



# ChangeMan<sup>®</sup> ZMF for Eclipse<sup>™</sup> User's Guide

© Copyright 2001 - 2018 Micro Focus or one of its affiliates.

This document, as well as the software described in it, is furnished under license and may be used or copied only in accordance with the terms of such license. Except as permitted by such license, no part of this publication may be reproduced, photocopied, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of Serena. Any reproduction of such software product user documentation, regardless of whether the documentation is reproduced in whole or in part, must be accompanied by this copyright statement in its entirety, without modification.

The only warranties for products and services of Micro Focus and its affiliates and licensors ("Micro Focus") are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Contains Confidential Information. Except as specifically indicated otherwise, a valid license is required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Third party programs included with the ChangeMan ZMF for Eclipse product are subject to a restricted use license and can only be used in conjunction with ChangeMan ZMF for Eclipse.

Product version: 8.1.2

Publication date: September 2018 (rebranded only)

# Table of Contents

---

	<b>Welcome to ChangeMan ZMF for Eclipse . . . . .</b>	<b>9</b>
	Before You Begin . . . . .	9
	Documentation . . . . .	10
	Related Documentation . . . . .	10
	Using the Manuals . . . . .	11
<i>Chapter 1</i>	<b>Overview of ChangeMan ZMF for Eclipse . . . . .</b>	<b>13</b>
	Features and Functions. . . . .	14
	Components . . . . .	15
	Usage Scenarios . . . . .	15
	Enterprise e-Commerce Development in Java . . . . .	15
	Enterprise COBOL Application Maintenance. . . . .	17
	Plug-in Comparison: Eclipse Versus RDz . . . . .	18
<i>Chapter 2</i>	<b>Working with the Serena Perspective . . . . .</b>	<b>21</b>
	Serena Perspective Overview . . . . .	22
	ZMF/Eclipse Preferences . . . . .	24
	Working with the Serena Perspective . . . . .	26
	Opening the Serena Perspective . . . . .	26
	Switching Between Perspectives . . . . .	27
	Showing the Serena Explorer View . . . . .	28
	Hiding the Serena Explorer View . . . . .	29
	Opening ZMF Table Views. . . . .	31
	Accessing ChangeMan ZMF Repositories . . . . .	33
	Logging On to a ChangeMan ZMF Server . . . . .	33
	Logging Off from a ChangeMan ZMF Server . . . . .	35
	z/OS Resources Available on the Server. . . . .	35
	ZMF Server Functions. . . . .	36
	Working with z/OS Data Set Libraries. . . . .	37
	Refreshing a Data Set . . . . .	39
	Allocating a New Data Set . . . . .	39
	Creating a New Data Set Member . . . . .	40
	Deleting a Data Set . . . . .	41
	Downloading a Data Set. . . . .	41
	Migrating a Data Set Offline . . . . .	42
	Recalling a Migrated Data Set . . . . .	42
	Searching. . . . .	43
	Working with z/OS Data Set Members . . . . .	46
	Editing a Data Set Member. . . . .	47
	Browsing a Data Set Member . . . . .	48
	Downloading a Data Set Member . . . . .	48
	Checking In and Building a Data Set Member . . . . .	49

---

Submitting a Job for Execution . . . . .	49
Submitting an XML Services Request . . . . .	49
Deleting a Data Set Member . . . . .	50
Refreshing a Data Set Member . . . . .	50
Working with z/OS Unix HFS Folders . . . . .	50
Refreshing a Unix HFS Folder . . . . .	51
Deleting a Unix HFS Folder . . . . .	51
Downloading a Unix HFS Folder . . . . .	51
Working with z/OS Unix HFS Files . . . . .	53
Refreshing a Unix HFS File . . . . .	54
Deleting a Unix HFS File . . . . .	54
Downloading a Unix HFS File . . . . .	54
Editing a Unix HFS File . . . . .	55
Browsing a Unix HFS File . . . . .	55
Checking In and Building a Unix HFS File . . . . .	55
Working with z/OS Jobs . . . . .	55
Browsing Job Output . . . . .	56
Canceling a Job and Deleting Job Output . . . . .	57
Working with ZMF Applications . . . . .	57
Adding an Application to a Desktop Workspace . . . . .	60
Working with ZMF Packages . . . . .	62
Working with ZMF Components . . . . .	67
ZMF Operations on Baseline Library Components . . . . .	67
ZMF Operations on Package Components . . . . .	69
ZMF Operations on Promoted Components . . . . .	73
ZMF Notifications . . . . .	75
Filtering z/OS Data Sets . . . . .	77
Creating a New Data Set Filter . . . . .	77
Modifying a Data Set Filter . . . . .	79
Renaming a Data Set Folder . . . . .	79
Deleting a Data Set Folder . . . . .	80
Refreshing a Data Set Filter . . . . .	80
Filtering z/OS Unix HFS Files . . . . .	81
Creating a New Unix HFS Folder . . . . .	81
Modifying a Unix HFS Filter . . . . .	83
Renaming a Unix HFS Filter . . . . .	83
Deleting a Unix HFS Filter . . . . .	84
Refreshing Unix HFS Filters . . . . .	84
Filtering z/OS Jobs . . . . .	84
Creating a New Job Filter . . . . .	85
Modifying an Existing Job Filter . . . . .	86
Renaming a Job Filter . . . . .	86
Deleting a Job Filter . . . . .	87
Refreshing Job Filters . . . . .	87
Filtering ZMF Applications . . . . .	87
Modifying an Application Folder . . . . .	87
Filtering Package Views . . . . .	88
Working with ZMF Releases . . . . .	91

<i>Chapter 3</i>	<b>Working with the Java Perspective . . . . .</b>	<b>95</b>
	Java Perspective Overview . . . . .	96
	Opening the Java Perspective . . . . .	97
	Switching Between Perspectives . . . . .	98
	Enabling ZMF Functions in the Team Menu . . . . .	99
	Dynamic Integration with ChangeMan ZMF . . . . .	100
	ZMF Functions in the Package Explorer . . . . .	101
	Project-Level ZMF Functions . . . . .	102
	Folder-Level ZMF Functions . . . . .	105
	Component-Level ZMF Functions . . . . .	107
<i>Chapter 4</i>	<b>Working with RDz Perspectives . . . . .</b>	<b>109</b>
	z/OS Projects Perspective Overview . . . . .	110
	RDz Projects and Subprojects . . . . .	110
	Using the Smart Editor . . . . .	111
	Checking a Component into ZMF from an RDz Project . . . . .	118
<i>Chapter 5</i>	<b>ChangeMan ZMF Component Functions . . . . .</b>	<b>121</b>
	Creating a New Component . . . . .	122
	Deleting a Component . . . . .	124
	Scratching a Component under Change Control . . . . .	125
	Renaming a Component under Change Control . . . . .	127
	Checking Out a Component . . . . .	129
	Checkout Parameters Window . . . . .	131
	Select File Window . . . . .	132
	Checking In a Component . . . . .	134
	Viewing Component History . . . . .	138
	Component History during Checkout and Checkin . . . . .	139
	Component History in Contextual Menus . . . . .	140
	Locking and Unlocking Components . . . . .	141
	Browsing a Component . . . . .	141
	Browsing a Component Listing . . . . .	142
	Editing a Component . . . . .	143
	Building a Component . . . . .	143
	Recompiling a Component . . . . .	150
	Relinking a Component . . . . .	155
	Viewing the Component Staging Versions List . . . . .	159
	Browsing Component Staging Versions . . . . .	162
	Comparing Component Staging Versions . . . . .	162
	Comparing Components to Baseline or Promotion . . . . .	163
	Source-to-Load Relationships . . . . .	164
	Component Bill of Materials . . . . .	165
	Component Impact Analysis . . . . .	168
	Query Component Functionality . . . . .	171
<i>Chapter 6</i>	<b>ChangeMan ZMF Package Functions . . . . .</b>	<b>175</b>
	Deleting a Change Package . . . . .	190
	Undeleting a Change Package . . . . .	191

