, this specifies: • Version requirements for the fix. • whether version number is Inclusive or Exclusive
Optional Properties (to populate the manifest.mf for this package)[b]	**Name**	Name of this deployment package.
	Description	Description of this deployment package.
	Vendor	Vendor of the deployment package.
	Contact Address	Contact information for the vendor/developer, such as an e-mail address.
	Copyright	Copyright statement for this package.
	License URL	URL for a license file for the this package.
	Documentation URL	URL to any document for this package.
	Icon URL	URL to an icon file for this package.
	Required Storage (kB)	Minimum amount of persistent storage required by this deployment package as a product.

a. For additional details for each item, see the "Deployment Admin Specification" in the *OSGi Service Platform Service Compendium* as described in"Related documents" on page viii.

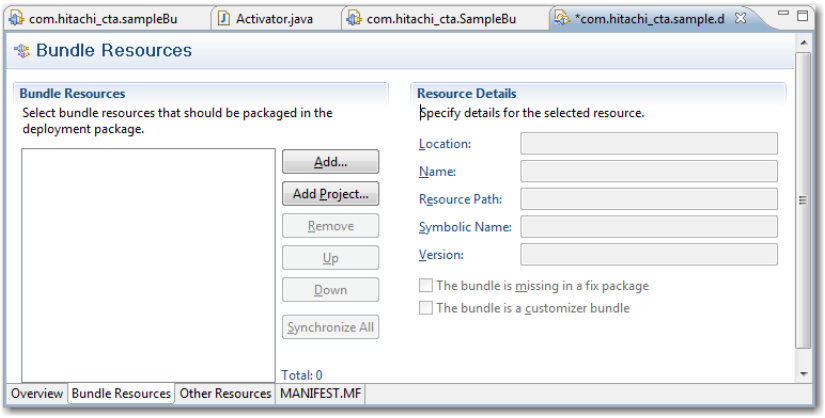
b. You can provide this information later when releasing the package as a product.

Editing the Bundle Resource list

You can add or remove bundle(s) included in a deployment package or change the order in the archive for each of the bundles in a deployment package by changing the order in the list on the **Bundle Resources** tab.

> **IMPORTANT**
>
> The Eclipse menu option **Edit > Undo** or [Ctrl-Z] does not work for this window. Changes are effective immediately upon your action in real time. If you want to be able to undo changes if results are unanticipated, you can coordinate when you save the file to permit **File > Revert** to restore to a desired state.

1. If the deployment package you wish to edit is not open, select it from the Package Explorer tree view.
2. To change bundle resources, click the **Bundle Resources** tab.



continued...

Editing the Bundle Resource list, continued

The following table coordinates with the figure on the preceding page:

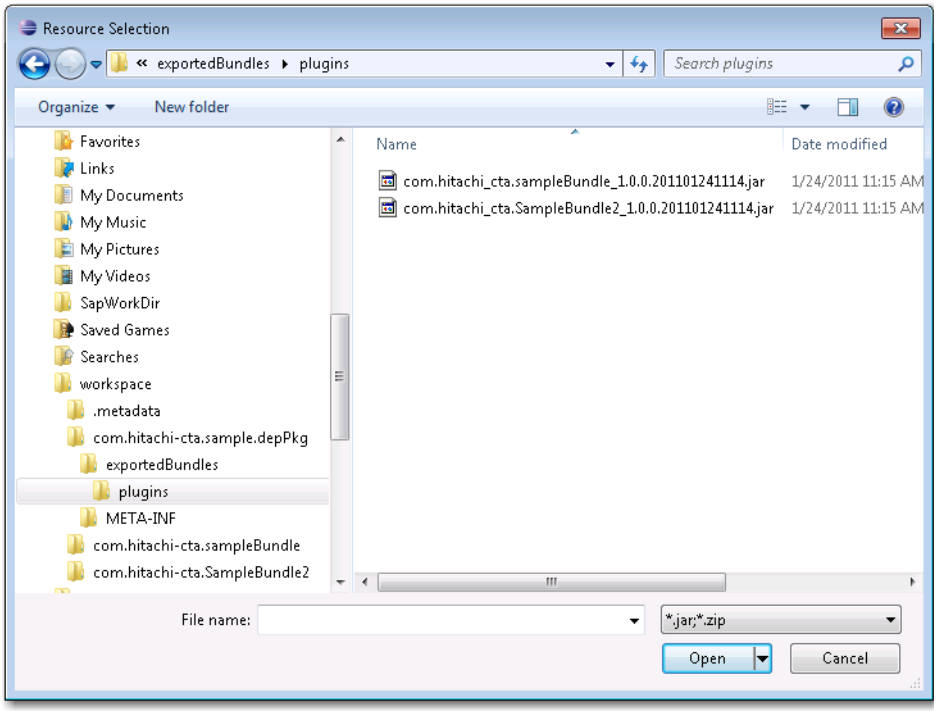
Item		Description
Bundle Resources area	**Add** button	Opens the *Resource Selection* window so that you can choose a JAR file for the bundle resource you want to include in this deployment package. The selected JAR file then shows in the Bundle Resources list and explicitly in the manifest.mf.[a]
	Add Project button	Click to include a different bundle development project in your workspace to the archive target of this deployment package.
	Remove button	Removes the JAR file for the selected bundle resource from the archive target for the deployment package.[b]
	Up button	Promotes the placement of the bundle resource as stored in the deployment package archive. Bundles are executed sequentially based on their order in this list.
	Down button	Demotes the placement of the bundle resource as stored in the deployment package archive. Bundles are executed sequentially based on their order in this list.
	Synchronize All button	Reads symbolic name and version information from bundle resources in this deployment package and updates the descriptions in MANIFEST.MF.
Resource Details area	**Location** field	Read-only field that displays the fully qualified path to the item selected in the Bundle Resources list.
	Name field	Read-only field that displays the actual file name (the contents of the **Resource Path** field plus the **Symbolic Name** field plus the **Version** field) for the JAR file of the item selected in the Bundle Resources list.
	Resource Path field	Type the path to store this resource in the deployment package archive
	Symbolic Name field	Read-only field that displays the Resource ID for the bundle resource.
	Version field	Read-only field that displays the Version for the bundle resource.
	The bundle is missing in a fix package check box	The selected bundle resources will not be stored in the deployment package archive. Use only if the selected bundle is being deployed to a correction/patch.
	The bundle is a customizer bundle check box	Specifies whether or not a bundle is a customizer bundle.

a. Updating the bundle does not affect deployment package archive output.
b. Removing the JAR file from the Bundle Resources list only removes it from the bundles folder within the directory for the deployment package you are currently editing.

continued...▶

Editing the Bundle Resource list, continued

3. To include an existing bundle resource (JAR) in this deployment package, click **Add**. The *Resource Selection* window opens.

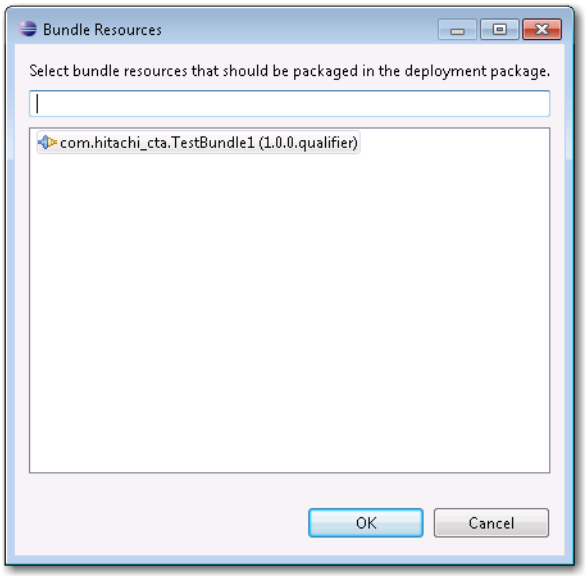


Note: Use the <Shift> or <Control> keys to select more than one file.

4. Navigate to the JAR file for the desired bundle resource, and then click **Open**. The selected JAR file is added to the list of bundle resources for this deployment package.

5. If you want to add a bundle project to this deployment package:

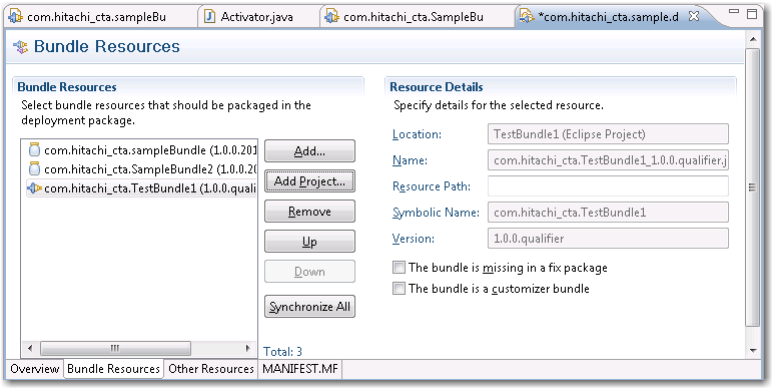
   a. Click **Add Project**. The *Bundle Resources* window opens.



continued...▶

Editing the Bundle Resource list, continued

b. Select the bundle development project you want to add, and then click **OK**. The list of included bundle resources on the Bundle Resources



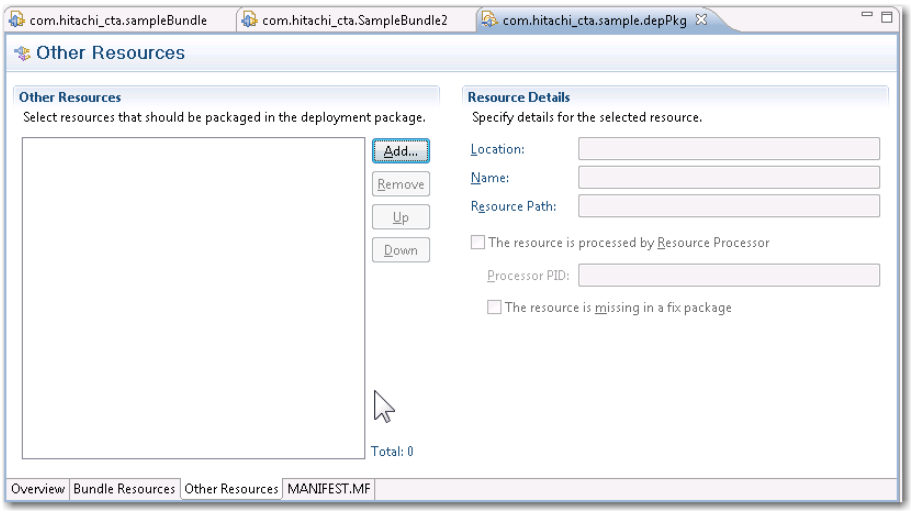
6. Bundles are executed from the top of the list downward. Click **Up** and/or **Down** as needed for each bundle to create the desired order of bundle resources for this deployment package.

7. When you are satisfied with the results on this tab, **Save** the deployment package.

   Note: You can continue making changes to the deployment package for ongoing development purposes.

Editing Other Resources list

You can use other resources that are not JAR bundles or plug-ins to a deployment package using the Other **Resources** tab:



continued...

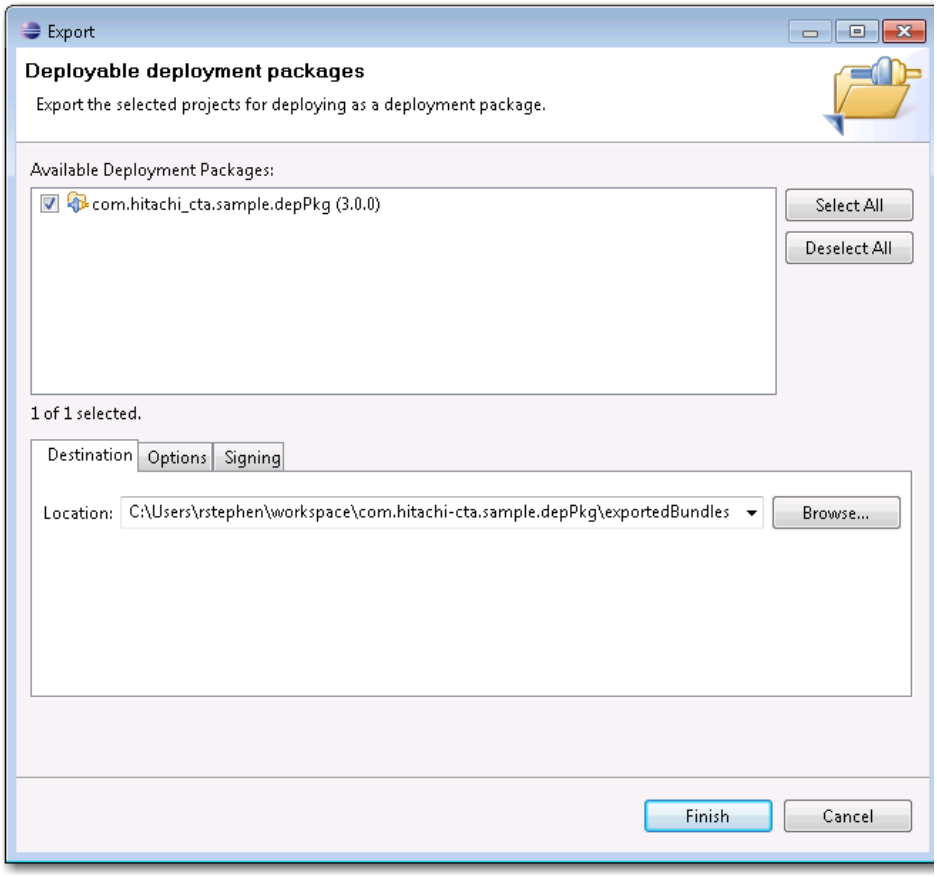
Editing Other Resources list, continued

Item		Description
Other Resources area	**Add** button	Opens the *Resource Selection* window so that you can choose a file for the processed resource you want to include in this deployment package. The selected file then shows in the Other Resources list and is copied to the resources directory within the directory for the deployment package.
	Remove button	Deletes the file for the selected resource that was copied into the bundles folder in the directory for the deployment package. This file is also removed as an archive target from the deployment package.
	Up button	Promotes the placement of the resource as stored in the deployment package archive. Executed resources are executed sequentially based on their order in this list.
	Down button	Demotes the placement of the resource as stored in the deployment package archive. Executed resources are executed sequentially based on their order in this list.
Resource Details area	**Location** field	Read-only field that displays the fully qualified path to the item selected in the **Other Resources** list.
	Name field	Read-only field that displays the actual file name for the file of the item selected in the **Other Resources** list.
	Resource Path field	Type the relative path to store this resource in the deployment package archive
	The resource is processed by Resource Processor check box	Specifies whether or not a resource is a processed resource. When this check box is selected, the **Processor PID** field becomes available.
	Processor PID field	Specifies the persistent identifier for the selected resource's processor.
	The resource is missing in a fix package check box	The selected resources will not be stored in the deployment package archive. Only available if **The deployment package** is a fix package is selected on the Overview tab. Use only if the selected resource is being deployed to a correction/patch.

Exporting a deployment package archive

You can create an archive file for your deployment package with following procedure.

1. When you have the bundle resources showing as desired for deployment, double-click the MANIFEST.MF file for the deployment package project in the Package Explorer tree view.
2. Select the **Overview** tab, and then click **Export Wizard** to create the deployment package. The *Export* window opens.



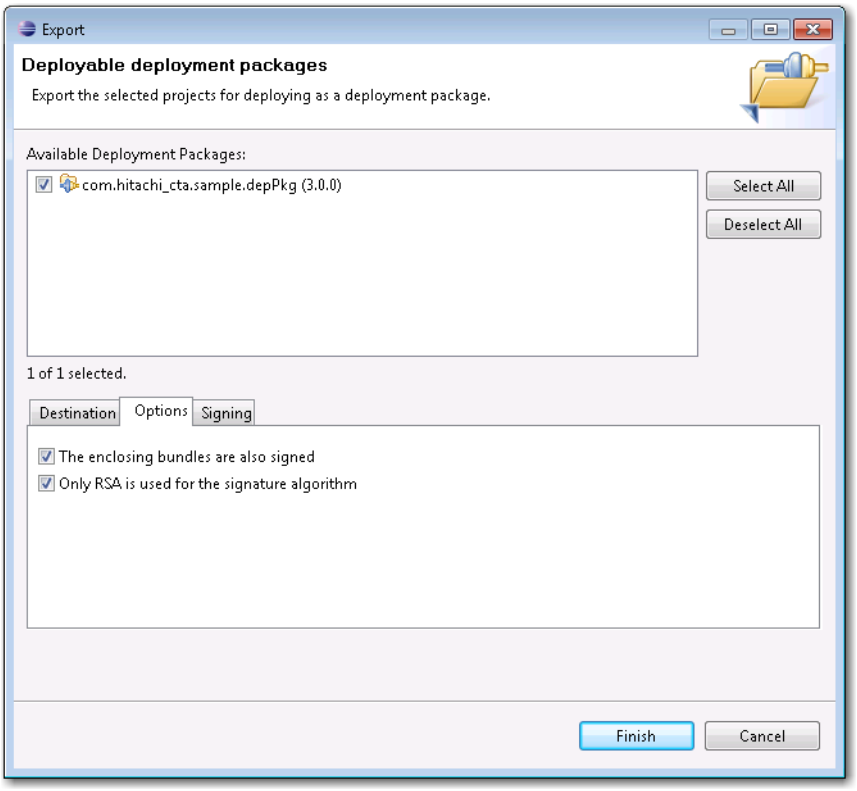
Note: You can export more than one deployment package by selecting all the check boxes for the deployment packages you want to export.