Editing Package Properties . . . . .	192
Package Properties Wizard Screens . . . . .	192
Removing Scratch Records from a Package . . . . .	193
Removing Rename Records from a Package . . . . .	193
Freezing a Package . . . . .	194
Unfreezing and Refreezing Package Categories . . . . .	195
Unfreezing Package Categories . . . . .	195
Refreezing Package Categories . . . . .	196
Unfreezing and Refreezing Package Components . . . . .	197
Promoting a Package . . . . .	199
Promotion Parameter Window . . . . .	201
Promotion Type and Component Selection Window . . . . .	203
User Variables Window . . . . .	205
Promotion Overlays Window . . . . .	206
Displaying the Scheduled Package Promotions View . . . . .	206
Deleting, Holding, Releasing, or Updating a Scheduled Package Promotion	207
Demoting a Package . . . . .	209
Demotion Parameters Window . . . . .	210
Demotion Type and Component Selection Window . . . . .	212
User Variables Window . . . . .	213
Auditing a Package . . . . .	213
Specify Audit Parameters . . . . .	216
Select Applications for Audit . . . . .	220
Audit Package - User Variables . . . . .	221
Resetting Audit Lock . . . . .	221
Approving or Rejecting a Package for Installation . . . . .	222
Rebuilding Installation JCL . . . . .	224
Backing a Package Out of Production . . . . .	225
Reverting a Package to Development Status . . . . .	227
Querying Packages . . . . .	228
Querying Package Components . . . . .	235
Viewing the Component Work List . . . . .	237
Viewing Source-to-Load Relationships . . . . .	237
Viewing Scratch/Rename Components . . . . .	238
Viewing Promotion History . . . . .	239
Viewing Promotion Libraries . . . . .	239
Viewing Baseline Libraries . . . . .	240
Displaying Site Activities . . . . .	241
<b>Appendix A Error Messages and Troubleshooting . . . . .</b>	<b>243</b>
Unidentified Nested Exception at ZMF Logon . . . . .	244
Logoff Fails Due to Invalid Session . . . . .	244
Component Browse Function Fails . . . . .	244
Checkout to RDz Project Fails . . . . .	245
Code Page Issues with Comments in ZDDOPTS . . . . .	245
ZDDOPTS Updates Do Not Affect Workbench . . . . .	247

**Index. . . . . 249**





# Welcome to ChangeMan ZMF for Eclipse

---

**Product Description** ChangeMan<sup>®</sup> ZMF for Eclipse™ provides plug-in integration between ChangeMan ZMF, the industry-leading software change management system for IBM System z mainframes, and the software development environments provided by the following products:

- Eclipse™ Workbench
- IBM<sup>®</sup> RDz

These products include software developed by the Eclipse Project (<http://www.eclipse.org>).

ZMF for Eclipse is an optional licensed feature of ChangeMan ZMF. It is distributed as part of the **ChangeMan ZMF Client Pack**.

## Before You Begin

**Readme** Refer to the Readme file for the latest updates on software compatibility, patches, known issues, workarounds, and other late-breaking information before you install ChangeMan ZMF for Eclipse. This information changes frequently and is refreshed in the Readme file between updates to this manual.

The latest Readme for ZMF for Eclipse can be downloaded from the **ChangeMan ZMF Client Pack** product download page on the Serena Customer Support Web site at <http://support.serena.com>.

This document assumes that you are familiar with the Integrated Development Environment (IDE) provided by the Eclipse Workbench or IBM Rational Developer for z Systems (RDz) on a Microsoft Windows platform. It also assumes familiarity with software change control concepts and functions implemented in ChangeMan ZMF. If you are new to ChangeMan ZMF, refer to the "Related Documentation" table below for information not included in this manual.

- IBM RDz with WebSphere Application Server on Microsoft Windows.

**Disclaimers** The instructions in this manual do not replace the documentation for:

- **Eclipse Workbench from the Eclipse Project.**  
Your primary source of information about the Eclipse Workbench is the Eclipse Project Web site (<http://www.eclipse.org>).
- **IBM Rational Developer for z Systems (RDz).**  
The documentation provided by IBM is your primary source of information regarding those products. IBM also provides technical reference information and an online library for RDz and related products on the Rational developerWorks Web site (<http://www.ibm.com/developerworks/rational/>).
- **Apache Tomcat, IBM WebSphere Application Server, or other Web application server.**  
Platform-specific information related to your Web application server environment should be obtained from your Web application server vendor. For example, your primary source of information about Apache Tomcat is the Apache Software Foundation Web site (<http://tomcat.apache.org/>). Your primary source of

information regarding IBM WebSphere Application Server (WAS) is the documentation provided by IBM. IBM maintains an online documentation library for WAS at <http://www-01.ibm.com/software/webservers/appserv/was/library/>.

- **Microsoft Windows or other client operating systems.**  
Platform-specific information about your development operating system environment should be obtained from your operating system vendor. For example, your primary source of information about Microsoft Windows is the Microsoft Web site (<http://www.microsoft.com>).

Serena makes no representations or warranties regarding Eclipse Workbench or Rational Developer for z Systems, that the instructions contained in this document are valid. Refer to the product documentation supplied with these products for authoritative information.

Change Bars    Change bars are not used due to the large number of changes since the last edition.

## Documentation

ZMF for Eclipse includes information resources in the form of downloadable documentation on the Serena Customer Support Web site.

### Product Documentation

The following documentation (PDF) for ZMF for Eclipse may be downloaded from the **ChangeMan ZMF Client Pack** product download page on the Serena Customer Support Web site.

Manual	Description
<i>Serena ChangeMan ZMF for Eclipse Installation and Configuration</i>	Installation, configuration, and customization options for the ZMF for Eclipse plug-in.
<i>Serena ChangeMan ZMF for Eclipse User's Guide</i>	User's Guide for ZMF for Eclipse in Eclipse and RDz developer workbench environments.
<i>Serena ChangeMan ZMF Web Services Getting Started Guide</i>	Reference information concerning the Web Services integration technology used by ZMF and ZMF for Eclipse.

### Related Documentation

Refer to the following documents to learn more about ChangeMan ZMF functionality and its integration with the workbench environment. These documents may be downloaded from the **ChangeMan ZMF** product download page on the Serena Customer Support Web site.

Manual	Description
<i>Serena ChangeMan ZMF Installation Guide</i>	Installation and setup instructions for ChangeMan ZMF — including XMLSPACE setup, a prerequisite for ZMF for Eclipse.

Manual	Description
<i>Serena ChangeMan ZDD Server Installation Guide</i>	Installation and setup information for the ChangeMan ZDD server on the mainframe, including customization information for the ZDDOPTS client configuration library. Settings defined in ZDDOPTS apply to ZMF for Eclipse.
<i>Serena ChangeMan ZMF User's Guide</i>	How-to instructions for the functions and facilities of ChangeMan ZMF. Developer functions documented here are also available through ZMF for Eclipse.
<i>Serena ChangeMan ZMF Messages Guide</i>	Explanations & recovery tips for messages issued by ZMF. These messages are passed to ZMF for Eclipse.

## Using the Manuals

PDF and  
Adobe Reader

The documentation for ChangeMan ZMF and ZMF for Eclipse are provided in the Adobe Portable Document Format (PDF). To view PDF files, use Adobe® Acrobat Reader®, which is freely available for download from <https://get.adobe.com/reader/>.



**TIP** Be sure to download the *latest full version* of Adobe Acrobat Reader. Older more basic versions may not include the multiple-book search feature.

Finding  
Information

The PDF documents include the following features to help you find and use information:

- **Bookmarks.** All of the manuals contain predefined bookmarks that make it easy for you to quickly jump to a specific topic. By default, the bookmarks appear to the left of each online manual.
- **Links.** Cross-reference links within a manual enable you to jump to other sections within the manual and to other manuals with a single mouse click. These links appear in blue.
- **Advanced search.** Starting with version 6, Adobe Reader includes an advanced search feature that enables you to search across multiple PDF files in a specified directory.

Multi-Document  
Full Text Search

To search across multiple PDF documents at once for a text string, perform the following steps (requires Adobe Reader 6 or higher):

- 1 In Adobe Reader, select **Edit | Search** (or press CTRL+F).
- 2 Select the drive or folder containing the PDF documents you want to search.
  - a In the search window under **Where would you like to search?**, select the **All PDF Documents in** option.
  - b Scroll down to the bottom of the pull-down menu and select **Browse for Location**.
  - c When the browse window appears, navigate to the location containing the documents you want to search and click **OK**.
- 3 In the text box below your search location, enter the word or phrase for which you want to search.
- 4 Optionally, select one or more of the additional search options, such as **Whole words only** or **Case-Sensitive**.
- 5 Click the **Search** button.



# Chapter 1

---

## Overview of ChangeMan ZMF for Eclipse

Features and Functions	14
Components	15
Usage Scenarios	15
Plug-in Comparison: Eclipse Versus RDz	18

## Features and Functions

Serena® ChangeMan® ZMF for Eclipse™ is a Java plug-in for the Eclipse™ Workbench and IBM Rational® Developer for z Systems® (RDz) integrated development environments (IDEs). The plug-in integrates the mainframe software change management capabilities of ChangeMan ZMF directly into these workbench environments.

### Benefits

Some benefits of Eclipse integration with ChangeMan ZMF are the following:

- Tool availability — including the latest GUI editors, debuggers, generators, and application programmer interface (API) toolkits for Java, COBOL, and Assembler when working with mainframe software assets.
- A unified point of access through the workbench to multiple ChangeMan ZMF repositories. The Dimensions for Eclipse plug-in adds unified access through the Serena Explorer view to Dimensions CM resources as well.
- Full integration with the robust version control, build management, system test, and multisystem deployment capabilities of a software change control system — ChangeMan ZMF — that is optimized for enterprise-scale projects.
- Mainframe levels of security and disaster protection for the software repository.

### Supported ZMF Functions

Nearly all the developer functions of ChangeMan ZMF are supported by ZMF for Eclipse. In addition, many ZMF tasks are performed automatically "under the covers" as you work with development resources in the workbench. For example, you can:

- Perform development lifecycle tasks from the desktop — including checkouts from baseline, checkin to a change package, component builds, package freeze, package audit, package create, promote/demote, and more.
- Perform Release Management tasks from the desktop — including Unblock Release, Block Release, Approve Release, Backout Release, Revert Release, Audit Release, Test Release, Area Reset Notify, Area Checkin Notify, and more.
- Work with controlled software assets and personal development libraries in the native z/OS file system or the z/OS Unix Hierarchical File System (HFS).
- Perform mass downloads with automated mapping of library types to directories, automated checkout and lock when the resource is changed, library synchronization against multiple repository resources, and mass checkins with automatic package creation when working with desktop development libraries in the workspace.
- Submit jobs or ZMF XML Services data streams to the mainframe for execution.
- Access the JES subsystem to review mainframe started task and job output.

---

# Components

## Workbench Plug-in

ZMF for Eclipse consists of a Java plug-in that is compatible with both the open-source Eclipse Workbench and IBM's mainframe-oriented Rational Developer for z Systems, which is built on Eclipse technology. The plug-in is installed in the workbench environment.

## Web Application Server

The ZMF for Eclipse plug-in invokes the ChangeMan ZMF Web Services to manage function-specific connectivity to ChangeMan ZMF. The ZMF Web Services reside on a Web application server that executes either on the development system where the workbench resides or at some network location accessible to the client. ZMF for Eclipse works with Web application servers that support the Apache Axis SOAP library.

## Software Repository

The Web application server and ZMF Web Services use TCP/IP to connect to the ChangeMan ZMF software repository on the mainframe. ChangeMan ZMF manages software assets to meet the demanding change control requirements of enterprises and organizations running large-scale, mission-critical applications.

# Usage Scenarios

Because it is built on cross-platform technology such as Java and Web Services, ZMF for Eclipse has the flexibility to support a wide range of use cases and installation scenarios. The following are just two examples:

- [Enterprise e-Commerce Development in Java](#)
- [Enterprise COBOL Application Maintenance](#)

## Enterprise e-Commerce Development in Java

### *The Challenge*

The sheer scale and complexity of enterprise e-commerce applications require robust build management and cross-platform software deployment capabilities at the level of the software repository. Moreover, many industries are highly regulated and face specific compliance requirements for software change control. ChangeMan ZMF has long satisfied these requirements for traditional enterprise software repositories.

However, traditional enterprise programming languages and tools are not ideally suited to the development of e-commerce applications. Java has established itself as the preferred language for large-scale e-commerce development. In part, this is because the Java open-source community offers prebuilt software development toolkits (SDKs), software libraries, code generators, debuggers, test tools, automated documentation tools, and other development tools for multiple operating system platforms. This makes Java a

highly productive language for cross-platform and Web-enabled application development. Mainframe e-commerce applications could be developed more quickly and cost-effectively by deploying Java components wherever appropriate.