3. Select the desired deployment package(s) from the **Available Deployment Packages** list.
4. Choose a path for the deployment package by choosing a **Directory** from the list, or by clicking **Browse** to locate the desired folder.

continued...▶

Exporting a deployment package archive, continued

5. Make any desired changes on the **Options** tab.



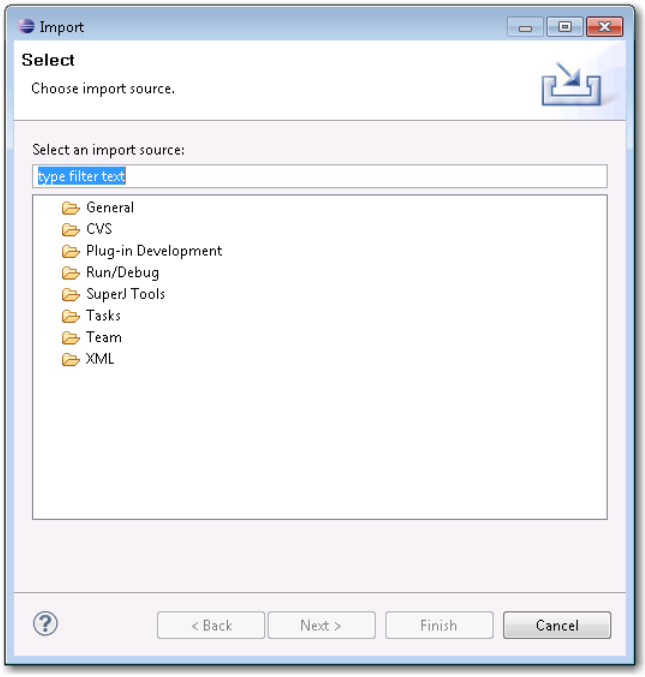
Export Option	Description
The enclosing bundles are also signed	If you select this check box, when the deployment package is signed, its enclosed bundles will also be signed.
Only RSA is used for the signature algorithm	If you select this check box, the signer program will accept only the RSA algorithm as a key.

6. If desired, and if you have complete information for authentication, add a digital signature on the **Signing** tab. Do not use this option unless you have all of the information for the Keystore and authentication.
7. Click **Finish**.

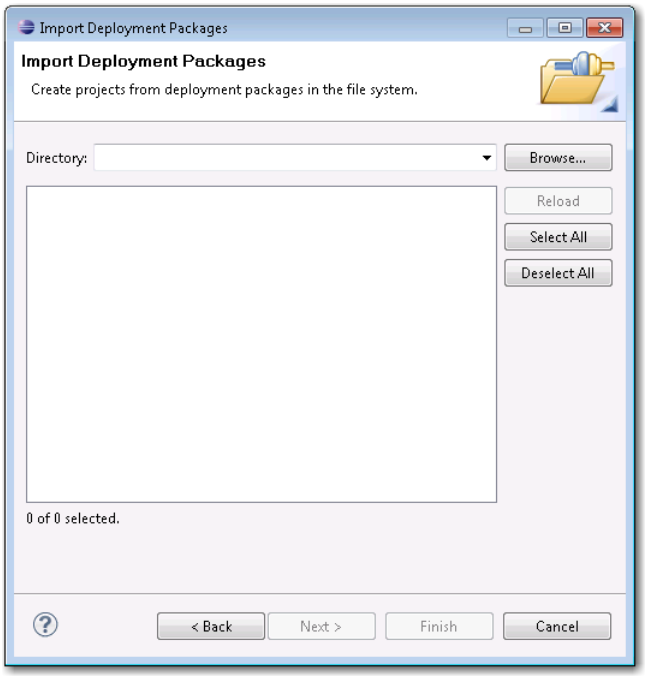
Importing a deployment package

You can import deployment packages from other projects as follows:

1. Select **File > Import** from the main menu of Eclipse to open the *Import* window.



2. Select **SuperJ Tools > Deployment packages**, and then click **Next** to open the **Import Deployment Packages** pane in the *Import* window.



continued...

Importing a deployment package, continued

3. Locate the desired directory in either of these ways:
   - Type a path in the **Directory** field, and then click **Reload**.
   - Click **Browse** to navigate to the desired directory.
4. Select the projects, and then click **Finish**.
5. If a selected file is signed, a delete confirmation window opens. Choose signature files not delete if you want to make the project read-only.

5 USING THE CPE SIMULATOR

This chapter describes how to use the CPE Simulator to test SJF deployment packages.

Contents

Topics covered in this document include: **Page**

Note: The first topic begins on the following page.

Testing Deployment Packages

You can test deployment packages using the SJF WebAdmin tool. There are two types of deployment targets:

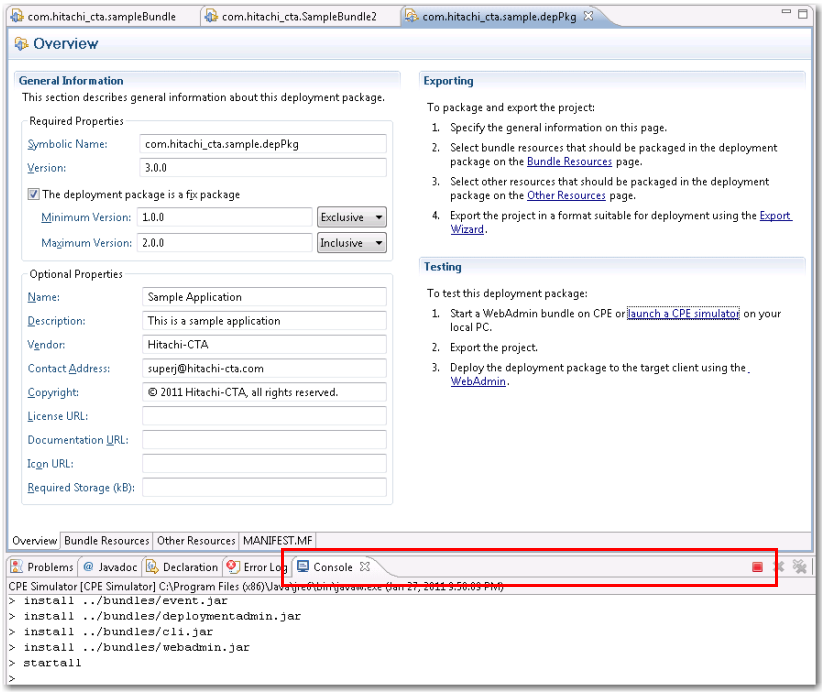
- CPE Simulator running on the local PC (local host)
- Deploying the packages to the target CPE using the WebAdmin tool

The WebAdmin tool installs a framework administration tool to the CPE that is on the same LAN. This section describes how to use the CPE Simulator and WebAdmin bundle as development tools for testing deployment packages.

Using the CPE Simulator to test deployment packages

In order to confirm that the creating deployment package runs properly on the CPE, package archives can be test-deployed in the CPE Simulator. Boot the CPE simulator on the local PC with the following procedure.

1. Open the **Overview** tab.
2. Click the **launch a CPE Simulator** link.
3. Confirm that the CPE simulator is running by looking at the Eclipse standard **Console** tab.

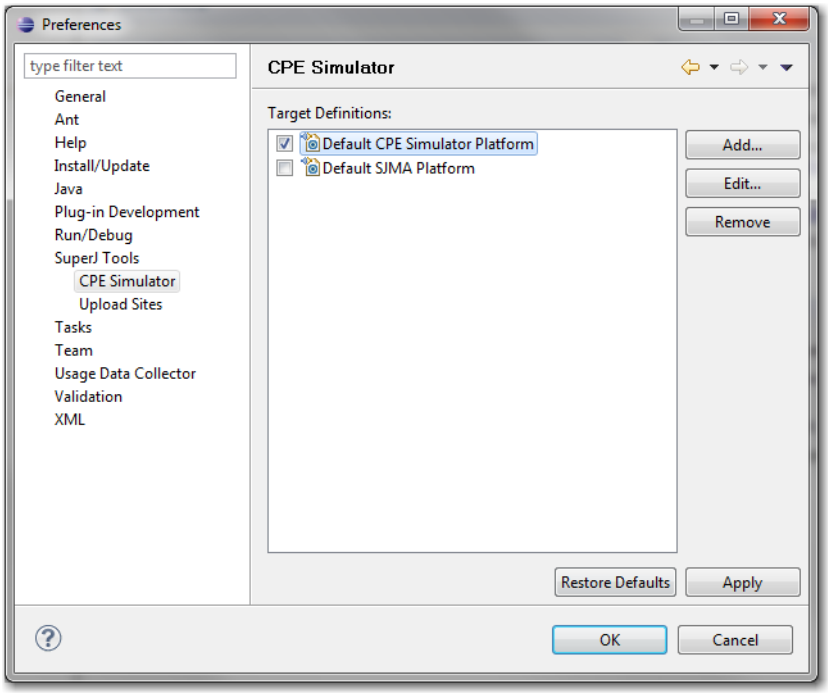


4. When you are finished, click **Terminate** on the Eclipse standard console view to stop the CPE simulator.

Custom-booting the CPE Simulator

Booting options of CPE simulator can be customized with the following procedure.

1. Choose **Window > Preferences** from main menu to open *Preferences* window.
2. Choose **SuperJ Tools > CPE** from the list of setting items in the tree panel, and then select **Default CPE Simulator Platform** as shown here:



3. Click **OK** to save the setting and close the *Preferences* window.

Using the WebAdmin Tab

This section describes how to use the WebAdmin tab in conjunction with the CPE Simulator to test bundles and deployment packages. WebAdmin provides the following features:

Bundle Management	OSGi Published Services	Java Packages
Deployment Packages	Permission Management	Framework Property
System Property	Log Viewer	Command Line Interface

Note: The newly-developed WebAdmin tab is not a fully-tested feature of the SJT at the time of this publication. Content for this section is being developed.

For the WebAdmin tab to function properly, you must perform "SJF Web Console Feature List" on page 5-5—page 5-8.

Introduction

The SJF Web Console is a web-based tool for managing bundles on the SuperJ Framework (SJF). The SJF Web Console provides bundle lifecycle management (install, start, update, stop, uninstall) and bundle state management (header information, and dependencies on packages and services).

Web Browser

The SJF Web Console requires a Web browser to operate. We recommend the following browsers:

- Microsoft Internet Explorer ver.7.0 or higher
- Mozilla Firefox version 3.6 or higher

JavaScript must be enabled in the browser used.

SJF Web Console Feature List

Table 5-1 shows the features provided by the SJF Web Console.

Table 5–1: SJF Web Console Features (Page 1 of 2)

Tab	Feature	Description
Bundles	Detailed information about bundles	Displays information about the installed bundles
	Installation	Installs the bundle. There are two installation methods: • specifying the URL • specifying a file name
	Start	Starts the bundle
	Update	Updates the bundle
	Stop	Stops the bundle
	Uninstallation	Uninstalls the bundle
	Setting the start level (Set)	Sets the start level of the bundle
Java Packages	Detailed information on packages	Displays a list of packages that are exported by bundles
	Package refresh	Updates package dependencies between bundles (export / import)
Deployment Packages	Detailed information on deployment packages	Displays information about installed deployment packages
	Installation	Installs the deployment package. There are two installation methods: • specifying the URL • specifying a file name
	Start	Starts the bundle in the deployment package
	Update	Updates a deployment package
	Stop	Stops the bundles in the deployment package
	Uninstallation	Uninstalls the deployment package
Permissions	View a list of permissions	Displays a list of permissions to a particular bundle
	Setting Permissions	Sets the permissions given to a particular bundle. This can be done before the bundle is applied.
	Permission initialization	Initializes minimum required permissions

Table 5-1: SJF Web Console Features (Page 2 of 2)

Tab	Feature	Description
Services		Displays information about the services registered by the bundle
System Properties		Displays system properties
Logs	Displays log information generated by the logging service	
Framework Properties	Set the framework start level	Sets the framework start level
	Setting the initial bundle start level	Sets the initial bundle start level.

Overall system configuration

The SJF Web Console is available as a bundle that runs on the SuperJ Framework. The SJF Web Console provides a servlet to manipulate the framework using the HTTP service. You operate the SJF Web Console using a Web browser on a client PC. Figure 5-1 shows the system configuration for the SJF Web Console.

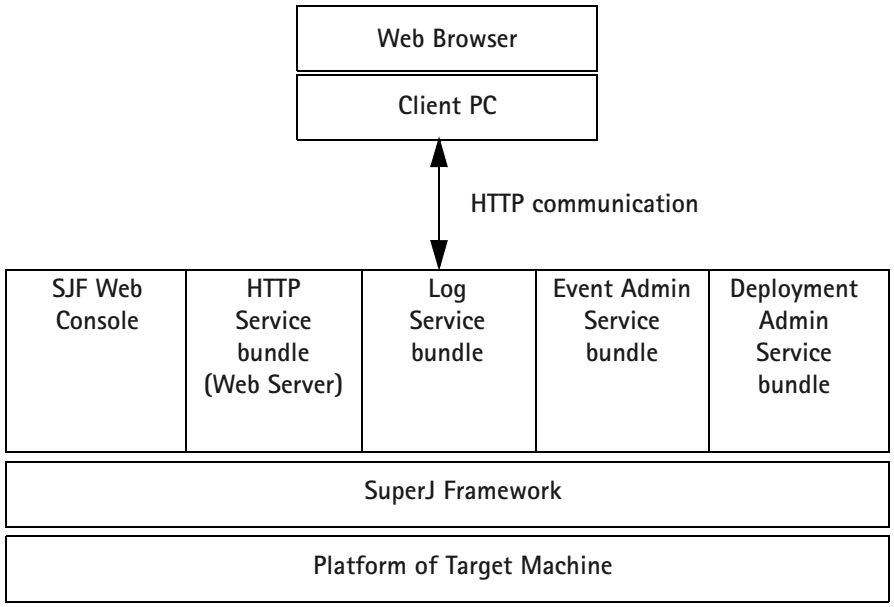


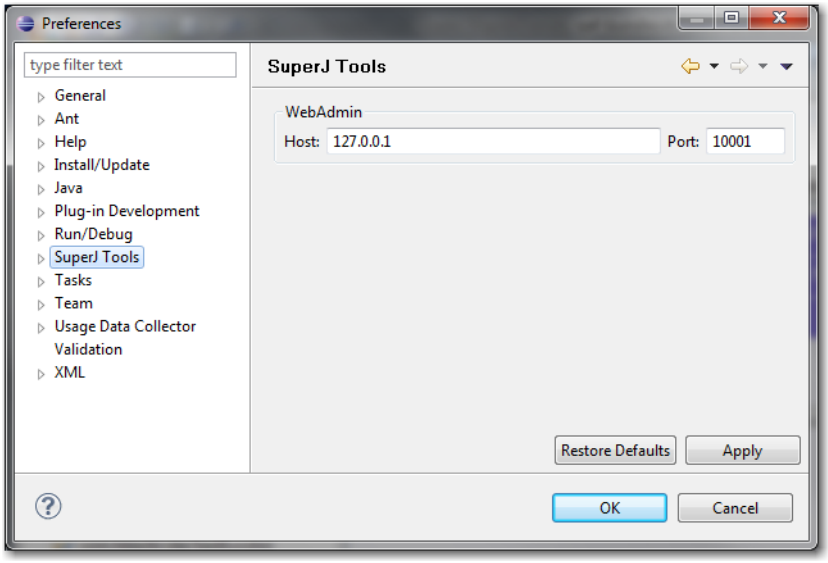
Figure 5-1: System Configuration

Preparing to use the WebAdmin tab

Follow this procedure to set up the WebAdmin tool so that you can test-deploy a deployment package with the CPE Simulator.

1. Choose **Window > Preference** from main menu to open *Preferences* window.

2. Choose **SuperJ Tools** from the tree panel to access the Host and Port settings for the WebAdmin tool.



continued...

SJF Web Console Feature List, continued

3. Edit the WebAdmin Host and Port settings as follows:

Target	Host	HTTP Port Number
Local CPE Simulator	127.0.0.1	Determined when booting the CPE Note: The default value is 10001.
Remote SJMA	Remote SJMA address	Determined when booting the CPE

4. Click **OK** to close the *Preferences* window.
5. Export the deployment package project to a file as described in "Exporting a deployment package archive" on page 4-12.

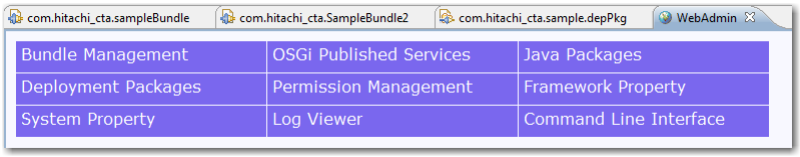
IMPORTANT!

- The CPE simulator cannot run more than one process. If you start the simulator while another process running, an error message displays.
- The framework data directory is removed when the CPE simulator starts, regardless of `-init` options set in Program Arguments.

Using the WebAdmin tab with the CPE Simulator to test bundles

Once the CPE Simulator is running and the WebAdmin host and port settings direct SJT as desired, you can begin testing bundle management options as follows:

1. Open the **Overview** tab for the deployment package.
2. In the Testing area, click the **WebAdmin** link to open the *WebAdmin* tab.



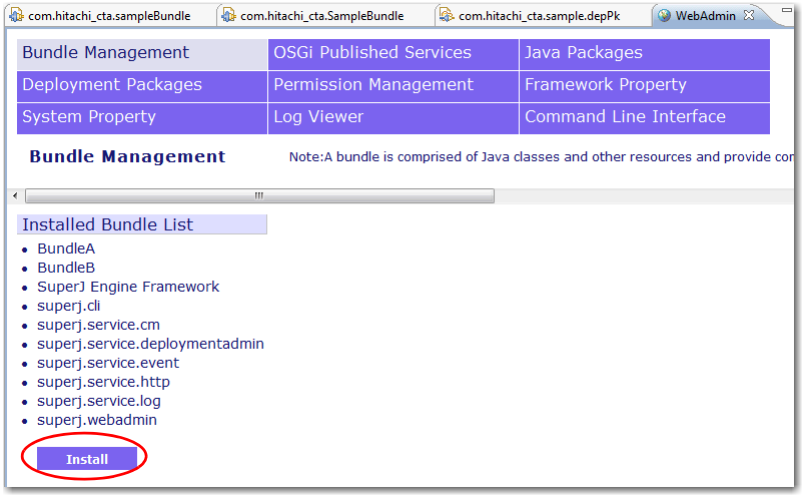
IMPORTANT!

Note the following items when starting the SJF Web Console.