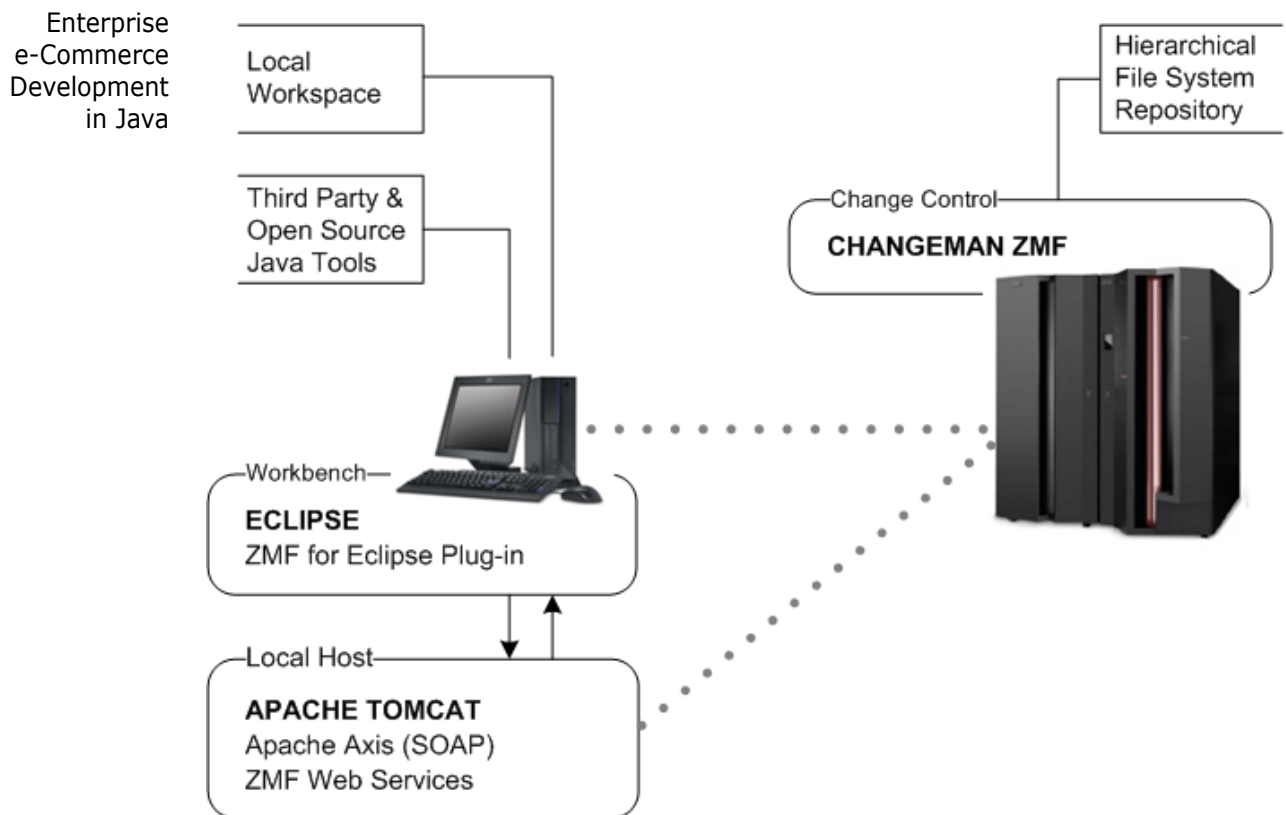
### The Solution

For enterprise-scale e-commerce applications, ChangeMan ZMF and ZMF for Eclipse support the z/OS Unix Hierarchical File System (HFS) as personal development libraries and for baseline, staging, promotion, and production libraries. HFS libraries provide native support for long file names and Java classpaths.

Complementing z/OS Unix System Services on the mainframe is the ZMF for Eclipse client for the open-source Eclipse Workbench and IBM's Rational Developer for z Systems. ZMF for Eclipse uses web services technology to integrate these popular and productive, Java-oriented IDEs with the enterprise repository management capabilities of ChangeMan ZMF. Eclipse integration is important because the Java developer community has largely adopted the open-source Eclipse Workbench IDE as its development platform of choice.

### Implementation Use Case

The diagram below illustrates a usage scenario in which Java code for enterprise ecommerce is developed in a desktop workspace managed by the open-source Eclipse Workbench on Windows. Open-source development toolkits, code libraries, and plug-ins for Java can all be leveraged within the workbench, and unit testing occurs against the Java Virtual Machine running on the desktop.



For ZMF Web Services connectivity to ChangeMan ZMF, Apache Tomcat is chosen as the Web application server.



Using ZMF Web Services under the covers, the new, one-click application download feature of ZMF for Eclipse copies an entire ZMF application to the desktop at once. When a downloaded component is changed, ZMF for Eclipse automatically checks it out of baseline, stages it to development, and locks it in the ZMF change package automatically. The ZMF for Eclipse mass checkin feature uploads the entire application and checks in all components automatically to a newly created change package.

Mapping between ZMF libraries and application file types and directories on the workbench is automated. Library synchronization is supported by ZMF for Eclipse between local workspace projects and ZMF baseline or staging libraries. Side-by-side difference comparison and merge are supported in the workbench.

System testing and deployment into production are managed by ChangeMan ZMF.

## **Enterprise COBOL Application Maintenance**

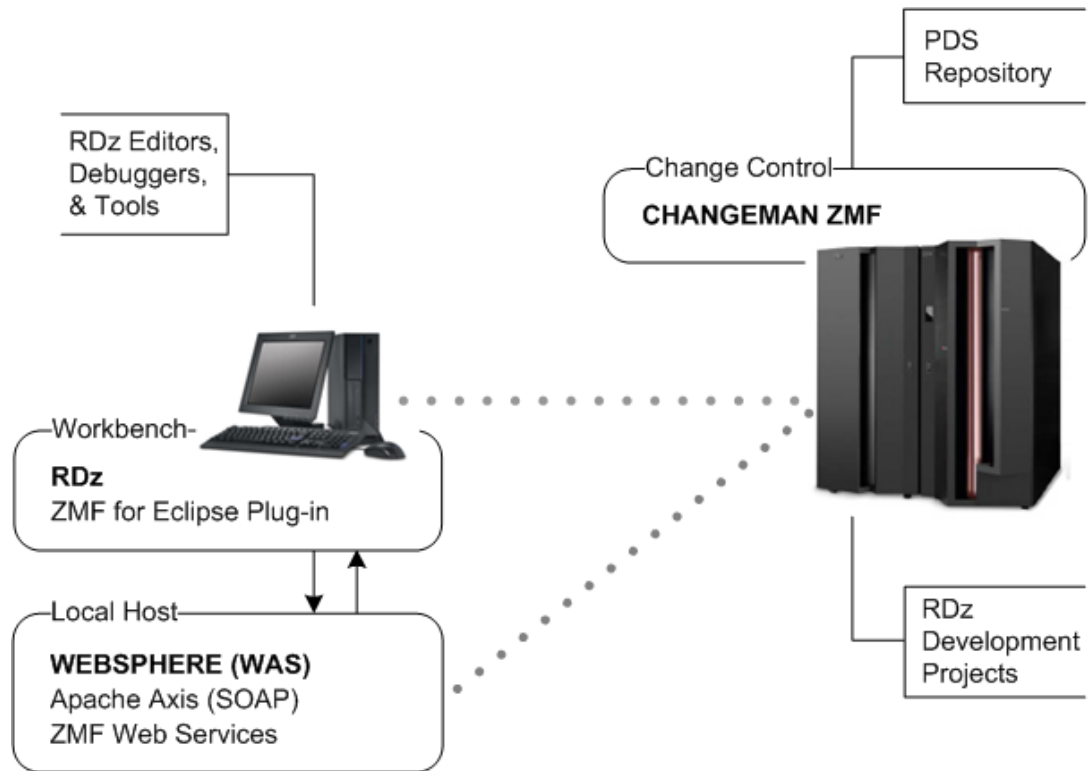
### ***Enterprise Development Requirements***

For the maintenance of enterprise COBOL applications, including applications built over the DB2 relational database, IBM's Rational Developer for z Systems (RDz) workbench provides a powerful and wide-ranging set of development, debugging, and testing tools with GUI ease-of-use. Built on an Eclipse base, RDz's development tools complement the robust, enterprise-scale software change control, build management, and deployment features of ChangeMan ZMF.

### ***Implementation Use Case***

In this use case, COBOL code, JCL, and associated DB2 plans and artifacts are checked out of a ZMF repository baseline into an RDz development project. Each RDz project maps to a personal development library in the native z/OS file system on the mainframe. Keeping development libraries on the mainframe under RDz project management maintains control of corporate software assets. The diagram below illustrates the basic features of this usage scenario.

Enterprise  
COBOL Application  
Maintenance



For ZMF Web Services connectivity to ChangeMan ZMF, IBM WebSphere Application Server (WAS) is chosen as the Web application server.

The Serena perspective and the Serena Explorer view provide a unified and configurable view of software resources in all personal development libraries on the mainframe, as well as baseline, staging, and promotion libraries in the ZMF repository. Compressed output listings can be read on the desktop in the Serena perspective thanks to a Serena-provided editor installed with the plug-in. The z/OS Explorer view of RDz provides access to project resources. Job output in the JES job queue can be accessed in either view.

## Plug-in Comparison: Eclipse Versus RDz

ZMF for Eclipse plug-in behavior varies somewhat with the IDE in which it is installed. The following table compares plug-in behavior in the open-source Eclipse Workbench versus Rational Developer for z Systems.