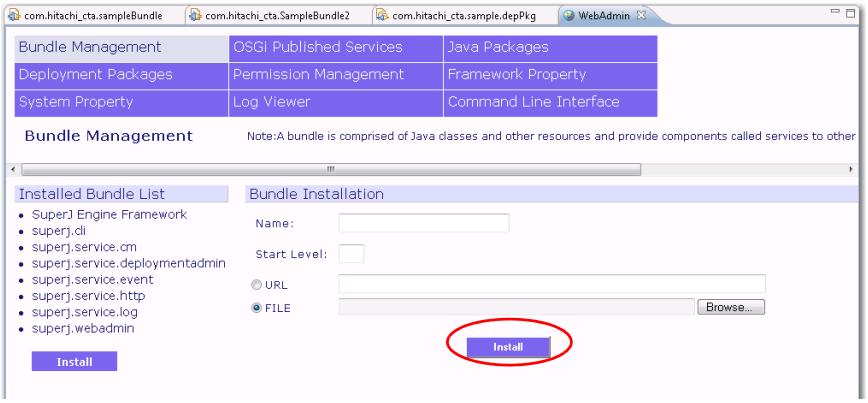
- If a bundle required for the SJF Web Console is uninstalled, the framework will not work.
- When a required bundle other than sjef.jar is updated, the framework becomes temporarily unusable.
- If the framework is updated, it becomes temporarily unusable.
- Starting the SJF Web Console may cause a framework update, which could cause the framework to shut down. If this occurs, restart the framework.

Using the WebAdmin tab with the CPE Simulator to test bundles, continued

3. Click **Bundle Management**. The **Installed Bundles List** displays.



4. Click **Install**. The **Bundle Installation** area opens.

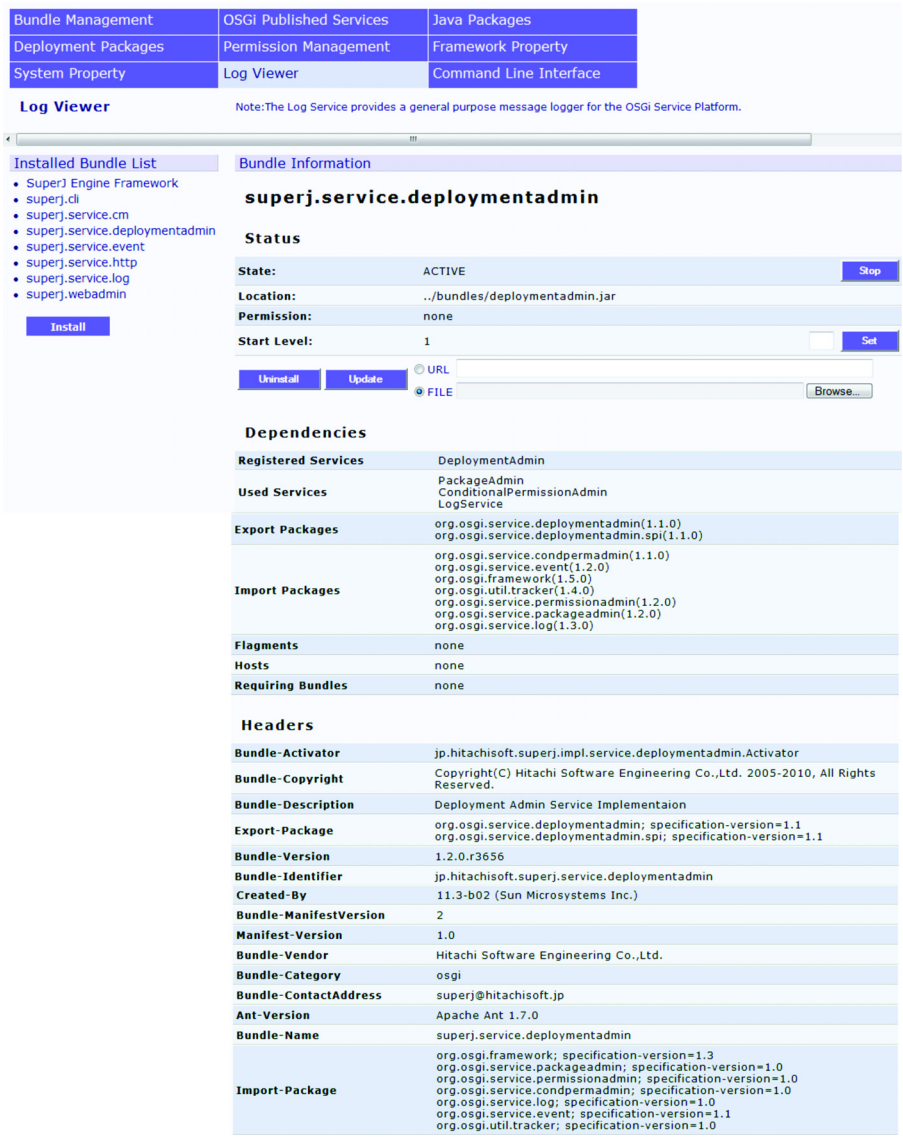


5. Type the URL or click **Browse** to navigate to the file location of a bundle you wish to install, and then click the **Install** button in the **Bundle Installation** area. If installation succeeds, a message displays.

continued...▶

Using the WebAdmin tab with the CPE Simulator to test bundles, continued

6. To view detailed information about an installed bundle, click the desired bundle from the **Installed Bundle List:**



continued...

Using the WebAdmin tab with the CPE Simulator to test bundles, continued

The following functions are available from the **Bundle Information** list:

- Click **Stop** to deactivate the selected bundle.
- To change the desired Start Level by type the desired integer in the field and click **Set**.
- Click **Uninstall** to remove the selected bundle.
- Click **Update** to provide a URL or file path to an updated JAR file for this bundle.
- Items listed in the **Used Services** heading of the **Dependencies** area are hyperlinked to the Service Information as described in “Testing OSGi Published Services” on page 5-24.
- Items listed in the **Export Packages** and **Import Packages** headings of the **Dependencies** area are hyperlinked to the Java Package Information for the correlating service/package, as described in “Viewing Java package information” on page 5-39.
- All other fields in the **Dependencies** area are display-only.

Managing Bundles

This section describes how to view bundle information, and how to install, uninstall, update, start, and stop bundles using the SJF Web Console.

Viewing bundle information

The following describes how to display information about a bundle, and describes the bundle information available.

1. On the SJF Web Console toolbar, click **Bundles**.
2. In the **Bundle List**, click a bundle. Detailed information about the bundle is displayed.

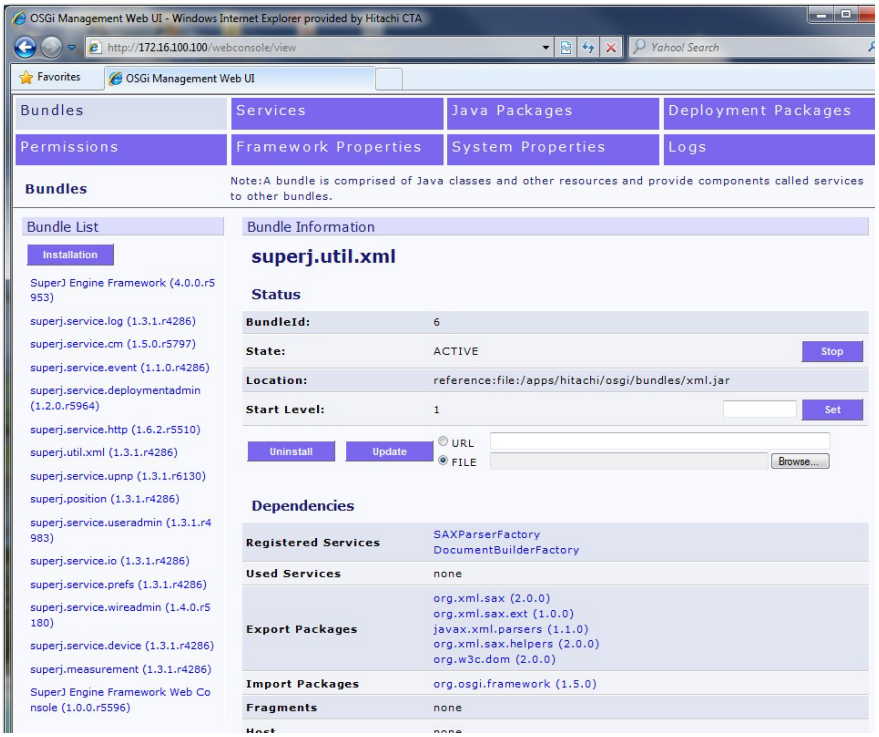


Figure 5-2: Display of bundle information

Table 5-2: Bundle information (Page 1 of 2)

Group	Item	Description
Status	BundleId	ID of the bundle
	State	State of the bundle.
	Location	File location of the bundle
	Permission	Permissions set for the bundle
	Start Level	Start level set for the bundle
Uninstall		Uninstalls the bundle
Update		Updates the bundle

Table 5-2: Bundle information (Page 2 of 2)

Group	Item	Description
Dependencies	Registered Services	List of registered services
	Used Services	List of services in use
	Export Packages	List of exported packages
	Import Packages	List of imported packages
	Fragments	List of bundle fragments
	Host Bundle	Name of host bundle
	Requiring Bundles	List of bundles that require this bundle
Headers	Bundle-Version	Version of the bundle
	Bundle	Multiple options

Installing a bundle

This section describes how you can use the SJF Web Console to install a bundle in the SuperJ Framework. This involves selecting the bundle by identifying its URL or file location.

To install a bundle:

1. On the SJF Web Console toolbar, click the **Bundles** button.
2. Click the **Installation** button on the left side of the Bundles page. The Bundle Installation window appears as shown in Figure 5-3.

Figure 5-3: Installing Screen bundle

3. Ensure that the **Automated** checkbox is selected. When this checkbox is selected, the **Location** field is automatically set to match the entry in the **URL** field or **File** field.
4. Identify the bundle by doing one of the following:
   - Select the **URL** option and enter the URL in the accompanying text field. The URL you enter appears in the **Location** field.
   - Select the **FILE** option, then then click the **Browse** button to find and select the bundle file. The file path appears in the **Location** field.
5. Click the **Install** button to complete the installation. A confirmation message is displayed.

continued...

Managing Bundles, continued

Updating a Bundle

To update a bundle using the SJF Web Console, use the following procedure:

1. On the SJF Web Console toolbar, click the **Bundles** button.

Figure 5-4: Bundle state management screen ("Update" button).

2. In the **Bundle List** on the left side of the Bundles page, click the bundle name that you want to update. Information about the bundle appears in the **Bundle Information** section of the page.
3. Identify the bundle by doing one of the following:
   - Select the **URL** option and enter the URL in the accompanying text field. The URL you enter appears in the **Location** field.
   - Select the **FILE** option, then then click the **Browse** button to find and select the bundle file. The file path appears in the **Location** field.
4. Click the **Update** button. A confirmation message is displayed.

Uninstalling a bundle

To uninstall a bundle, perform these steps:

1. On the SJF Web Console toolbar, click the **Bundles** button.
2. In the **Bundle List** on the left side of the Bundles page, click the bundle name that you want to uninstall. Information about the bundle appears in the **Bundle Information** section of the page.
3. Click the **Uninstall** button. A confirmation message is displayed.

Starting a bundle

To start a bundle, perform the following steps:

1. On the SJF Web Console toolbar, click the **Bundles** button.

continued...▶

Managing Bundles, continued

2. In the **Bundle List** on the left side of the Bundles page, click the bundle name that you want to start. Information about the bundle appears.

   Note: If the bundle was not previously started, its **State** will be **INSTALLED**. If the bundle was previously started and then stopped, its **State** will be RESOLVED.

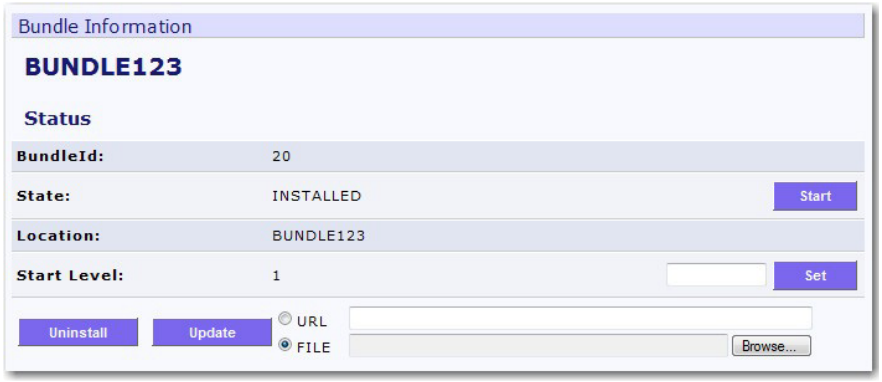


Figure 5-5: Bundle state management screen ("Start" button).

3. Click the **Start** button. The bundle **State** changes to ACTIVE.

Stopping a bundle

The following describes how to stop the bundle.

1. On the SJF Web Console toolbar, click the **Bundles** button.
2. In the **Bundle List** on the left side of the Bundles page, click the bundle name that you want to stop. Information about the bundle appears in the **Bundle Information** section of the page.

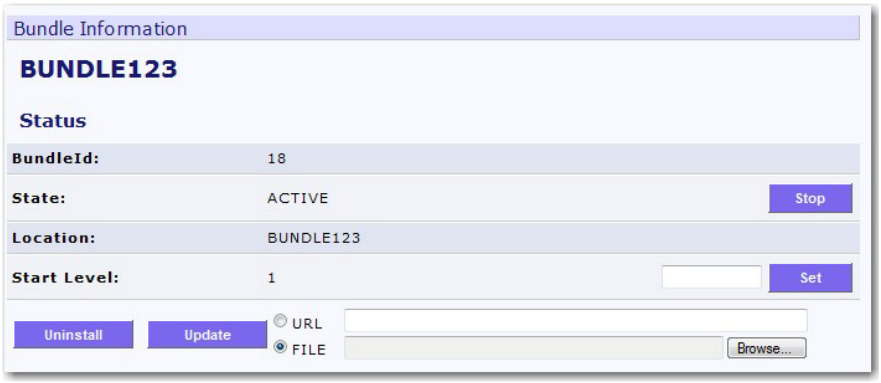


Figure 5-6: Bundle state management screen ("Stop" button)

3. Click the **Stop** button. The bundle **State** changes to RESOLVED.

continued...

Managing Bundles, continued

Setting the bundle start level

The bundle start level determines the priority given to each bundle when the framework is started. A lower start level signifies a higher priority.

> **⚠ CAUTION!**
>
> The framework start level must always be greater than or equal to the start level assigned to any bundle in the framework. If the start level of a bundle exceeds that of the framework, the framework may not start correctly.

To set the start level of a bundle, perform the following steps:

1. On the SJF Web Console toolbar, click the **Bundles** button.
2. In the **Bundle List** on the left side of the Bundles page, click the bundle name that you want to start. Information about the bundle appears in the **Bundle Information** section of the page.

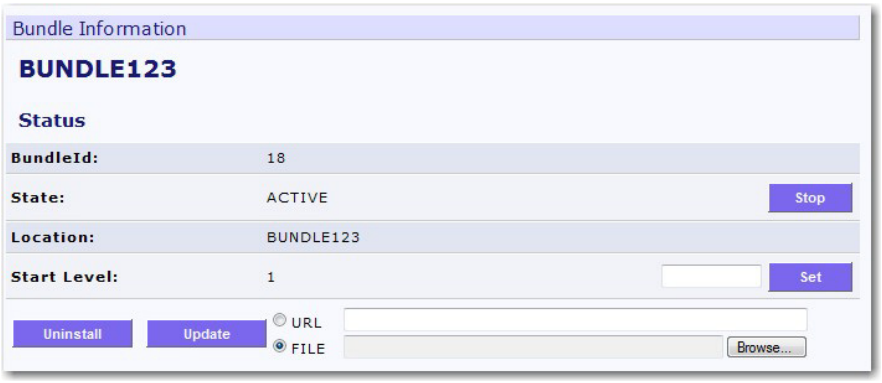


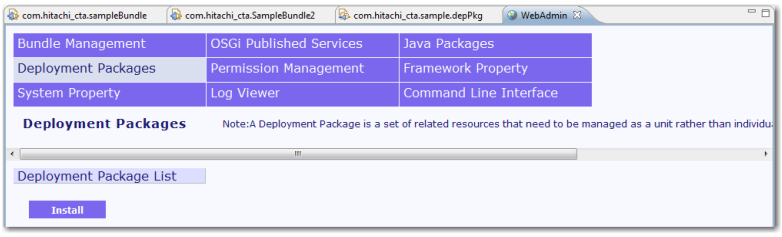
Figure 5–7: Screen bundle state management (start level setting)

3. Enter a number in the **Start Level** field.
4. Click the **Set** button to set the start level that you entered.

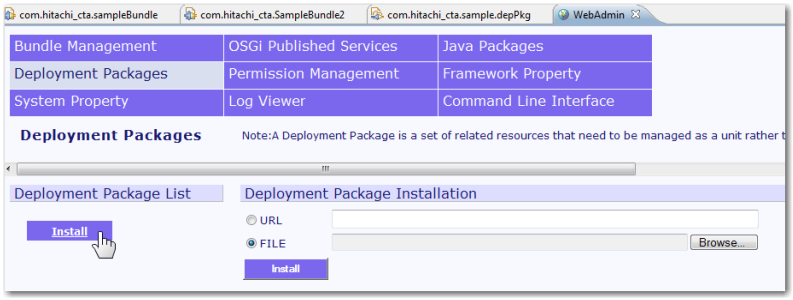
Using the WebAdmin tab to test deployment packages

Once the CPE Simulator is running and the WebAdmin host and port settings direct SJT as desired, you can begin testing deployment package options as follows:

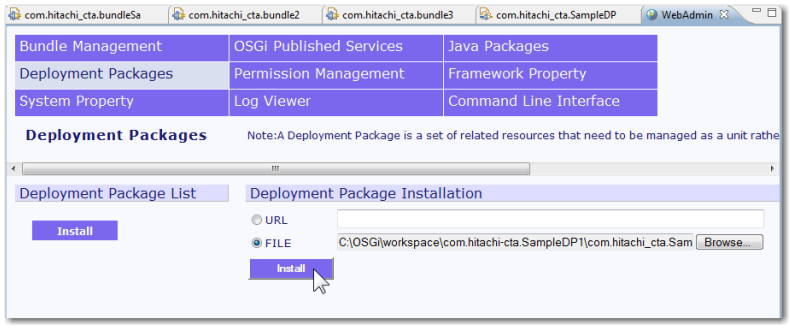
1. On the **WebAdmin** tab, click **Deployment Packages**, and then click **Install**.