	<b>Eclipse Workbench</b>	<b>RDz</b>
NAVIGATION	<ul style="list-style-type: none"> <li>■ Serena Explorer</li> <li>■ Package Explorer (Java)</li> <li>■ Remote System Explorer</li> </ul>	<ul style="list-style-type: none"> <li>■ Serena Explorer</li> <li>■ Package Explorer (Java)</li> <li>■ z/OS Projects Explorer (WebSphere)</li> </ul>
PROJECT MANAGEMENT	<ul style="list-style-type: none"> <li>■ Eclipse projects on desktop</li> </ul>	<ul style="list-style-type: none"> <li>■ Eclipse projects on desktop</li> <li>■ RDz projects on mainframe</li> </ul>
CHECK-OUT LOCATION	<ul style="list-style-type: none"> <li>■ Mainframe development libraries</li> <li>■ ChangeMan ZMF repository change package</li> </ul>	<ul style="list-style-type: none"> <li>■ Mainframe development libraries</li> <li>■ ChangeMan ZMF repository change package</li> <li>■ RDz projects (mainframe libraries)</li> </ul>

	<b>Eclipse Workbench</b>	<b>RDz</b>
LOCAL WORKSPACE	<ul style="list-style-type: none"> <li>■ Download z/OS data sets</li> <li>■ Download z/OS Unix HFS directories</li> <li>■ Mass download of ZMF application</li> <li>■ Mass checkin to ZMF change package</li> <li>■ Checkout from package staging libraries</li> <li>■ Checkout from promotion libraries</li> </ul>	<ul style="list-style-type: none"> <li>■ Download z/OS data sets</li> <li>■ Download z/OS Unix HFS directories</li> <li>■ Mass download of ZMF application</li> <li>■ Mass checkin to ZMF change package</li> <li>■ Checkout from package staging libraries</li> <li>■ Checkout from promotion libraries</li> </ul>
Z/OS DATA SET ACCESS	<ul style="list-style-type: none"> <li>■ Create</li> <li>■ Delete</li> <li>■ Browse</li> <li>■ Edit</li> <li>■ Download</li> </ul>	<ul style="list-style-type: none"> <li>■ Create</li> <li>■ Delete</li> <li>■ Browse</li> <li>■ Edit</li> <li>■ Download</li> </ul>
Z/OS UNIX HFS OBJECT ACCESS	<ul style="list-style-type: none"> <li>■ Create</li> <li>■ Delete</li> <li>■ Browse</li> <li>■ Edit</li> <li>■ Download</li> </ul>	<ul style="list-style-type: none"> <li>■ Create</li> <li>■ Delete</li> <li>■ Browse</li> <li>■ Edit</li> <li>■ Download</li> </ul>
JES JOB ACCESS	<ul style="list-style-type: none"> <li>■ Submit jobs</li> <li>■ Cancel jobs</li> <li>■ Display job output</li> </ul>	<ul style="list-style-type: none"> <li>■ Submit jobs</li> <li>■ Cancel jobs</li> <li>■ Display job output</li> </ul>
XML SERVICES	<ul style="list-style-type: none"> <li>■ Submit XML requests to ZMF</li> <li>■ View XML Services replies in editor</li> </ul>	<ul style="list-style-type: none"> <li>■ Submit XML requests to ZMF</li> <li>■ View XML Services replies in editor</li> </ul>
JAVA INTEGRATION	<ul style="list-style-type: none"> <li>■ Team menu of Java perspective</li> <li>■ z/OS Unix Hierarchical File System</li> <li>■ Open-source Java toolkits</li> </ul>	<ul style="list-style-type: none"> <li>■ Team menu of Java perspective</li> <li>■ z/OS Unix Hierarchical File System</li> <li>■ Open-source Java toolkits</li> </ul>
LANGUAGE SUPPORT		<ul style="list-style-type: none"> <li>■ RDz development &amp; test tools which include COBOL, PL/I, Assembler, Java editors</li> </ul>



## Chapter 2

---

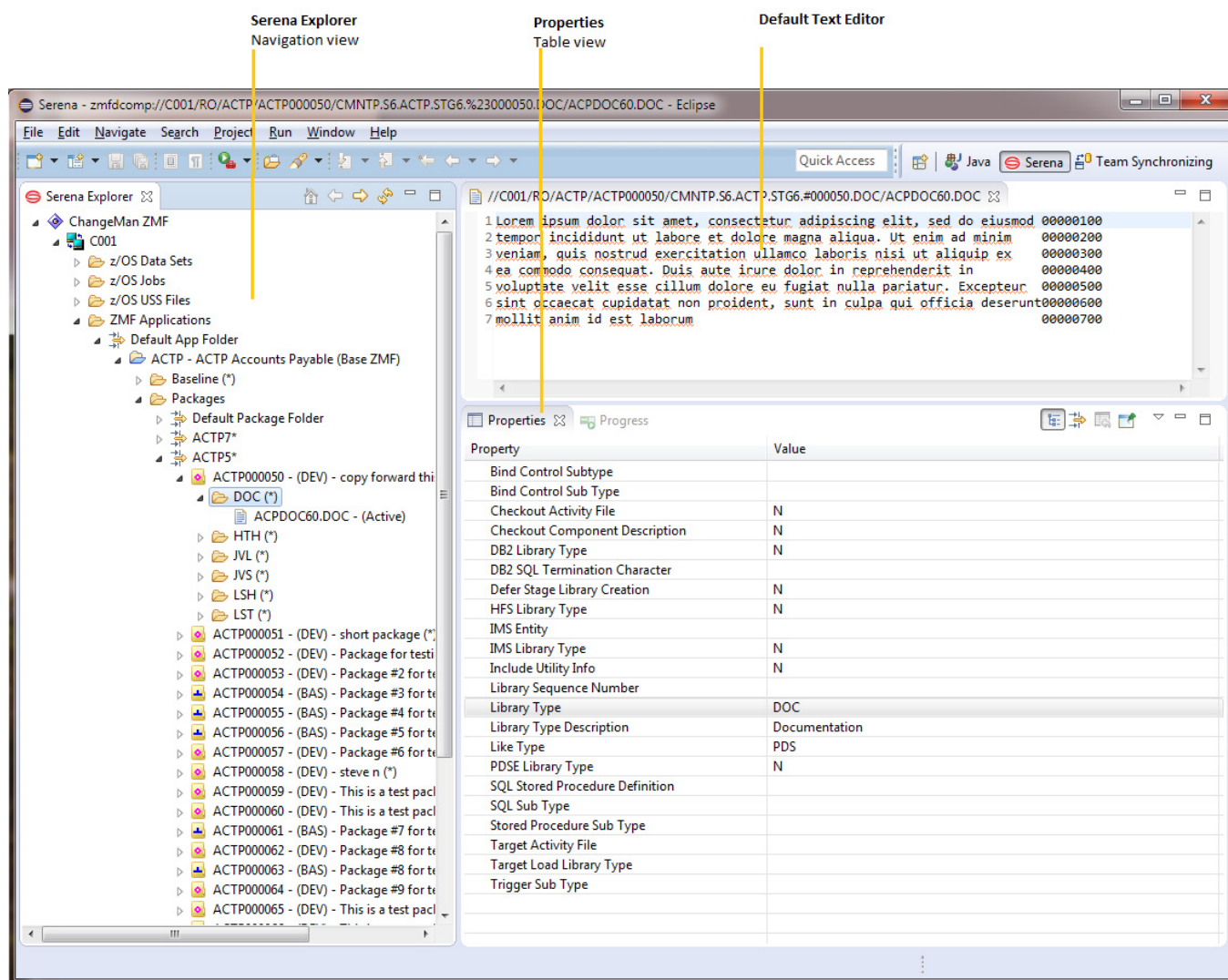
# Working with the Serena Perspective

Serena Perspective Overview	22
ZMF/Eclipse Preferences	24
Working with the Serena Perspective	26
Accessing ChangeMan ZMF Repositories	33
ZMF Server Functions	36
Working with z/OS Data Set Libraries	37
Working with z/OS Data Set Members	46
Working with z/OS Unix HFS Folders	50
Working with z/OS Unix HFS Files	53
Working with z/OS Jobs	55
Working with ZMF Applications	57
Working with ZMF Packages	62
Working with ZMF Components	67
ZMF Notifications	75
Filtering z/OS Data Sets	77
Filtering z/OS Unix HFS Files	81
Filtering z/OS Jobs	84
Filtering ZMF Applications	87

## Serena Perspective Overview

The ZMF for Eclipse plug-in provides a default *perspective* which contains *views* that are unique to the integration and an *editor* area that displays your files in the editors you have associated with each filename extension. Multiple files show in multiple editor tabs.