2. The **Deployment Package Installation** area is displayed.



3. Type the URL or click **Browse** to locate the file of an exported deployment package that you want to test, and then click **Install** as shown here:



A message indicates whether Eclipse successfully installed the deployment package.

Managing Deployment Packages

This section describes how to view deployment package information, and how to install, uninstall, update, start, and stop deployment packages using the SJF Web Console.

Viewing deployment package information

To view details of a deployment package, perform the following steps:

1. On the toolbar, click **Deployment Packages**.
2. In the **Deployment Package List**, click a deployment package. Detailed information about the deployment package is displayed.

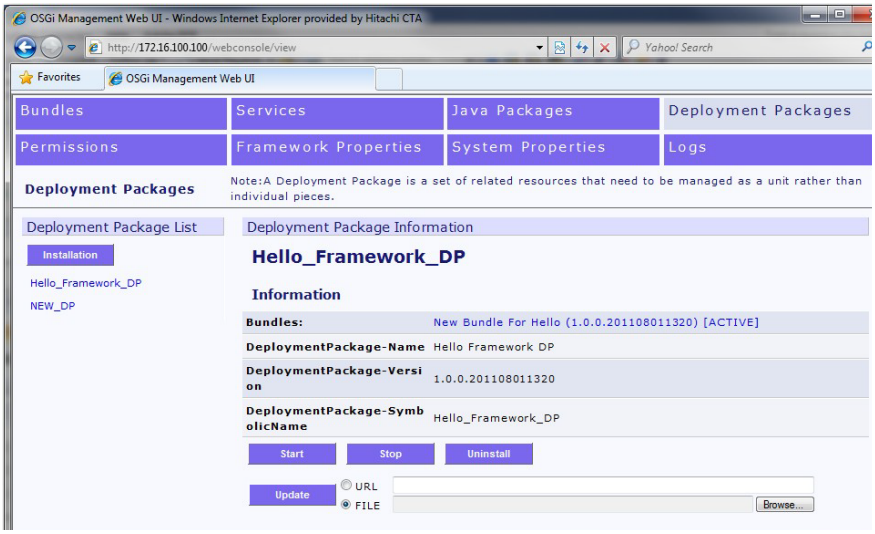


Figure 5-8: Deployment Package Information

Table 5-3: Deployment package information

Information	Description
Bundles	Bundles in the deployment package
DeploymentPackage-Symbolic Name	Symbolic name of the deployment package
DeploymentPackage-Version	Version number of the deployment package version
Start	Starts all bundles in the deployment package
Stop	Stops all bundles in the deployment package
Uninstall	Uninstalls the deployment package
Update	Updates the deployment packages

continued...▶

Managing Deployment Packages, continued

Installing the Deployment Package

You can use the SJF Web Console to install a deployment package in the SuperJ Framework.

To install a deployment package, use the following procedure:

1. On the SJF Web Console toolbar, click the **Deployment Packages** button.
2. Click the **Installation** button on the left side of the Deployment Packages page. The Deployment Package Installation window appears as shown in Figure 5-9.

Figure 5-9: Deployment package installation screen

3. Identify the deployment package by doing one of the following:
   - Select the **URL** option and enter the URL in the accompanying text field.
   - Select the **FILE** option, then then click the **Browse** button to find and select the bundle file.
4. Click the **Install** button to complete the installation. A confirmation message appears.

continued...

Managing Deployment Packages, continued

Updating Deployment Packages

To update a deployment package using the SJF Web Console, use the following procedure:

1. On the SJF Web Console toolbar, click the **Deployment Packages** button.
2. In the **Deployment Package List** on the left side of the page, click the deployment package that you want to update.
3. Identify the updated deployment package by doing one of the following:
   - Select the **URL** option and enter the URL in the accompanying text field.
   - Select the **FILE** option, then then click the **Browse** button to find and select the bundle file.

Figure 5-10: Deployment package status management

4. Click the **Update** button. A confirmation message appears.

Uninstalling a Deployment Package

The following describes how to uninstall a deployment package.

1. On the SJF Web Console toolbar, click the **Deployment Packages** button.
2. In the **Deployment Package List** on the left side of the page, click the deployment package that you want to uninstall.
3. Click the **Uninstall** button. A confirmation message appears.

continued...▶

Managing Deployment Packages, continued

Starting the deployment package

This procedure describes how to start a deployment package. When you start a deployment package, all the bundles in the package will be started.

1. On the SJF Web Console toolbar, click the **Deployment Packages** button.
2. In the **Deployment Package List** on the left side of the page, click the deployment package that you want to start.

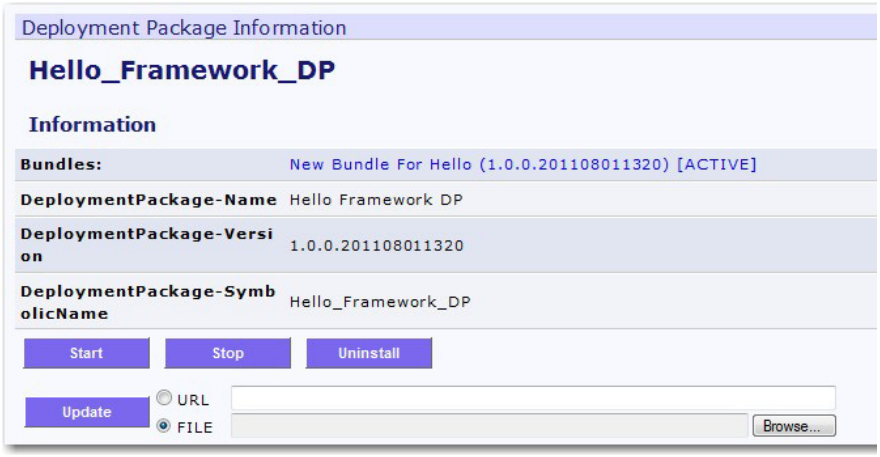


Figure 5-11: Deployment Package management screen ("Start" button).

3. Click the **Start** button. If the deployment package starts successfully, the status indication at the end of the **Bundles** field will change to **(ACTIVE)**.

Stopping the deployment package

The following describes how to stop the deployment package. When you stop a deployment package, all the bundles in the deployment package will be stopped.

1. On the SJF Web Console toolbar, click the **Deployment Packages** button.
2. In the **Deployment Package List** on the left side of the page, click the deployment package that you want to stop.
3. Click the **Stop** button. If the deployment package stops successfully, the status indication at the end of the **Bundles** field will change to **(RESOLVED)**.

The SJF Web Console has the following restriction in the Installation screen for bundles and deployment packages.

⚠CAUTION!

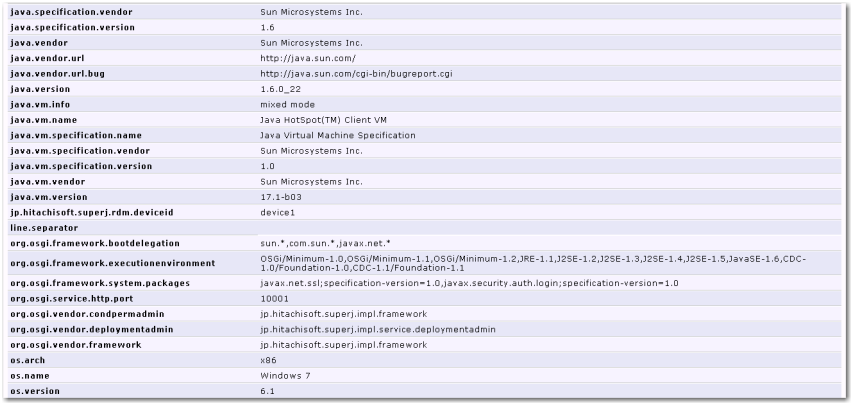
If you install a new deployment package containing a bundle with a symbolic name that matches the name of an installed bundle, the installation will fail.

Checking system properties

Once the CPE Simulator is running and the WebAdmin host and port settings direct SJT as desired, you can view options for system properties as follows:

1. On the **WebAdmin** tab, click **System Properties**. The System Information table displays information about the operating system and Java VM for the computer running Eclipse, as shown in the following figure.

   Note: The information in this table is display-only.

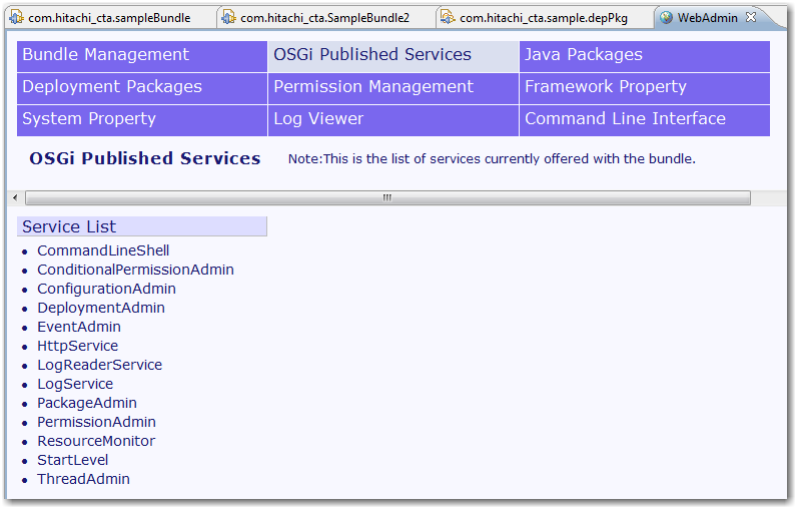


java.specification.vendor	Sun Microsystems Inc.
java.specification.version	1.6
java.vendor	Sun Microsystems Inc.
java.vendor.url	http://java.sun.com/
java.vendor.url.bug	http://java.sun.com/cgi-bin/bugreport.cgi
java.version	1.6.0_22
java.vm.info	mixed mode
java.vm.name	Java HotSpot(TM) Client VM
java.vm.specification.name	Java Virtual Machine Specification
java.vm.specification.vendor	Sun Microsystems Inc.
java.vm.specification.version	1.0
java.vm.vendor	Sun Microsystems Inc.
java.vm.version	17.1-b03
jp.hitachisoft.superj.rdm.deviceid	device1
line.separator	
org.osgi.framework.bootdelegation	sun.*,com.sun.*,javax.net.*
org.osgi.framework.executionenvironment	OSGi/Minimum-1.0,OSGi/Minimum-1.1,OSGi/Minimum-1.2,JRE-1.1,J2SE-1.2,J2SE-1.3,J2SE-1.4,J2SE-1.5,JavaSE-1.6,CDC-1.0/Foundation-1.0,CDC-1.1/Foundation-1.1
org.osgi.framework.system.packages	javax.net.ssl;specification-version=1.0,javax.security.auth.login;specification-version=1.0
org.osgi.service.http.port	10001
org.osgi.vendor.condpermadmin	jp.hitachisoft.superj.impl.framework
org.osgi.vendor.deploymentadmin	jp.hitachisoft.superj.impl.service.deploymentadmin
org.osgi.vendor.framework	jp.hitachisoft.superj.impl.framework
os.arch	x86
os.name	Windows 7
os.version	6.1

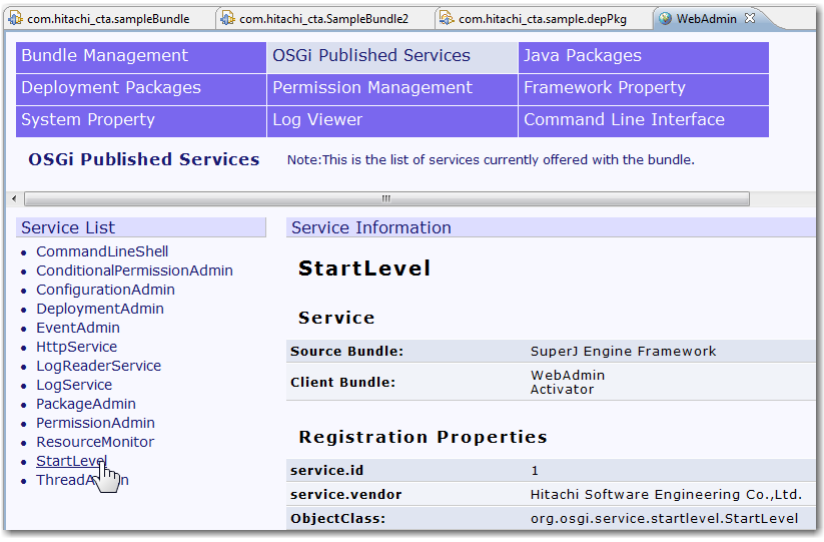
Testing OSGi Published Services

Once the CPE Simulator is running and the WebAdmin host and port settings direct SJT as desired, you can check which services are published (advertised) by a bundle as follows:

1. On the **WebAdmin** tab, click **OSGi Published Services**. The **Service List** displays the services that the selected bundle makes available for use by other bundles.



2. To find out the information published about a service, click the desired service in the **Service List** to view the **Service Information** published to other bundles for the selected bundle. In the following example, the StartLevel service was clicked:



Viewing Services

This section describes how to view service information using the SJF Web Console.

Displaying service information

To display detailed information about services, perform the following steps:

1. On the SJF Web Console toolbar, click the **Services** button.
2. In the **Service List** on the left side of the Services page, click the service that you are interested in. Information about the service appears in the **Service Information** section of the page.

Figure 5-12: Display of service information

Table 5-4 describes the Service Information fields.

Table 5-4: Service Information

Item		Description
Service	Source Bundle	The name of the service bundle
	Client Bundle	The name of the bundle using the service
Registration Properties	Registration Properties	Property Information

Viewing Java Packages

This section describes how to view Java package information and how to refresh packages using the SJF Web Console.

View package details

To view Java packages in the SuperJ Framework using the SJF Web Console, perform the following steps:

1. On the SJF Web Console toolbar, click the **Java Packages** button.
2. In the **Java Packages List** on the left side of the Java Packages page, click the package that you are interested in. Information about the package appears in the **Java Package Information** section of the page.

Figure 5-13: Java Package Information page

Table 5-5: Java Package information

Information	Description
Version	Package version
Exporters	Bundle or bundles that export the package
Importers	Bundle or bundles that import the package

Refreshing Java Packages

To refresh a Java package, perform the following steps:

1. On the SJF Web Console toolbar, click the **Java Packages** button.

2. In the **Java Packages List** on the left side of the Java Packages page, click the **Refresh all packages** button. A green confirmation message appears.

Figure 5-14: Refreshing Java packages

Setting and verifying permission management

The **Permission Management** feature allows an administrator to grant Java permissions to bundles on a framework.

The Conditional Permission Admin service is defined by the OSGi Service Platform Core Specification r4, extending the concepts of dynamic permission definition of the OSGi Permission Admin service by introducing conditional permissions. It is registered under the org.osgi.service.condpermadmin.ConditionalPermissionAdmin interface, and is available only if the framework is started with security.

A conditional permission is associated with a set of conditions and a set of permissions. The permissions of the conditional permission become granted when all associated conditions become satisfied. Whatever is possible with Default Permissions (PermissionAdmin) is also possible with conditional Permissions (ConditionalPermissionAdmin) applied to the bundle, while conditional Permissions can extend additional permissions when the criteria are met that are defined in the Permission Information statement.

Within the interface of PermissionAdmin, `DENY` is not defined, so a specified permission is only and always allowed. Thus, capability of the target keeps increasing whenever permissions are registered.

In WebAdmin, the following method is called when a permission is set:

```
ConditionalPermissionAdmin#setConditionalPermissionInfo(String name, ConditionInfo[] conditions, PermissionInfo[] permissions)
```

The arguments for this method are specified by WebAdmin as follows:

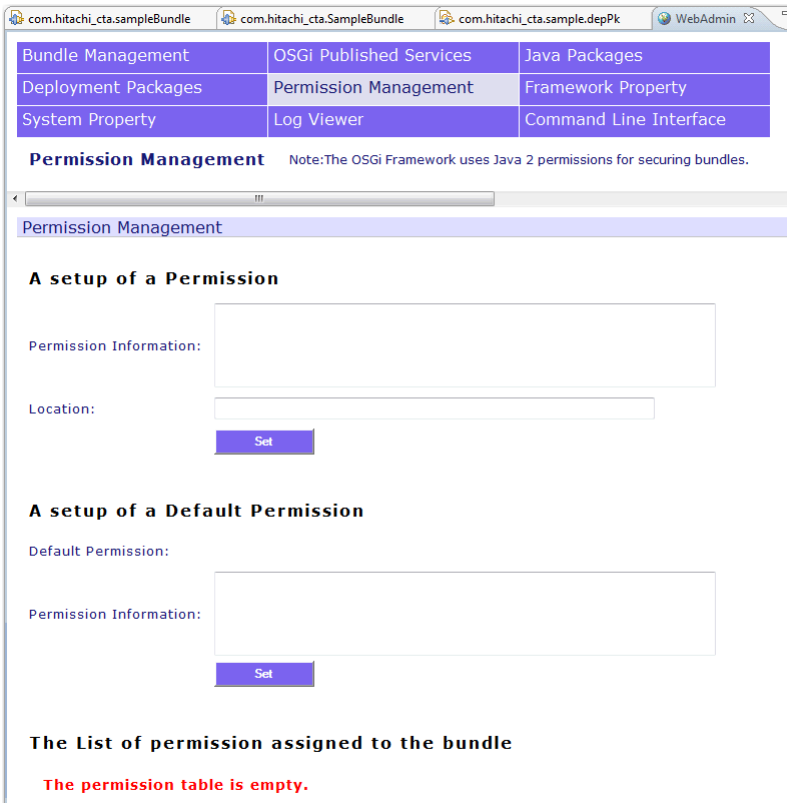
Argument	Specification
name	Populated with the string from the **Location** field
conditions	An array where the length is one and condition is created using `BundleLocationCondition`
permissions	An array where the length is one. The permission is created using `encodedPermission`.

continued...