The default **Serena** perspective displays the **Serena Explorer** view in the left navigation pane, one or more editors in the upper right pane, and the **Properties** view in the pane at the lower right. The active view is highlighted in blue.



### Serena Explorer Navigation View

The **Serena Explorer** view in the Serena perspective provides hierarchical navigation among the mainframe resources available to ZMF for Eclipse. These include ChangeMan ZMF packages and components, as well as your personal development libraries on the mainframe and the output of mainframe jobs you submitted for execution.

- **Expand or collapse a node** in the hierarchy by clicking on its triangle symbol.
- **Open the contextual menu for a resource** by right-clicking on its name.
- **Perform actions on a resource** by choosing commands from its contextual menu.

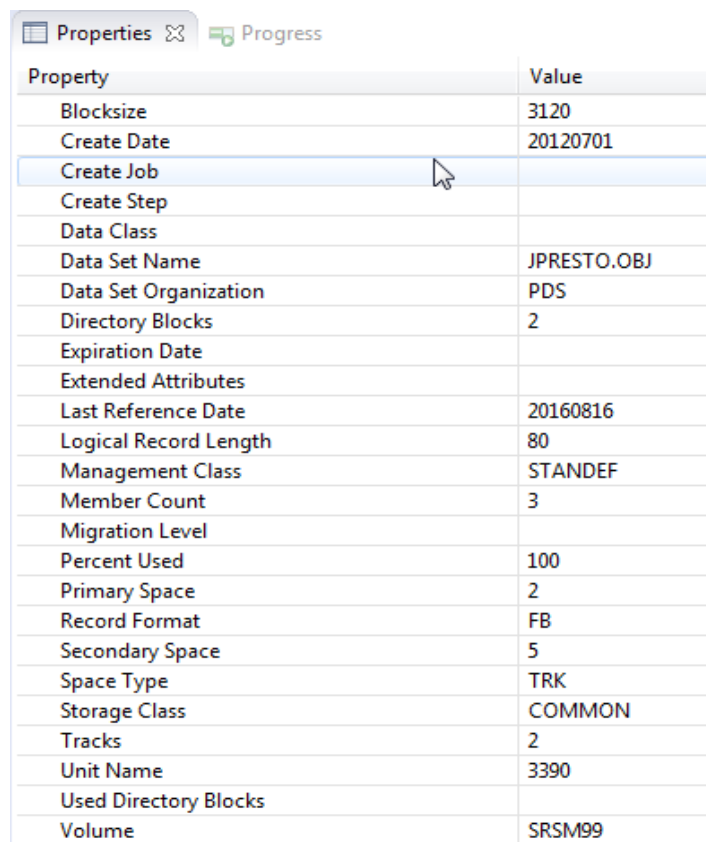
## Workbench Editors

When you double-click on a component or select **Browse** from its contextual menu in **Serena Explorer**, a tabbed editor pane opens in the editor area of the **Serena** perspective. The editor invoked by default is one associated in the workbench with the file type of the component in the workbench, but you can select other editors. In addition, Serena supplies its own editor for viewing compressed mainframe listings. Character code page translations between EBCDIC and ASCII or Unicode are handled transparently in the background.

## Properties Table View

The entries shown in the **Properties** view vary with the object selected in the **Serena Explorer** view. In the example shown above, the DOC library in package ACTP000050 is selected in **Serena Explorer**, so it is the ChangeMan ZMF properties of that library that appear in the **Properties** view.

The **Properties** view for a data set displays the following extended attribute values, as appropriate for the target data set:



Property	Value
Blocksize	3120
Create Date	20120701
Create Job	
Create Step	
Data Class	
Data Set Name	JPRESTO.OBJ
Data Set Organization	PDS
Directory Blocks	2
Expiration Date	
Extended Attributes	
Last Reference Date	20160816
Logical Record Length	80
Management Class	STANDEF
Member Count	3
Migration Level	
Percent Used	100
Primary Space	2
Record Format	FB
Secondary Space	5
Space Type	TRK
Storage Class	COMMON
Tracks	2
Unit Name	3390
Used Directory Blocks	
Volume	SRS099

## ZMF/Eclipse Preferences

The following pages are available for configuring preferences in ZMF for Eclipse:

- [Allocate](#)
- [Checkouts](#)
- [Settings](#)
- [Site Data](#)

These pages can be found under **Window | Preferences | ZMF/Eclipse**.

### Allocate

The **Allocate** preference page provides default values for data set allocation. The default values are used in either of the following situations:

- when a data set is allocated using the **New Data Set** function
- when a personal library is allocated during Checkout and the **Allocation Preferences Tab** option on the **Checkouts** preference page was selected

### Checkouts

The **Checkouts** preference page provides options for checking out components to a personal development library.

The preference options on this page are:

Preference	Description
Checkout Dialog Last Personal Library:	Identifies how ZMF4ECL fills in the personal library text box during checkout: <b>Last Personal Library Used</b> -- Retains the same library from checkout to checkout. <b>Personal Library Based on Mask</b> -- Substitutes appropriate variables into the personal library mask and uses the updated value.
Obtain Personal Library Allocation Parameters From:	Directs ZMF4ECL to get allocation parameters from one of the following: <b>ZMF Library Type Definition</b> -- Gets values from the ZMF library type definition. <b>Allocation Preferences Tab</b> -- Gets values from the <b>Allocate</b> preference page.
Personal Library Mask (Data Sets)	Specify the pattern for personal library data set.
Personal Library Mask (HFS)	Specify the pattern for personal library Hierarchical File System (HFS) file.
Auto-Allocate personal library	Directs ZMF4ECL to automatically allocate the specified personal library if it does not exist.



## Settings

Behavior of some options are controlled by the ZMF for Eclipse application server settings selected on the **Settings** preference page.

- **Web Services host** defines the host
- **Web Services port** defines the port
- **Web Services context** defines the context (directory from the jar file) to use on the web services host, and this allows you to easily change when you need to run multiple contexts on a single host.
- **Notification port** defines the port to be used for notifications.
- The **Show Notify Message Dialog** setting shows a dialog box with each notification message as it is received from ZMF.
- The **Maintain Job Card for each Server** setting enables you to run several different ZMF systems with custom job cards for each server. All functions that use a job card, such as build, promote, and audit, will use the server-specific job card if this option is set. If it is not set, only one job card is used for all servers.
- The **Translate Job Card to Uppercase** setting converts any lowercase characters that you specify in a JOB statement to uppercase before the job is submitted.

NOTE: The z/OS Job Entry Subsystem (JES) requires all keywords and user-supplied values on a JOB statement to be uppercase.