Setting and verifying permission management, continued

Once the CPE Simulator is running and the WebAdmin host and port settings direct SJT as desired, you can set Java class permissions for the selected bundle(s) as follows:

1. On the **WebAdmin** tab, click **Permission Management**.



Item	Description per Area	
	A setup of a Permission	A setup of a Default Permission
Permission Information field	Type the desired string to define the conditional permission(s) for the bundle(s) at the path specified in the Location field.	Type the desired string to define the default permission(s) for the bundle.
Location field	Type the path of bundle(s) you want affected by the conditional permission(s).	N/A (implicitly "*")
Set button	Applies the conditional permission settings only to the bundle(s) in the specified location	Applies the default permission settings to all bundles
The List of permission assigned to the bundle	Displays all permissions (both conditional and default) currently assigned to the selected bundle. If the list is empty, it implicitly means that java.security.AllPermission is applied to all bundles.	

Note: If the bundle does not have the permission installed AllPermission can be used. If you do not setup any permission, please leave the permission information blank.

continued...

Setting and verifying permission management, continued

2. To set a conditional permission for a bundle:
   a. Type the string for the desired bundle's conditional permission in the **Permission Information** field under **A setup of a Permission** area. Type the specified string into the constructor PermissionInfo(java.lang.String encodedPermission) of Java class org.osgi.service.permissionadmin.PermissionInfo as an argument. If you want to set up multiple permissions, delimit the string of permissions with commas (,) or press the <space bar> or <Enter> key between the permissions.
   b. Type the path for the bundle in the **Location** field. If you use a relative path, it is understood as relative to the bundle's persistent storage area. These permissions are stored under the bundle location string. Using the bundle location allows the permissions to be set before a bundle is downloaded.
   c. Click **Set** (under **A setup of a Permission** area) to enact these changes to the bundle's permissions.
3. To set a default permission for a bundle:
   a. Type the string for the desired bundle's conditional permission in the **Permission Information** field under **A setup of a Default Permission** area. Type the specified string into the constructor PermissionInfo(java.lang.String encodedPermission) of Java class org.osgi.service.permissionadmin.PermissionInfo as an argument. If you want to set up multiple permissions, delimit the string of permissions with commas (,) or press the <space bar> or <Enter> key between the permissions.
   b. Click **Set** (under **A setup of a Default Permission** area) to enact these changes to the bundle's default permissions.
4. **The List of permission assigned to the bundle** is updated to display the permissions that you just set, in addition to any previously set conditional and/or default permissions.

Managing Permissions

This section describes how to view, set, remove, and initialize permissions in the SuperJ Framework using the SJF Web Console.

Viewing Permissions

This section describes how to use the SJF Web Console to to view the permissions set for a bundle in the SuperJ Framework.

- On the SJF Web Console toolbar, click the **Permissions** button. The Permissions page is displayed.

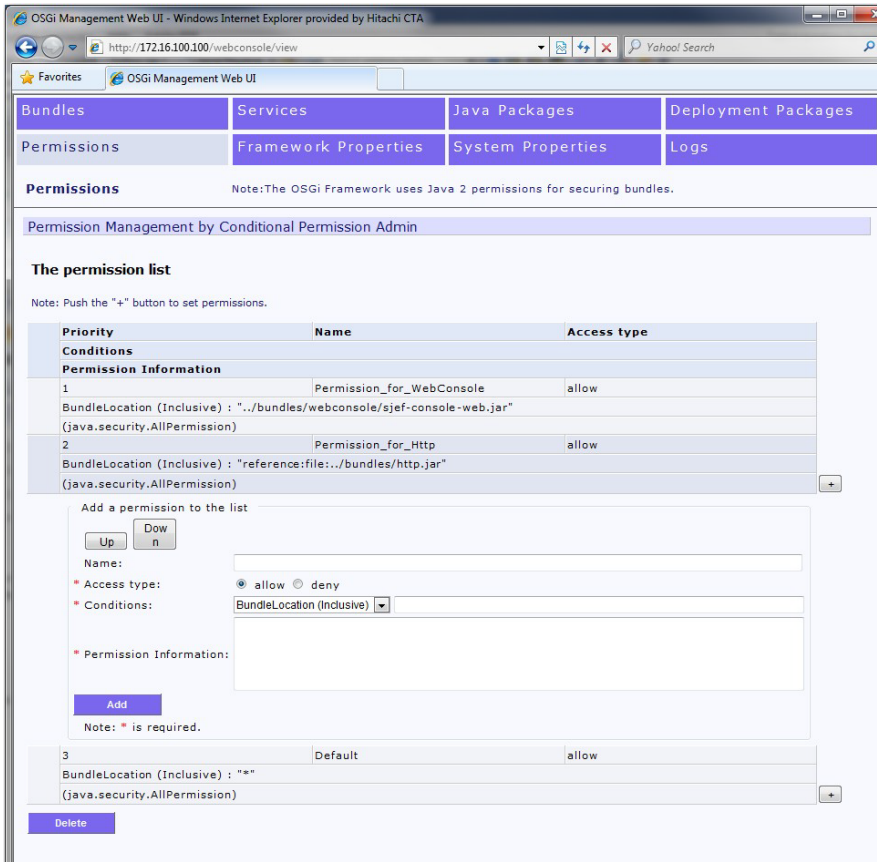


Figure 5-15: Permission information display

Table 5-6: Permissions information contents (Page 1 of 2)

Information	Information	Description
The permission list	Priority	Priority
	Name	Permission Name
	Access type	Type of access restrictions
	Conditions	Permission for the bundle Conditions
	Permission Information	Permission information for the bundle

Table 5-6: Permissions information contents (Page 2 of 2)

Information	Information	Description
Add a permission to the list	Name	Permission name
	Access type	Select one of the following options: • allow • deny
	Conditions	Conditions to apply to the permission
	Permission Information	Input field for the permissions
	Up	Increases the priority of the permission
	Down	Lowers the priority of the permission
	Add	Adds the permission as defined
	Delete	Removes the selected permission
Reset to the default setting	Reset	Reset to default permissions

Setting Permissions

To add a permission to the SuperJ Framework, perform the following steps. You can then apply this permission to one or more bundles.

1. On the SJF Web Console toolbar, click the **Permissions** button.
2. Under **Add a permission to the list,** click the **Up** button or the **Down** button to set the permission priority.
3. In the **Name** field, enter the name of the permission. If you leave this field blank, the name will be assigned automatically.
4. For **Access type**, select **Allow** or **Deny**.
5. For **Conditions**, use the drop-down box and the character strings field to specify the conditions under which this permission should be applied.
6. In the **Permission Information** field, enter the character strings to be used as an argument of the class constructor PermissionInfo (java.lang.String encodedPermission).
   - If you specify more than one permission, use a comma, space, or return.
   - If you specify AdminPermission and PackagePermission for example, enter the following text to the Permission Information text area.
     - Note that AllPermission will be set to the installed bundles with no permissions set when a permission is set.

- (org.osgi.framework.AdminPermission)
- (org.osgi.framework.PackagePermission "javax.servlet." "PackagePermission.EXPORT")

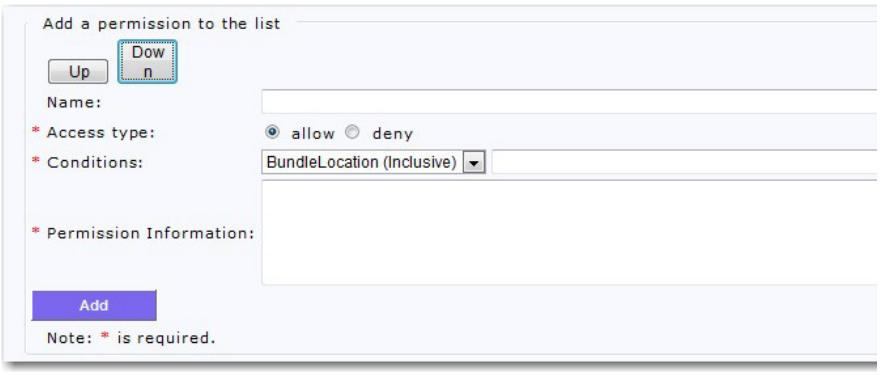


Figure 5-16: Permission Settings screen

7. Click the **Add** button to add the permission. A confirmation message is displayed.

Removing Permissions

To delete one or more permissions from the framework.

1. On the SJF Web Console toolbar, click the **Permissions** button.

2. In the **Permission List**, select the checkboxes for the permission(s) that you want to delete.

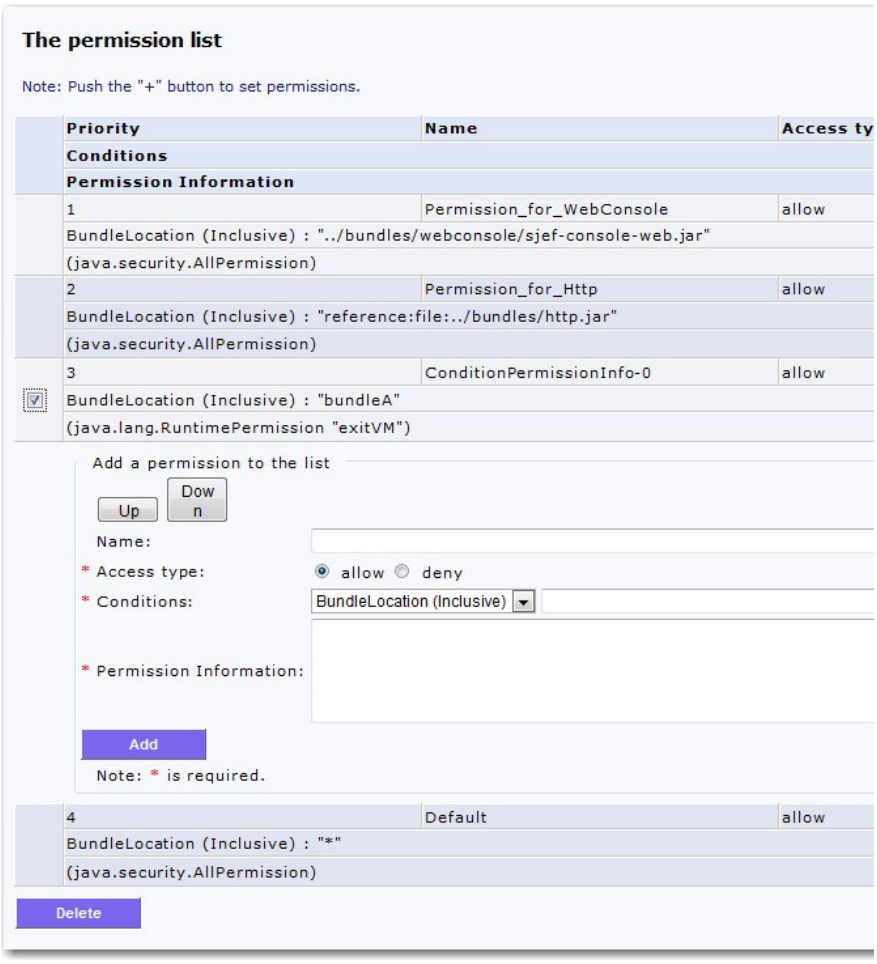


Figure 5–17: Remove permission

3. Click the **Delete** button. The permissions you selected are removed from the **Permission List**, and a confirmation message is displayed.

Initializing Permissions

The following procedure describes how to reset the SuperJ Framework permissions to the minimum permissions needed to run the SJF Web Console.

1. On the SJF Web Console toolbar, click the **Permissions** button.
2. Click the **Reset** button located at the bottom of the screen. A confirmation message is displayed.

3. Click the confirmation messge to reset the permissions. The default permissions are shown in the **Permission List**.

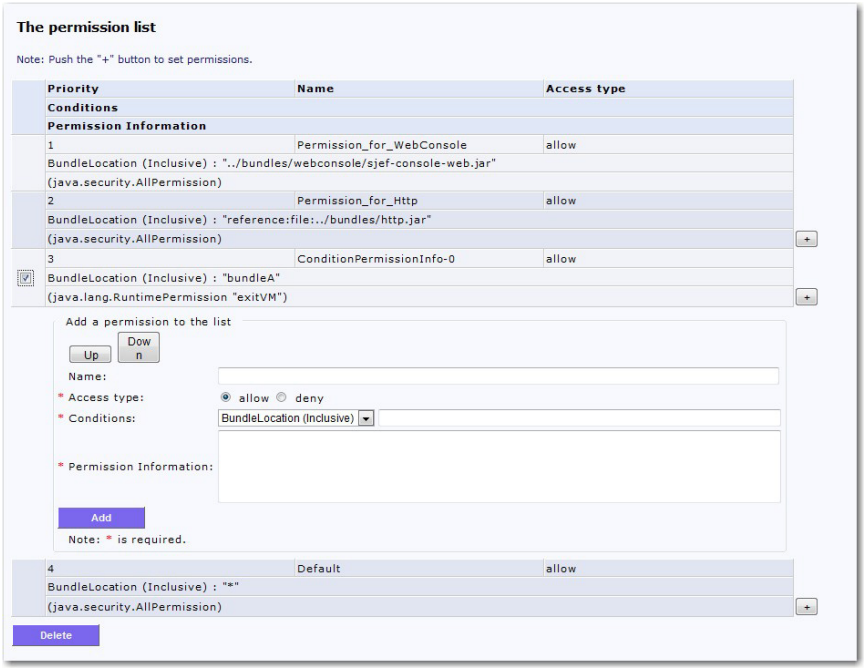


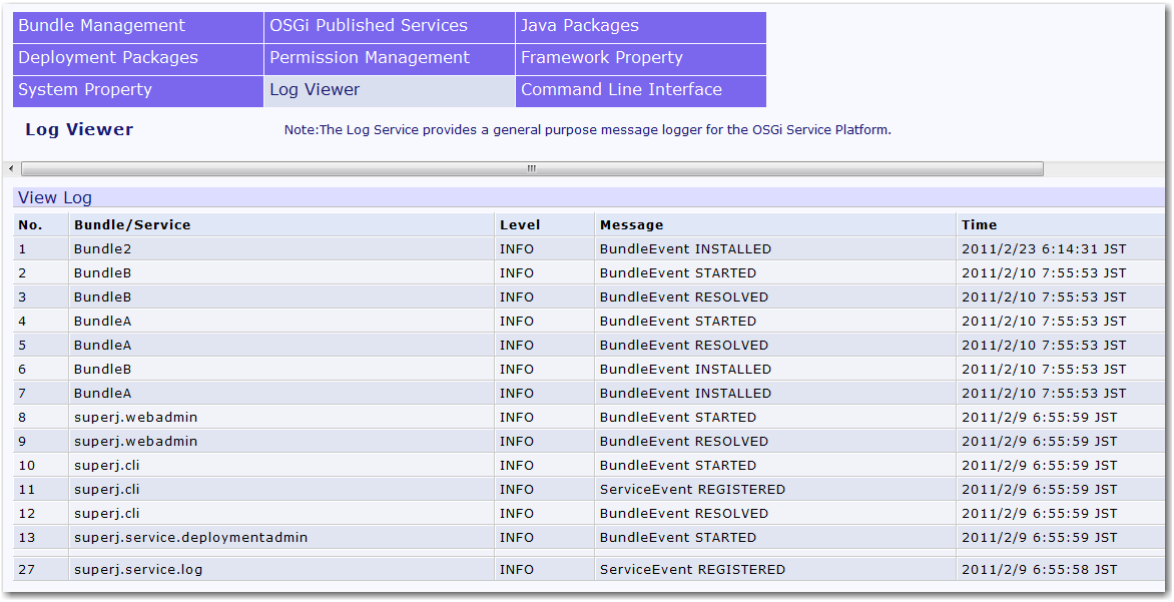
Figure 5–18: Permission initialization

Using the Log Viewer

Note: Content for this section is being developed.

Once the CPE Simulator is running and the WebAdmin host and port settings direct SJT as desired, you can view events that have affected a bundle in the Log Viewer as follows:

1. On the **WebAdmin** tab, click **Log Viewer.** The *View Log* pane opens:



No.	Bundle/Service	Level	Message	Time
1	Bundle2	INFO	BundleEvent INSTALLED	2011/2/23 6:14:31 JST
2	BundleB	INFO	BundleEvent STARTED	2011/2/10 7:55:53 JST
3	BundleB	INFO	BundleEvent RESOLVED	2011/2/10 7:55:53 JST
4	BundleA	INFO	BundleEvent STARTED	2011/2/10 7:55:53 JST
5	BundleA	INFO	BundleEvent RESOLVED	2011/2/10 7:55:53 JST
6	BundleB	INFO	BundleEvent INSTALLED	2011/2/10 7:55:53 JST
7	BundleA	INFO	BundleEvent INSTALLED	2011/2/10 7:55:53 JST
8	superj.webadmin	INFO	BundleEvent STARTED	2011/2/9 6:55:59 JST
9	superj.webadmin	INFO	BundleEvent RESOLVED	2011/2/9 6:55:59 JST
10	superj.cli	INFO	BundleEvent STARTED	2011/2/9 6:55:59 JST
11	superj.cli	INFO	ServiceEvent REGISTERED	2011/2/9 6:55:59 JST
12	superj.cli	INFO	BundleEvent RESOLVED	2011/2/9 6:55:59 JST
13	superj.service.deploymentadmin	INFO	BundleEvent STARTED	2011/2/9 6:55:59 JST
27	superj.service.log	INFO	ServiceEvent REGISTERED	2011/2/9 6:55:58 JST

Items listed in the **Bundle/Service** column are hyperlinked to the correlating installed bundle/service. The same information displays for the bundle as when you click the bundle from the list of installed bundles as described in “Using the WebAdmin tab with the CPE Simulator to test bundles” on page 5-8.

Note: Time is recorded in Japan Standard Time (JST), which is UTC/GMT +9 hours.

Viewing logs

To view logs, click the **Logs** button. Detailed information about the latest 20 logs is displayed.

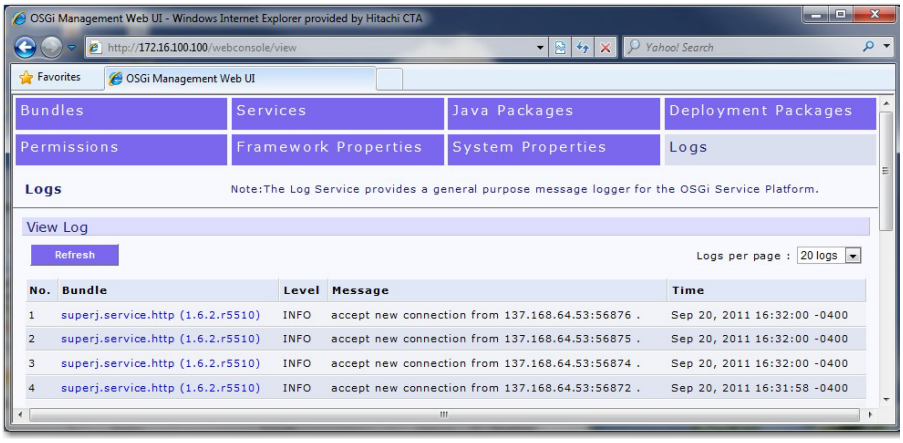


Figure 5-19: Log information screen

Table 5-7 describes the items in the Log Information screen.

Table 5-7: Content of log information

Information		Description
Log items	No.	Sequence number of the acquired log
	Bundle	Name of the bundle that output the log
	Level	Log level
	Message	Message
	Time	Time recorded in the log
Refresh		Obtains and displays the latest log information.
Logs per page		Sets the maximum number of log entries to retrieve and display

Changing the number of displayed logs

By default, when you view log infomation in the SJF Web Console, the last 20 logs are displayed. Use this procedure to change the number of logs displayed.

1. On the SJF Web Console toolbar, click the **Logs** button. Detailed information about the latest 20 logs is displayed.
2. Select the maximum number of displayed logs from the **Logs per page** drop-down.
3. Press the **Refresh** button.
4. The log list is updated. The number of logs displayed is now reset to the number you entered.

Viewing Java package information

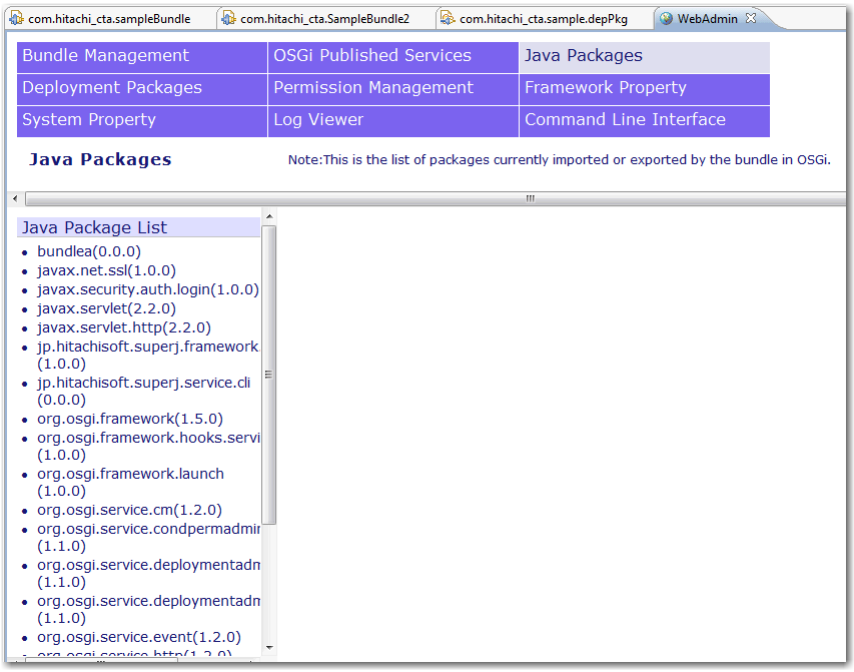
Note: Content for this section is being developed.

Once the CPE Simulator is running and the WebAdmin host and port settings direct SJT as desired, you can view and/or change the start level for a framework or a bundle as follows:

continued...▶

Viewing Java package information, continued

1. On the **WebAdmin** tab, click **Java Packages**. The **Java Packages List** displays the current list of packages imported or exported to OSGi by the selected bundle:



2. Click a package from the **Java Package List** to display its **Java Package Information**:

The following rules apply to Java package information in this window:

- If a package is imported by some bundles, the package is not deleted even if the bundle exporting the package is uninstalled or updated.
- If nobody imports a package, the package is refreshed when the bundle exporting the package is uninstalled or updated.
- Refreshing packages can be called with the `PackageAdmin#refreshPackages()` method. If this method is called, exported packages used by nobody are deleted.

The **Refresh** button in the **Java Packages** feature of WebAdmin corresponds to the `refreshPackages()` method.

Setting framework properties

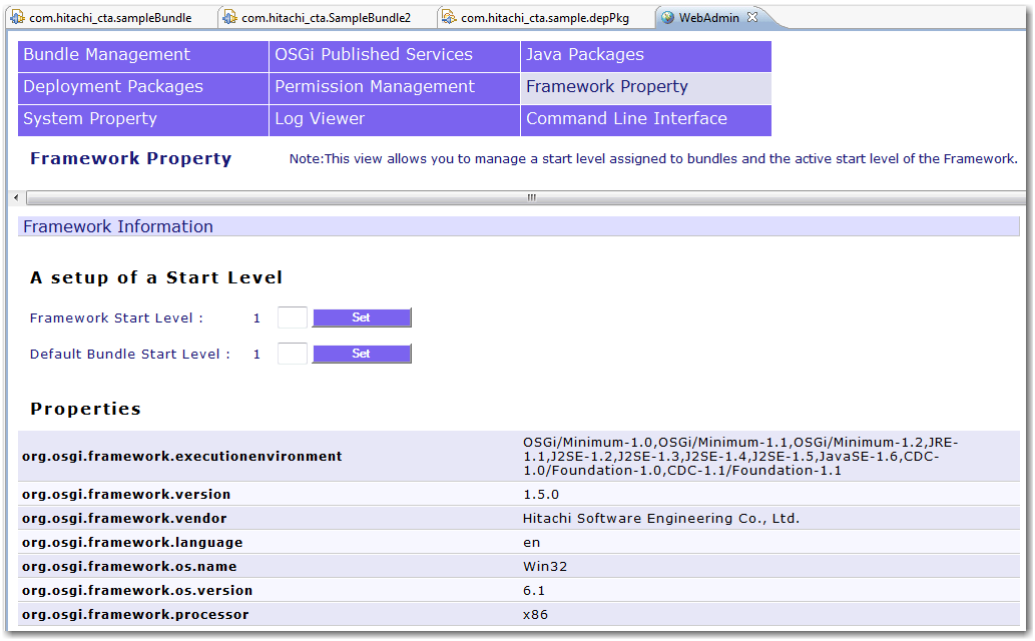
Note: Content for this section is being developed.

Use the bundle start level to:

- Control the order in which the bundle is started in the application.
- Modify the active start level of a bundle.
- Assign a specific start level to a bundle.
- Set the initial start level for newly installed bundles.

This section is about how to show the start level and property of the framework, default start level of the bundle. Once the CPE Simulator is running and the WebAdmin host and port settings direct SJT as desired, you can view and/or change the start level for a framework or a bundle as follows:

1. On the **WebAdmin** tab, click **Framework Property**. The **Framework Information** pane opens:



Item		Description
A setup of a Start Level	Framework Start Level	Setup of Framework start level
	Default Bundle Start Level	Setup of default bundle start level
Properties		Properties of framework

2. To change the start level for a framework, type the desired framework start level in the **Framework Start Level** field, and then click the adjacent **Set** button to enact your changes.

continued...

Setting framework properties, continued

3. To change the start level for a bundle, type the desired bundle start level in the **Default Bundle Start Level** field, and then click the adjacent **Set** button to enact your changes.

Managing Framework Properties

This section describes how you can view the SuperJ Framework properties using the SJF Web Console. These properties include the framework start level and bundle start level settings.

⚠CAUTION!

The framework start level must always be greater than or equal to the start level assigned to any bundle in the framework. If the start level of a bundle exceeds that of the framework, the framework may not start correctly.

Viewing framework properties

1. On the SJF Web Console toolbar, click the Frameçwork Properties button.

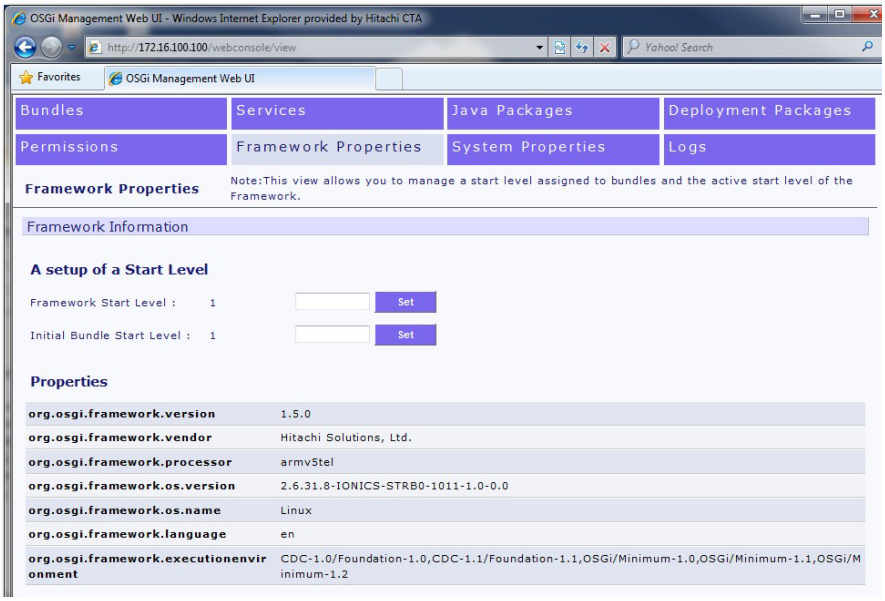


Figure 5–20: Framework property information screen

Table 5–8: Content of framework property information

Information	Information	Description
A setup of a Start Level	Framework Start Level	Sets the start level of the framework
	Initial Bundle Start Level	Sets the default start level applied to a bundle when added to the framework
Properties		Framework properties

continued...▶

Managing Framework Properties, continued

Setting the start level of the framework

To set the start level for the framework, perform the following prcedure:

1. On the SJF Web Console toolbar, click the **Framework Properties** button.

Figure 5-21: Framework start level setting screen

2. In the **Framework Start Level** field, enter a number.
3. Click the **Set** button. The start level changes to the value that you set.

Setting a default start level for bundles

This section describes how to set the default start level for bundles.

1. On the SJF Web Console toolbar, click the **Framework Properties** button.
2. In the **Framework Start level** field, enter a number.
3. Click the **Set** button. The default start level changes to the value set.

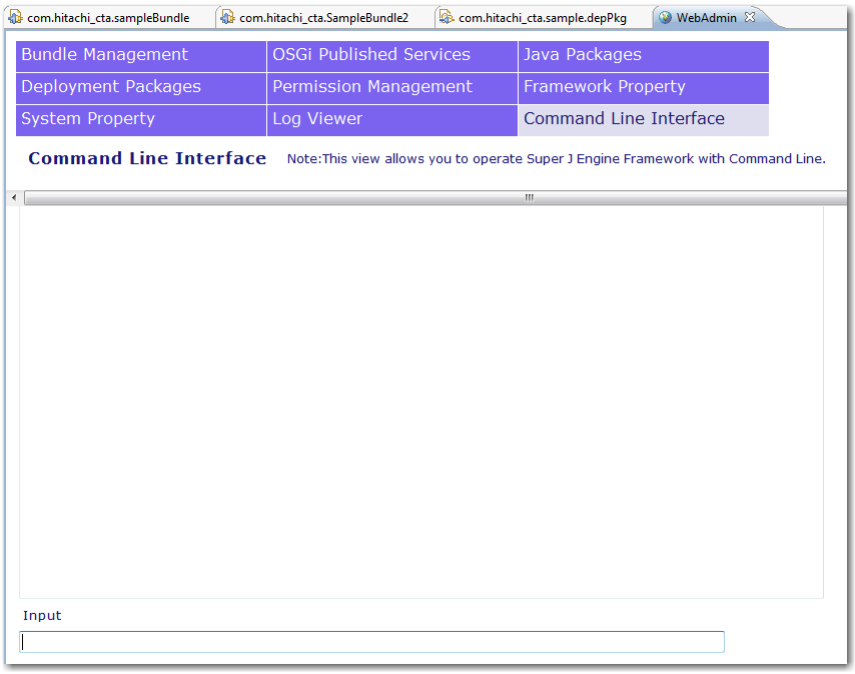
Figure 5-22: The default bundle start level setting screen

Using the Command Line Interface

Note: Content for this section is being developed.

Once the CPE Simulator is running and the WebAdmin host and port settings direct SJT as desired, you can issue commands directly to the bundles running within the CPE Simulator or on the targeted CPE (if webadmin.jar and cli.jar are installed and running with SJF and SJMA) without opening a terminal window or command prompt, as follows:

1. On the **WebAdmin** tab, click **Command Line Interface**:



2. You can use the **Input** field to interact with the SJF by issuing commands, including but not limited to:
   - help
   - bundleStat
   - version
   - headers

6 Creating a Provisioning Archive

This chapter describes how to create a provisioning archive (PAR). PAR files contain the packaging and information required for deploying a service in Hitachi-CTA's SuperJ Deployment Management System (SJDMS).

Note: Content for this section is being developed.

Contents

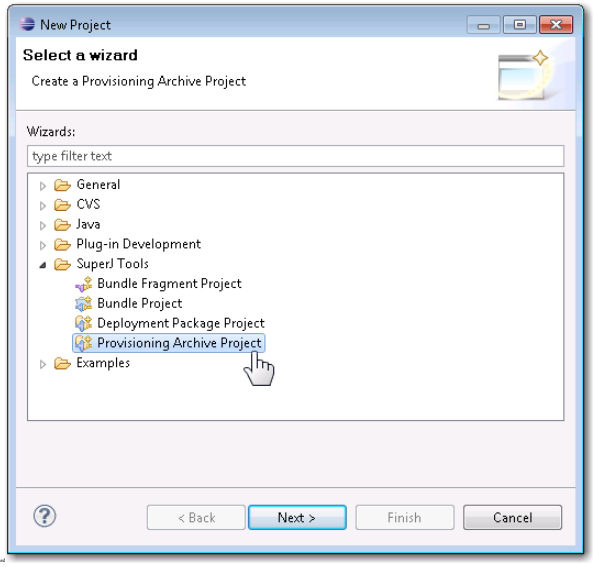
Topics covered in this document include: **Page**

Note: The first topic begins on the following page.

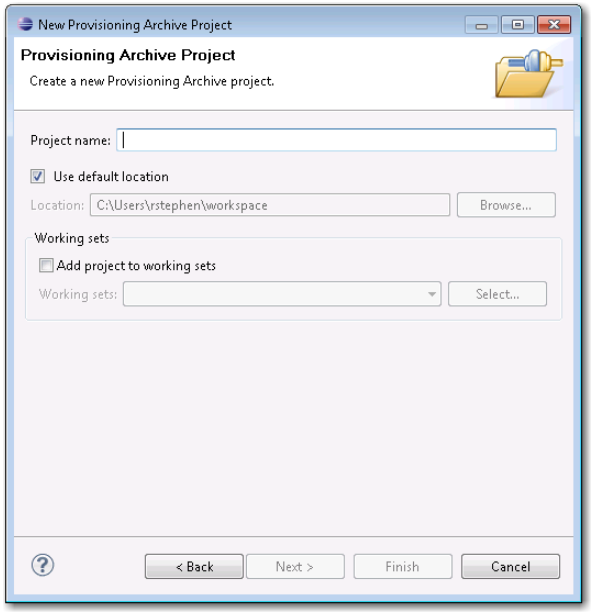
Creating a Provisioning Archive Project

To create a provisioning archive (PAR) file, you must first create a provisioning archive project using the following procedure.

1. From the main menu, select **File > New > Project.** The *New Project* window opens.
2. Under **SuperJ Tools**, select **Provisioning Archive Project**, and then click **Next.**



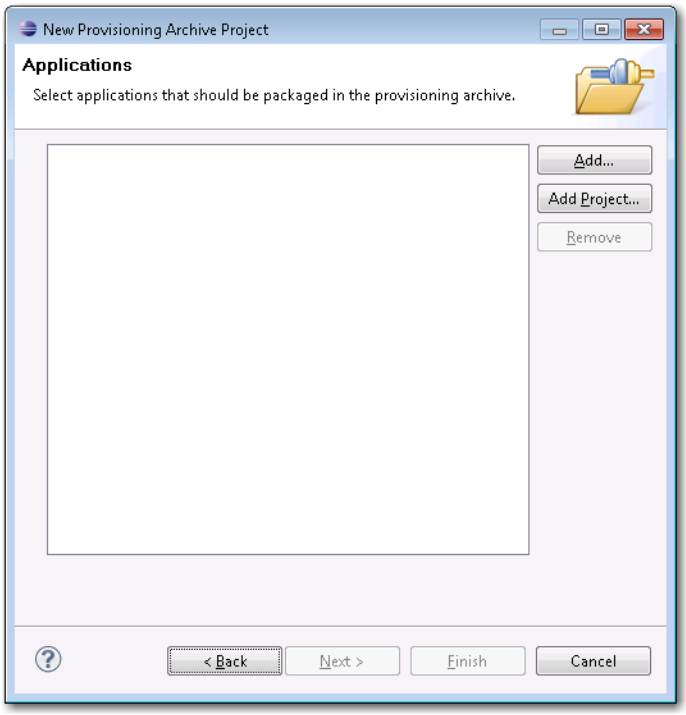
The *Provisioning Archive Project* page opens in the *New Provisioning Archive Project* window.



continued...

Creating a Provisioning Archive Project, continued

3. Complete the *Provisioning Archive Project* page as follows:
   a. Type a **Project name** for the provisioning archive project.
   b. If you do not want to use the default path for this PAR project, **Browse** to the desired path for the PAR project.
   c. If the PAR file(s) in this project should be part of Java working sets:
      - Click the check box for Add project to working sets, and then choose working sets from the list, or
      - Click **Select** to choose the desired **Working sets**, and then click **OK**.
4. On the *Provisioning Archive Project* page click **Next**. The *Applications* page opens in the *New Provisioning Archive Project* window.