- The **Display Batch Checkout Warning Message** setting displays a dialog box with a warning message when a batch checkout is performed. This is a reminder that the batch job needs to run to completion before you can work with the components.
- The **Launch Build after saving component** setting, if set, will automatically launch a build dialog after a component is saved.
- The **Auto Submit build if component history exists** setting will automatically submit a build after a component is saved if there is a history. Can be prevented by selecting **Build (with Dialog)** from the context menu.
- The **Prompt for Change Description** setting prompts for a change description upon checkin even if the ZMF SSV, or Staging Versions, option for a particular library type is configured as optional.
- The **Display Resource Filters** setting shows any active filters for each resource in the Serena Explorer Tree. With this option turned off, none of the filter values are displayed.
- **Sort Packages Descending on Serena Explorer** when checked will reverse the sort order of packages in the Serena Explorer from Ascending to Descending.
- The **Trace RDz Integration Calls** setting writes tracing/diagnostic information to a log file. Serena Support may ask a user to enable this option to assist in gathering debugging information for problems with the RDZ Integration.
- The **Trace HLLX processing** setting writes tracing/diagnostic information to a log file. Serena Support may ask a user to enable this option to assist in gathering debugging information for problems with HLLX processing.

For more information, see the *ChangeMan ZMF for Eclipse Installation and Configuration Guide*.

## Site Data

The **Site Data** preference page contains default values for primary and secondary contacts and a specific install date and time (from and to) information, and relative install date in days. These values apply to both DP and ALL sites and are used as defaults on the **Site Information Page** of the **Package Create** wizard if no other values are available.

## Working with the Serena Perspective

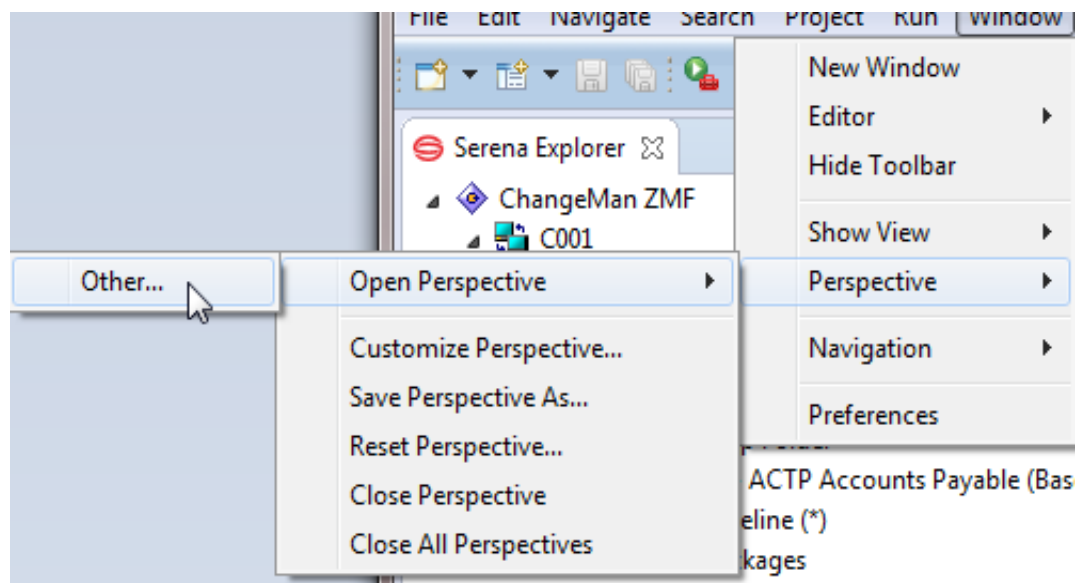
You can customize the **Serena** perspective as needed to suit your personal preferences and to rearrange your tools as you move from project to project or task to task. You should become familiar with the following workbench essentials.

- [Opening the Serena Perspective](#)
- [Switching Between Perspectives](#)
- [Showing the Serena Explorer View](#)
- [Hiding the Serena Explorer View](#)
- [Filtering Views in Serena Explorer](#)
- [Opening ZMF Table Views](#)

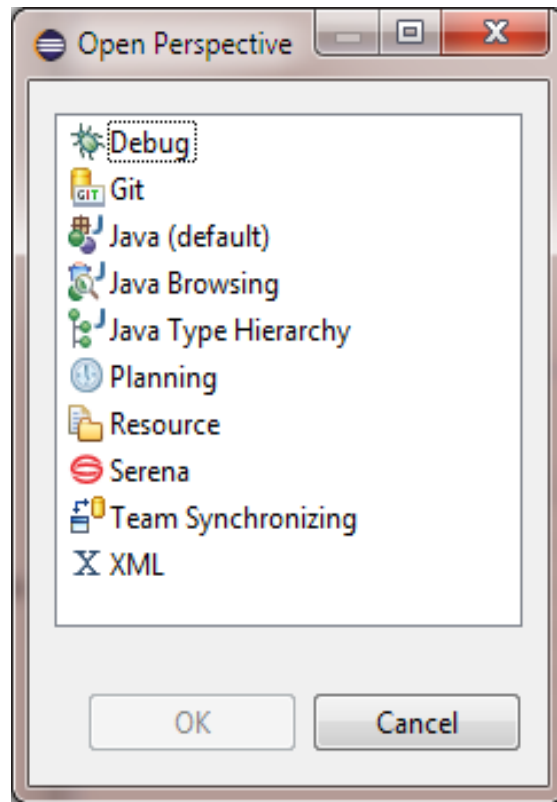
### Opening the Serena Perspective

To open the Serena perspective in the workbench, perform the following steps.

- 1 From the workbench **Window** menu, select **Perspective | Open Perspective | Other**.



- 2 When the **Open Perspective** dialog appears listing the "other" perspective choices, select the **Serena** perspective.

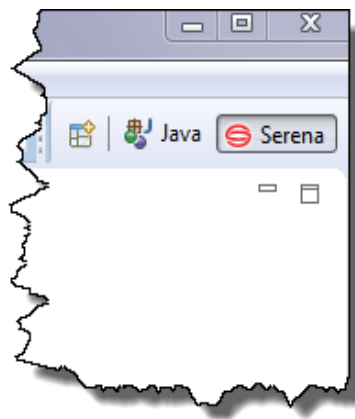


3 Click **OK**.

The **Serena** perspective displays, showing the **Serena Explorer** navigation view.

## Switching Between Perspectives

Switch between multiple open perspectives in the workbench by clicking on a perspective button in the upper right of the workbench window.



- **Switch to the Serena perspective** from other perspectives by clicking on the **Serena** button in the upper right of the workbench window.
- **Switch to the Java perspective** from the Serena perspective or elsewhere by clicking on the **Java** button.

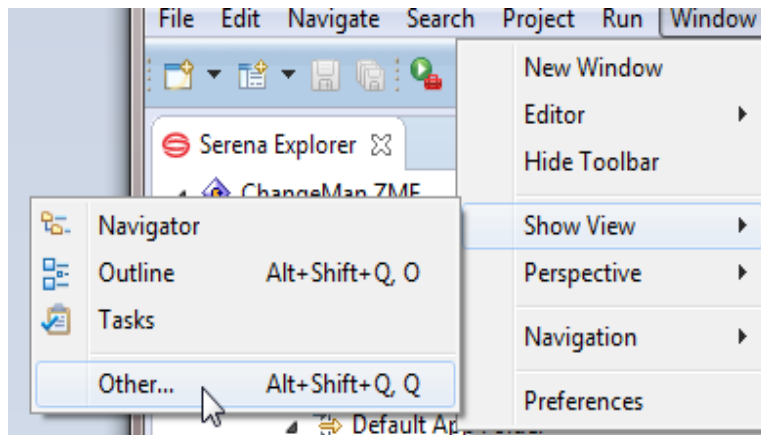
- **In RDz only:**
  - **Switch to the RDz Remote System Explorer perspective** from the Serena perspective or elsewhere by clicking on the **Remote System Explorer** button.
  - **Switch to the RDz z/OS Projects perspective** from the Serena perspective or elsewhere by clicking on the **z/OS Projects** button.