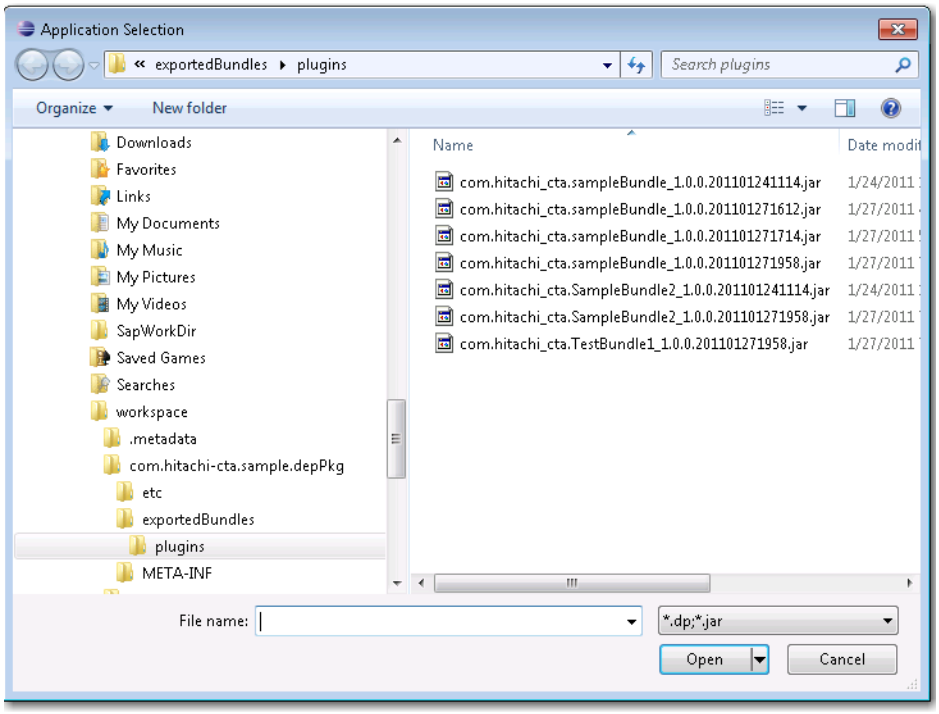
continued...

Creating a Provisioning Archive Project, continued

5. Do one or more of the following, as desired:

   - To add exported deployment packages to the PAR project:

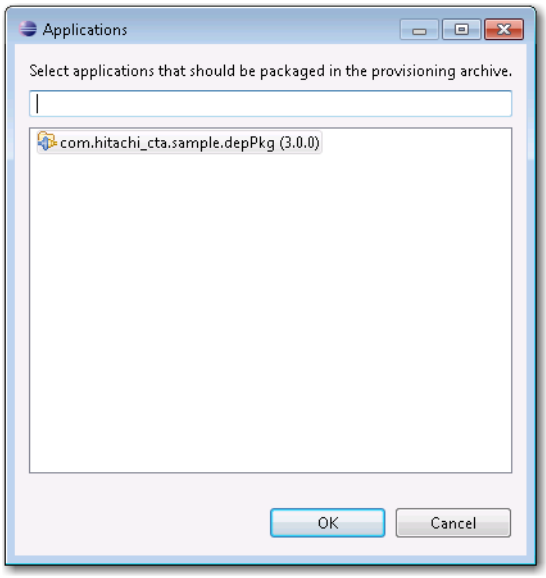
     - Click **Add** to open the *Application Selection* window.



     - Navigate to and select the desired exported deployment package (DP) file, and then click **Open** to add it to the PAR file that you are creating.

   - To add an unexported deployment package project to the PAR file:

     - Click **Add Project** to open the *Applications* window.



     - Select the desired .depPkg file(s), and then click **OK**.

continued...

Creating a Provisioning Archive Project, continued

6. Click **Finish**. The PAR project is created and displayed in the **Package Explorer** tree view, with the associated with the associated PAR project showing in the editor pane.

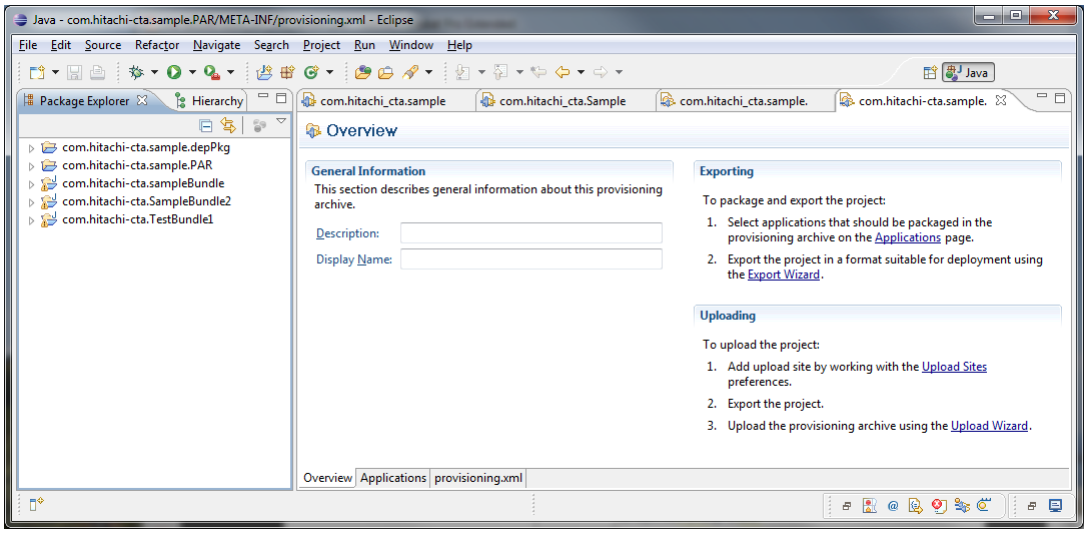


Figure 6–1: Overview tab in the PAR project editor

The PAR project editor contains the following tabs:

- Overview
- Applications
- provisioning.xml

Using the Provisioning Archive Editor

The provisioning archive editor opens automatically when you create a PAR project. To manually start the editor, in the Package Explorer tree view, select the desired provisioning archive project, expand **META-INF** folder, and then double-click on **provisioning.xml**.

You can use the Provisioning Archive Editor to:

- Add a deployment package (see page 6-10)
- Remove a deployment package (see page 6-11)
- Create a provisioning archive file (see page 6-13)
- Edit the provisioning descriptor file (provisioning.xml) (page 6-12)

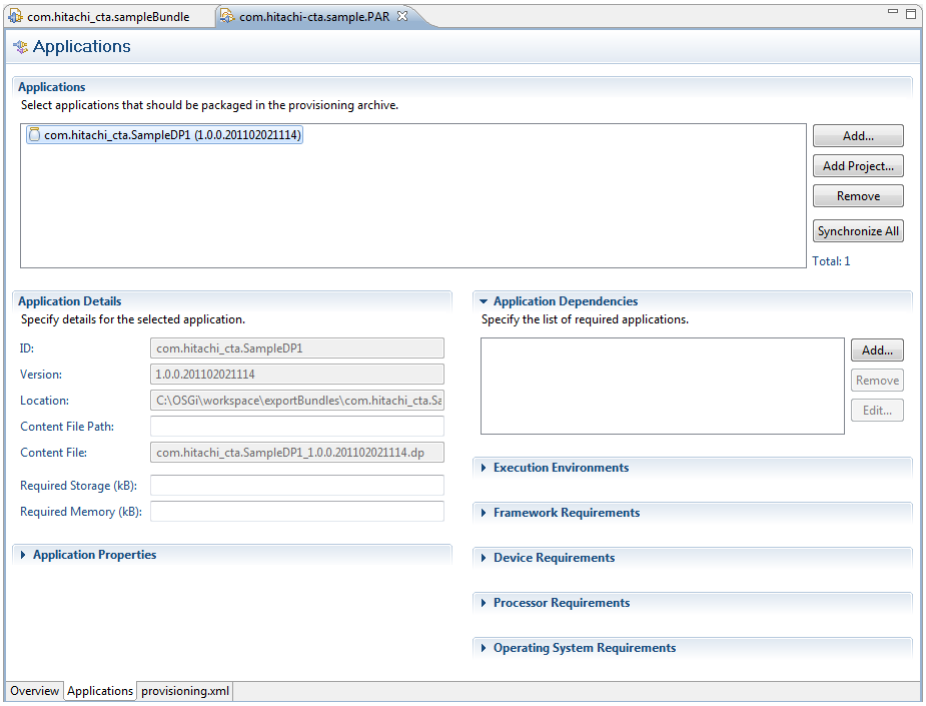
Working with the Overview tab for a PAR project

You can edit information about the provisioning archive from the **Overview** tab (see Figure 6-1 on page 6-5). The **Overview** tab has the following areas (panes):

Area	Description
General Information	Provides the following editable fields: • Description • Display Name
Exporting	Provides a process overview to package and export a PAR file, as well as links to the relevant tools for the process: • **Applications** – accesses the Applications tab in this window • **Export Wizard** – Export the provisioning archive (PAR) file as described in "Delivering a Provisioning Archive File" on page 6-13.
Uploading	Provides a process overview to upload the exported PAR project, as well as links to the relevant tools for the process: • Upload Sites • Upload Wizard

Working with the Applications tab for a PAR project

You can change the applications, their details, dependencies, and properties, execution environments, and other requirements on the **Applications** tab:



Note: Content for this section is being developed.

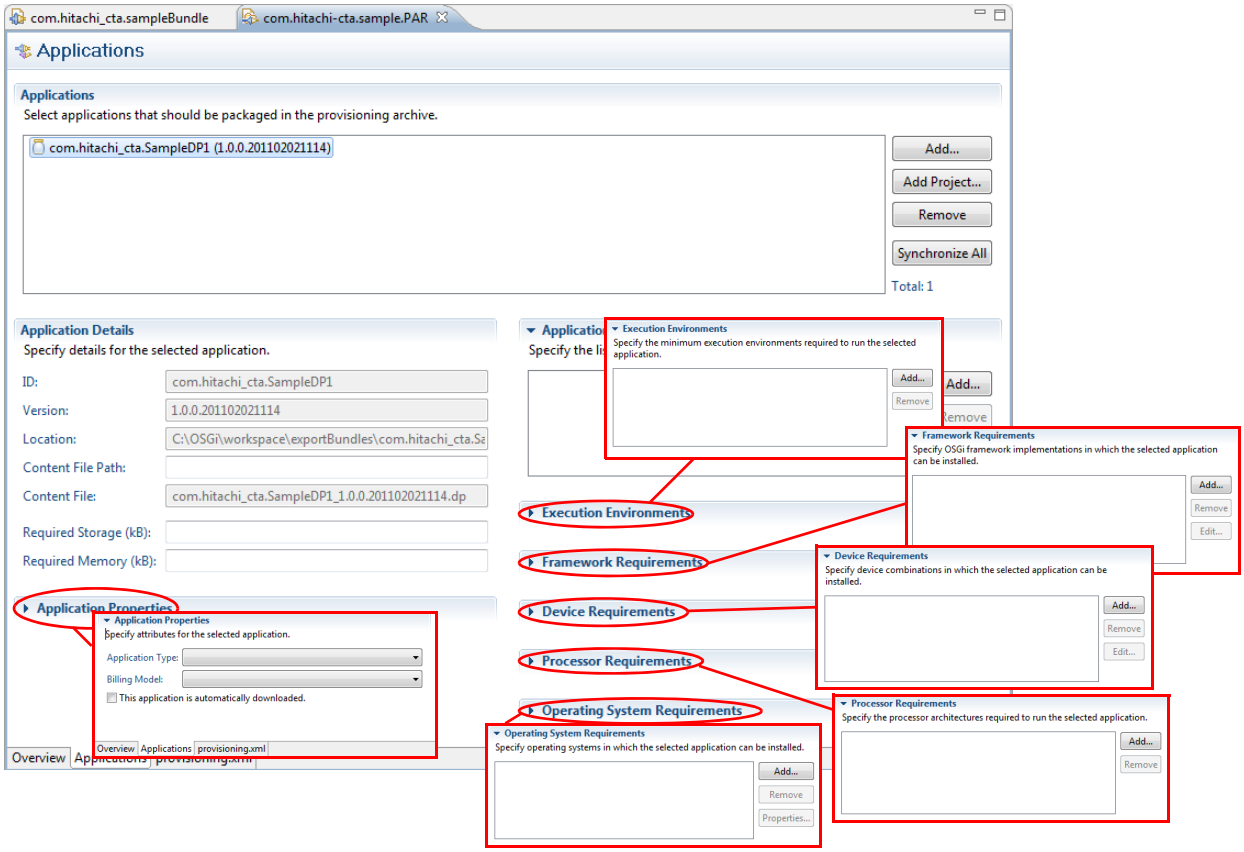
The **Applications** *tab* contains:

Area	Item	Description
Applications	list box	Displays the applications currently identified for packaging in the PAR.
	Add button	Allows you to select exported DP files to be included in the PAR. See “Adding a deployment package” on page 6-10.
	Add Project button	Allows you to select a deployment package project in the workspace to include in the archive target. See “Adding a deployment package project” on page 6-11.
	Remove button	See “Removing a deployment package” on page 6-11.
	Synchronize All button	See “Synchronizing applications in the PAR” on page 6-12.

Area	Item	Description
Application Details	**ID** field	
	Version field	
	Location field	
	Content File Path field	Identifies a path to store the application in the archive.
	Content File field	
	Required Storage (kB) field	Specifies the minimum amount of persistent storage required by the selected application.
	Required Memory (kB) field	Specifies the minimum amount of memory required by the selected application.
Application Dependencies	list box	Specifies the list of required applications.
	Add button	
	Remove button	
	Edit button	
Application Properties[a]	**Application Type** list	Specifies the attributes for the selected application.
	Billing Model list	
	This application is automatically downloaded check box	
Execution Environments[a]	list box	Specifies the minimum execution environments required to run the selected application.
	Add button	
	Remove button	
Framework Requirements[a]	list box	Specifies OSGi frameworks in which the selected application can be installed.
	Add button	
	Remove button	
	Edit button	
Device Requirements[a]	list box	Specifies device combinations in which the selected application can be installed.
	Add button	
	Remove button	
	Edit button	
Processor Requirements[a]	list box	Specifies the processor architectures required to run the selected application.
	Add button	
	Remove button	

Area	Item	Description
Operating System Requirements[a]	list box	Specifies operating system in which the selected application can be installed.
	Add button	
	Remove button	
	Properties button	

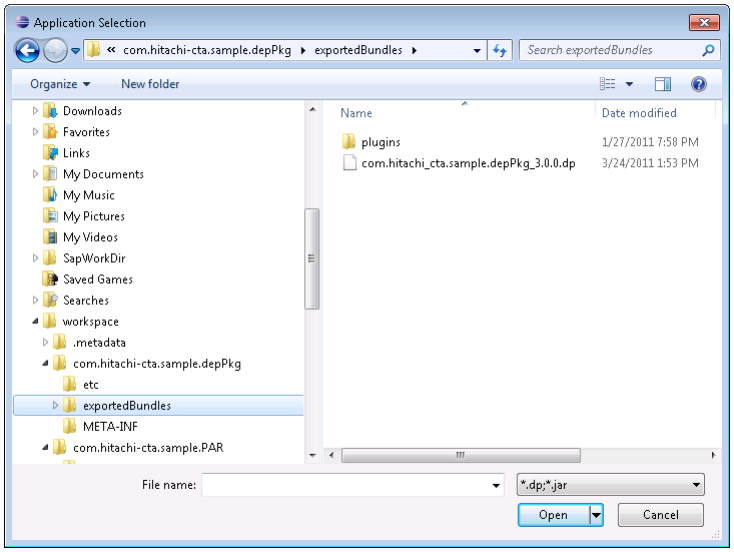
a. Click the arrowhead to the left of this heading to access its items.



Adding a deployment package

You can add a previously exported deployment package (DP) file to the Provisioning Archive using the Provisioning Archive Editor as follows:

1. Open the Provisioning Archive Editor as described in "Using the Provisioning Archive Editor" on page 6-6, and in the **Applications** area, click **Add**. The *Application Selection* window opens.

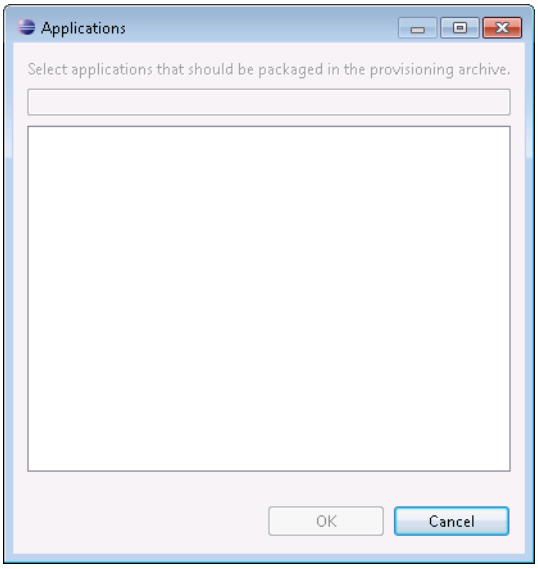


2. Navigate to the DP for the application that you want to add to the PAR file, select it, and then click **Open**. The **Applications** list box is updated to reflect the newly added application.
3. Click **Save** to keep the changes you have made to the PAR project.

Adding a deployment package project

You can add a previously exported deployment package (DP) project from your workspace to the Provisioning Archive using the Provisioning Archive Editor as follows:

1. Open the Provisioning Archive Editor as described in "Using the Provisioning Archive Editor" on page 6-6, and in the **Applications** area, click **Add Project**. The *Application* window opens.



2. Choose the desired DP projects from the drop list, and then click **OK**. The **Applications** list box is updated to reflect the newly added DP project.
3. Click **Save** to keep the changes you have made to the PAR project.

Removing a deployment package

You can remove a deployment package file from the Provisioning Archive using the Provisioning Archive Editor as follows:

1. Open the Provisioning Archive Editor as described in "Using the Provisioning Archive Editor" on page 6-6, and then select the **Applications** tab.
2. In the Applications list box, select the package that you no longer want to be in the archive target, and then click **Remove**.
3. The file you selected is removed from the deployment packages list.
4. Click **Save** to keep the changes you have made to the PAR project.

Synchronizing applications in the PAR

To synchronize applications in the PAR file:

1. Open the Provisioning Archive Editor as described in "Using the Provisioning Archive Editor" on page 6-6, and then select the **Applications** tab.
2. Click **Synchronize all**. The SJT reads the symbolic name and version from each deployment package to update the application information in the target archive.

Editing the provisioning descriptor

If you need to modify a deployment package — such as changing the content ID, version, text content of the .xml — you must edit the provisioning.xml file in the Provisioning Archive.

1. Open the Provisioning Archive Editor as described in "Using the Provisioning Archive Editor" on page 6-6.
2. Select the **provisioning.xml** tab.
3. Edit the XML file as required.
4. Click **Save**.

Delivering a Provisioning Archive File

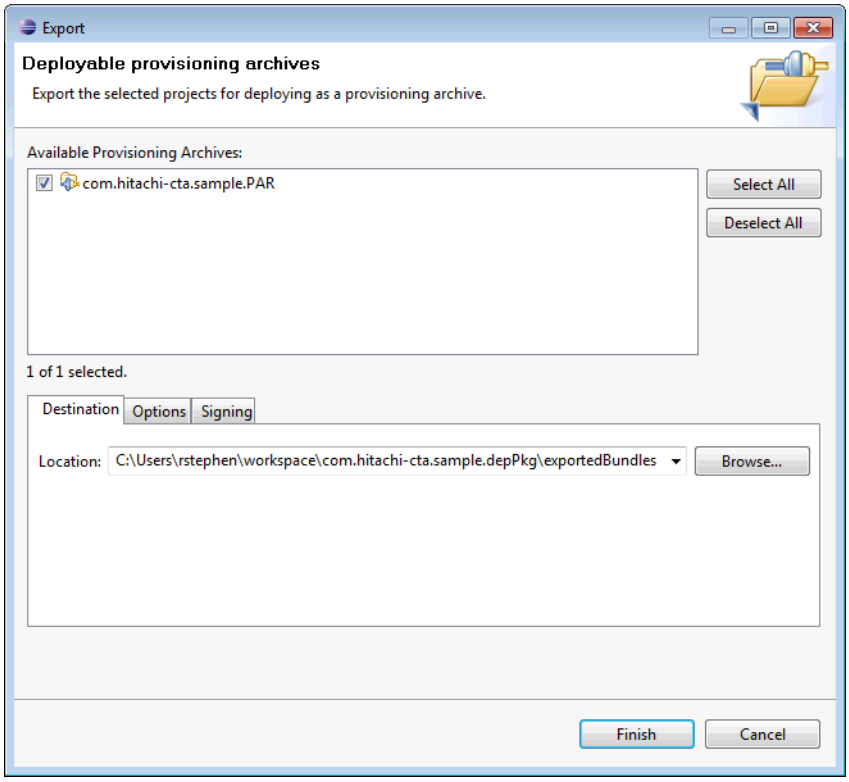
To deliver a PAR file, you must first export it and then upload it to the desired distribution point.

Note: Content for this section is being developed.

Exporting the PAR

To create a deliverable Provisioning Archive (PAR) file:

1. Open the Provisioning Archive Editor as described in "Using the Provisioning Archive Editor" on page 6-6, and then select the **Overview** tab.
2. In the Exporting area, click the **Export Wizard** link. The *Export* window opens.

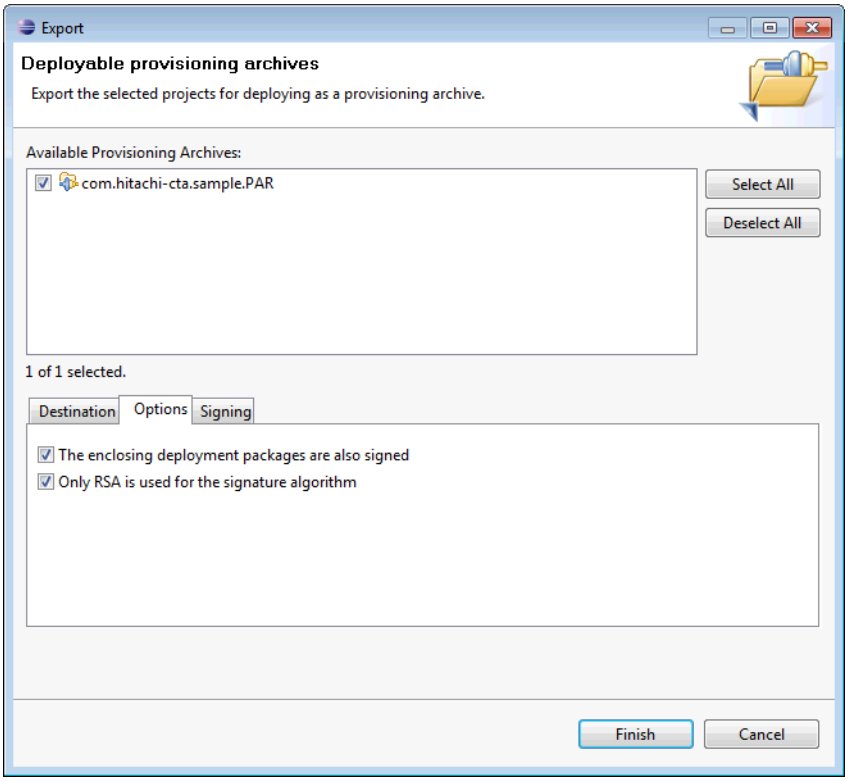


3. Type the file path for the new archive in the **Location** field on the **Destination** tab.

continued...

Exporting the PAR, continued

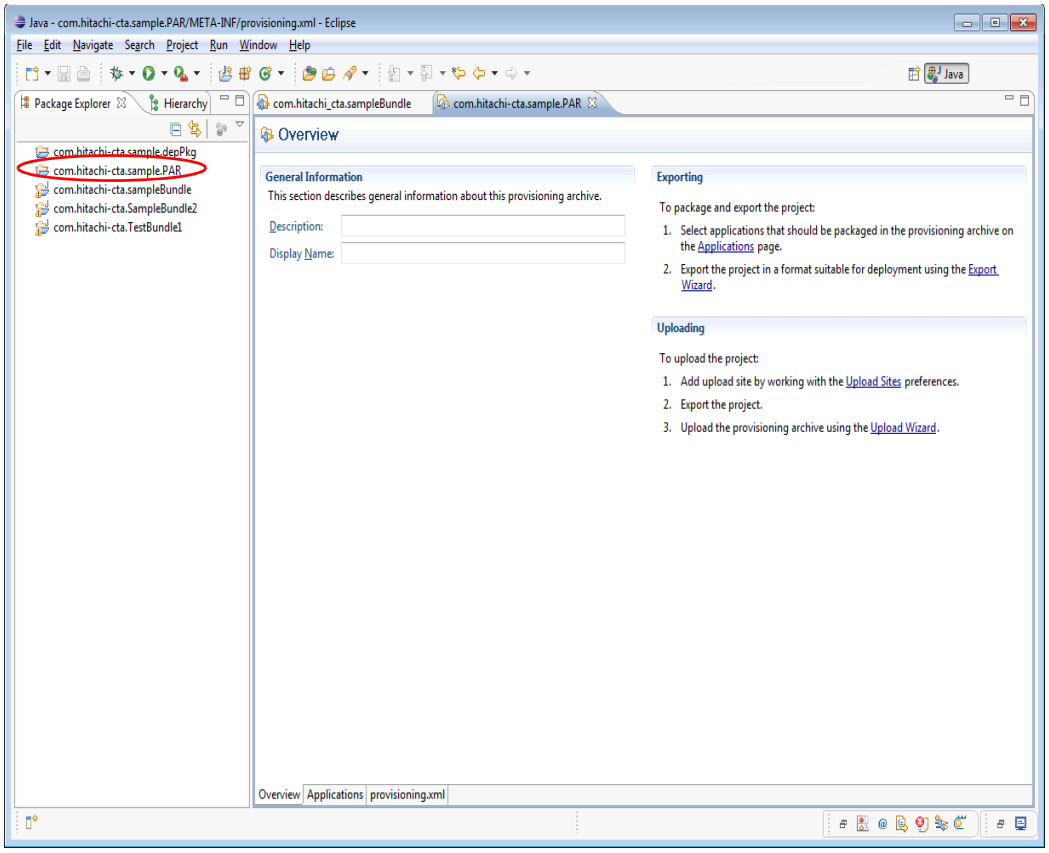
4. Select the **Options** tab and complete as desired:



continued...

Exporting the PAR, continued

5. Click **Finish.** The new PAR file now shows in the Package Explorer tree view.



Uploading PAR files

You can upload PAR files to the SJDMS server via SFTP or FTP as follows.

IMPORTANT!

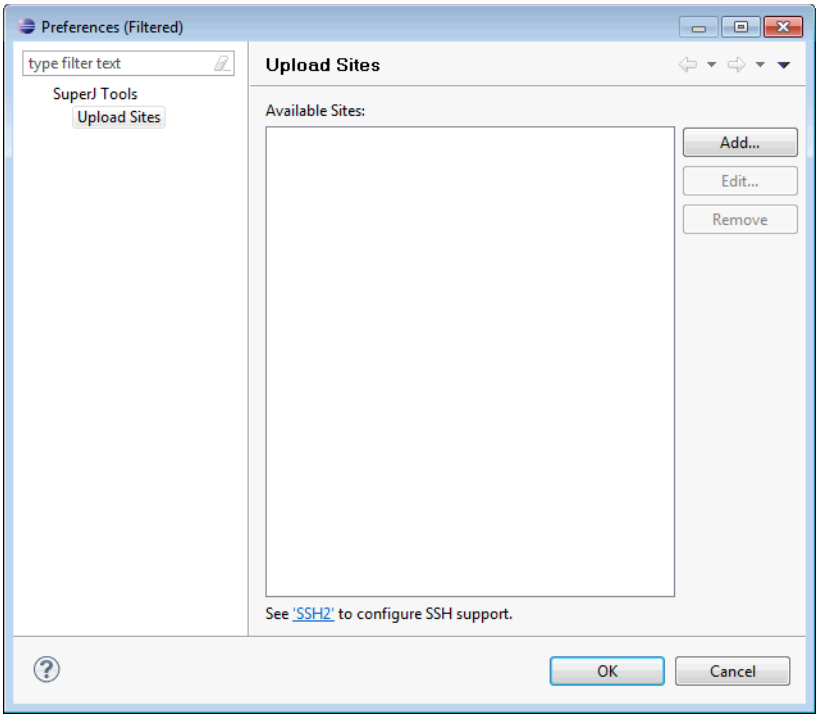
To use the FTP upload feature, you must install the Apache Commons Net bundle. See “Required software” on page 1-3 for details.

1. Open the Provisioning Archive Editor as described in “Using the Provisioning Archive Editor” on page 6-6, and then select the **Overview** tab.

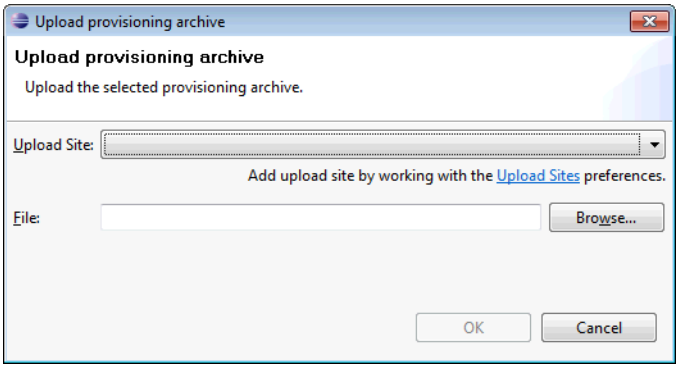
continued...

Uploading PAR files, continued

2. In the **Uploading** area, click the **Upload Sites** link. The *Preferences (filtered)* window opens with the *Upload pane*.



3. Make the desired changes, and then click **OK**.
4. Perform "Exporting the PAR" on page 6-13.
5. In the **Uploading** area, click the **Upload Wizard** link. The *Upload Provisioning Archive* window opens.



6. Choose an **Upload Site**, click **Browse** to navigate to the location of the exported **PAR** file, and then click **OK**.

GLOSSARY

Industry and Hitachi Terms

API (Application Programming Interface)

An interface implemented by a software program that enables it to interact with other software. An API facilitates interaction between applications, libraries, and operating systems to determine their vocabularies and calling conventions, and is used to access their services. An API can include specifications for routines, data structures, object classes, and communication protocols.

Application

Loosely defined, an application is a set of cooperating bundles that deliver a specific functionality to a user. From the functionality perspective, an application is a deployment package that contains one, or a set of cooperating, bundle(s) that deliver a specific functionality to the user. Adding hardware restrictions and dependencies to the application is a PAR file. Applications are created using Hitachi-CTA's SJT. Applications cannot be modified in the SJDMS repository; they can only be added or deleted. Applications can only be modified using the Hitachi-CTA's SJT. Even if an application has only one bundle, it would still be packaged as a deployment package in an application. An application may be part of a service or standalone.

Bundle

A bundle is an OSGi-defined modularization unit. A bundle is a JAR file that:

- contains the resources necessary to provide some functionality. These resources may be class files for the Java programming language, as well as other data such as HTML files, help files, icons, etc. A bundle JAR file can also embed additional JAR files that are available as resources and classes; however, this is not recursive.
- contains a manifest file describing the contents of the JAR file and providing information about the bundle. This file uses headers to specify information that the OSGi Framework needs to install correctly and activate a bundle. For example, it states dependencies on other resources, such as Java packages that must be available to the bundle before it can run.
- can contain optional documentation in the POSG-OPT directory of the JAR file or one of its sub-directories. Any information in this directory is optional. For example, OSGI-OPT directory is useful to store source code of a bundle.

Once a bundle is started, its functionality is provided and services are exposed to other bundles installed in the OSGi Framework.

CO (Central Office)

A physical building or other structure housing a significant amount of network switching, transport or other ancillary equipment. A term traditionally used in the telecommunications industry.

CPE (Customer Premises Equipment)

Equipment or a device that provides a network connection and resides on the premises of a customer of subscriber of network services.

CVM (C Virtual Machine)

A full-featured JVM (Java Virtual Machine) designed for higher-end, emerging, next generation consumer electronic and embedded devices: devices with a 32-bit processor and 2Mb+ of total memory. These devices include wireless communicators, such as devices running Symbian's EPOC OS; high-end PDAs, for example, devices running embedded Linux or Windows CE; and residential gateways, automotive telematic systems, and screenphones.

Deployment package
Resources grouped as a unit of management. A deployment package can be installed, updated, and uninstalled as a unit. Applications are created using the Hitachi OSGi SJT. As with a bundle, a deployment package is a reified concept in an OSGi Service Platform. It is not known by the OSGi Framework, but it is managed by the OSGi Deployment Admin service, which is implemented as Management Agent (SJMA/SJDMS Client) by Hitachi. A deployment package is a stream of resources (including bundles) which, once processed, will result in new artifacts (effects on the system) being added to the OSGi platform. These new artifacts can include:

- installed Bundles
- new configuration objects added to the Configuration Admin service
- new wire objects added to the Wire Admin service
- changed system properties, etc.

All the changes caused by the processing of a deployment package are persistently associated with the deployment package, so that they can be appropriately cleaned up when the deployment package is uninstalled. There is a strict no overlap rule imposed on deployment packages. Two deployment packages are not allowed to create or manipulate the same artifact. This means that a bundle cannot be in two different deployment packages. Any violation of this no overlap rule is considered an error and the installation or update of the offending deployment package must be aborted.

DHCP
Dynamic Host Configuration Protocol. A protocol that provides a means to dynamically allocate IP addresses to computers on a local area network. The request and grant process uses a lease concept with a controllable time period. DHCP is defined in RFC 2131.

DSA (Digital Signature Algorithm)
A US government standard digital signing algorithm for the Digital Signature Standard. The signature resource name must have an extension of .DSA.

Applications in which wavelengths do not conform to the ITU-T spec are simply known as *WDM* or CWDM (Coarse WDM).

FTP (File Transfer Protocol)
A protocol that allows users to copy files between their local system and any system they can reach on the network.

IP
Internet Protocol

JVM (Java Virtual Machine)
A virtual machine enables a set of computer software programs and data structures to use a virtual machine model for the execution of other computer programs and scripts. The model used by a JVM accepts a form of computer intermediate language from Sun commonly referred to as Java bytecode. This language conceptually represents the instruction set of a stack-oriented, capability architecture.

LAN
Local Area Network

Mb
Megabit

OEM
Original equipment manufacturer.

OS
Operation System

OSGi Framework
The bundle runtime environment. It is an OSGi R4 compliant Java program developed by Hitachi-CTA that can be deployed on CPE devices to manage and execute bundle(s) of applications. An OSGi framework is managed by the SJDMS and its management agent (SJMA) through an API it provides.

PAR
PAR is a JSR124-defined file provisioning descriptor and is not a part of OSGi framework. In SJDMS R3.0, a PAR file contains only *one* deployment package and specifies the hardware restrictions and dependencies required by the contained deployment package. In future releases, it could contain more deployment packages. A PAR file contains all the information for the SJDMS to deploy an application. Application developers deliver the application in a PAR file. SJT's PAR file manager can create a

PAR file with multiple deployment packages in it. The manifest inside the PAR is an xml file, provisioning.xml. It contains the information about the deployment package, hardware restriction, and dependencies.

Port

A point of signal ingress or egress to/from a piece of network equipment or a component. A port may be electrical or optical, or may refer to a connector attachment point.

RSA (Rivest, Shamir and Adleman)

A popular public key digital signing algorithm. The extension of the signature resource name must be .RSA.

Service

One, or a group of, application(s) that a Service Provider has decided needs to be delivered as a common package. The OSGi framework does *not* understand the concept of service, per se. Services are kept and maintained *only* on the SJDMS. It is possible to group services into groups of services in a multi-level tree format such that services can contains other services and applications. This allows the creation of service templates that can be re-used when provisioning new devices.

SFTP (Secure Shell File Transfer Protocol)

A UNIX shell protocol that uses secure encrypted communications to allow users to copy files between their local system and any system they can reach over an insecure network.

SSH (Secure SHell)

A UNIX shell program for logging in and executing commands on a remote computer, intended to replace rlogin and rsh, and provide secure encrypted communications between two untrusted hosts over an insecure network. X11 connections and arbitrary TCP/IP ports can also be forwarded over the secure channel.

SuperJ™ Engine

A JVM (Java Virtual Machine) for micro processors based on Sun's Connected Device Configuration (CDC) that offers high compatibility features and is optimized for ARM, MIPS, PPC, and Hitachi's Super-H RISC processor.

INDEX