## Showing the Serena Explorer View

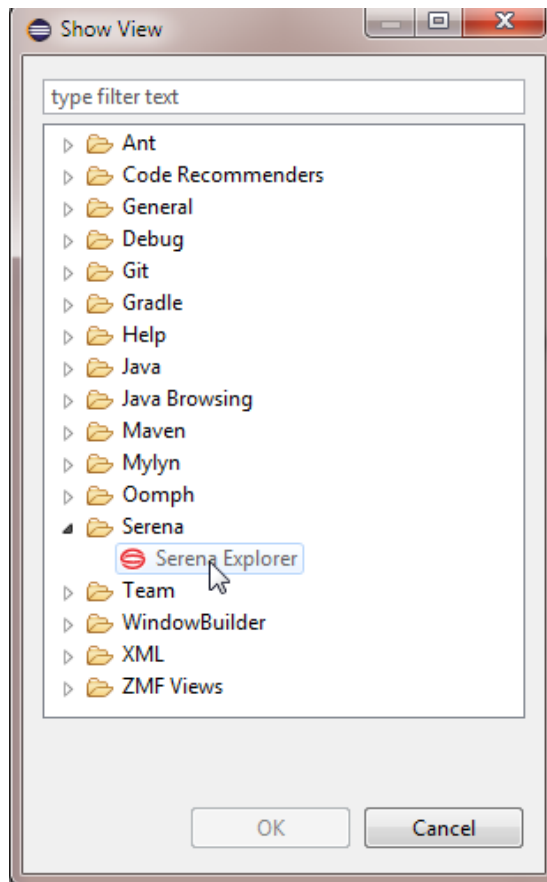
Within the Serena perspective, you can show or hide the **Serena Explorer** as needed to navigate among resources or to free screen space to work with one or more files.

To show the **Serena Explorer** view in any perspective where it is not displayed:

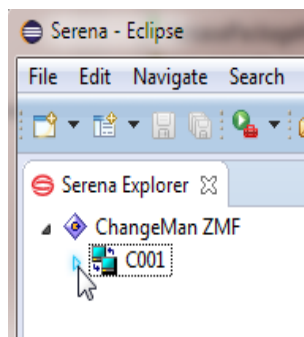
- 1 Select the area of the workbench where you want to display the Serena Explorer.
- 2 From the workbench **Window** menu, select **Show View | Other**.




- 3 When the **Show View** window displays, open the **Serena** folder and select **Serena Explorer**, then click **OK**.



- The **Serena Explorer** navigation view opens in the workbench window with the ChangeMan ZMF node collapsed. Click on the triangle to the left of this node to expand it and view available ZMF server connections.



## Hiding the Serena Explorer View

To hide the Serena Explorer view, click the Close button (  ) in the upper right of the **Serena Explorer** tab.

## Filtering Views in Serena Explorer

All resources shown in the **Serena Explorer** view are filtered via the use of folders. The default folders show all resources of a given type associated with your TSO user ID on the

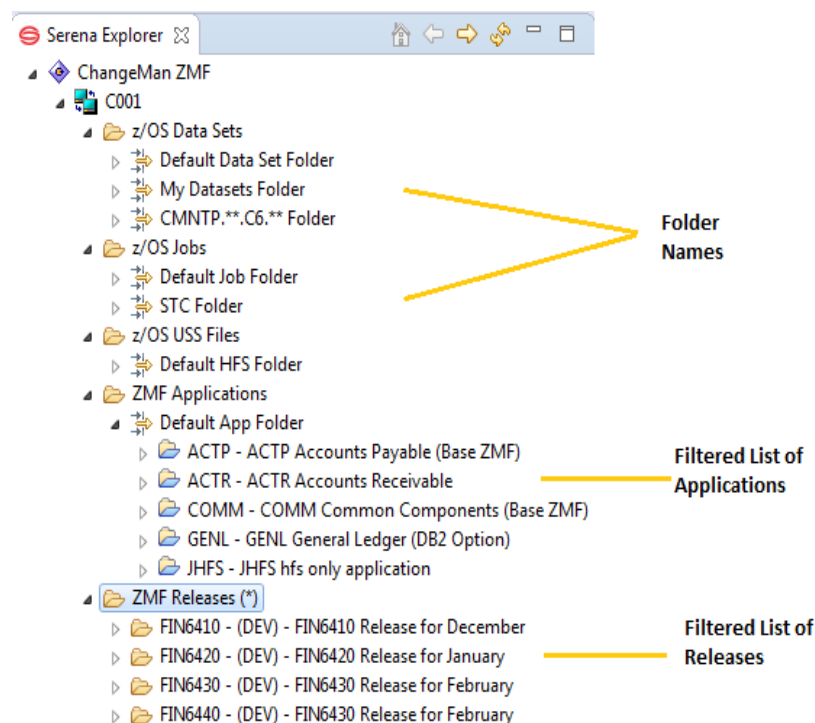
mainframe. Additional folders may be created, deleted, or modified at any time in ZMF for Eclipse.

For z/OS data sets, z/OS jobs, and z/OS Unix System Services (USS) Hierarchical File System (HFS) libraries, each filtered navigation view is represented as a directory node under the resource type. Expand a folder, then expand a filter node to access the libraries or jobs that meet the filter criteria. The same library or job may appear in multiple filter views.

For ZMF applications, multiple folders may be defined at a time, and each folder can have a filter applied. Expand a folder, then an application node to access the baseline libraries, change packages, and promotion libraries that you are authorized to work with.

For ZMF Releases, only one filter may be defined at a time, so the filtered releases are listed directly under the **ZMF Releases** node. Only one filter is active, so no filter nodes appear under the **ZMF Releases** node in **Serena Explorer** rather the current active filter which appears after the Label, for example ZMF Releases (F\*)

For example:



Create, delete, or modify filters from the contextual menu of the relevant folder. See the following topics for step-by-step instructions:

- [Filtering z/OS Data Sets](#)
- [Filtering z/OS Unix HFS Files](#)
- [Filtering z/OS Jobs](#)
- [Filtering ZMF Applications](#)

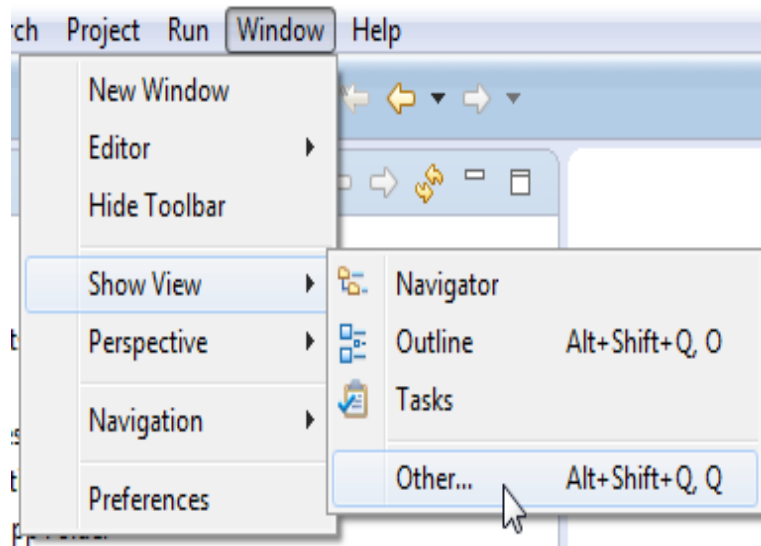
## Opening ZMF Table Views

ZMF for Eclipse provides a number of ZMF-specific, tabbed table views for the compact display of repository information. Many of these are shown automatically in the **Serena** perspective when you invoke a function from a contextual menu. But any of them can be requested on demand. Some, like the **ZMF Notifications** view, are frequently kept open at all times.

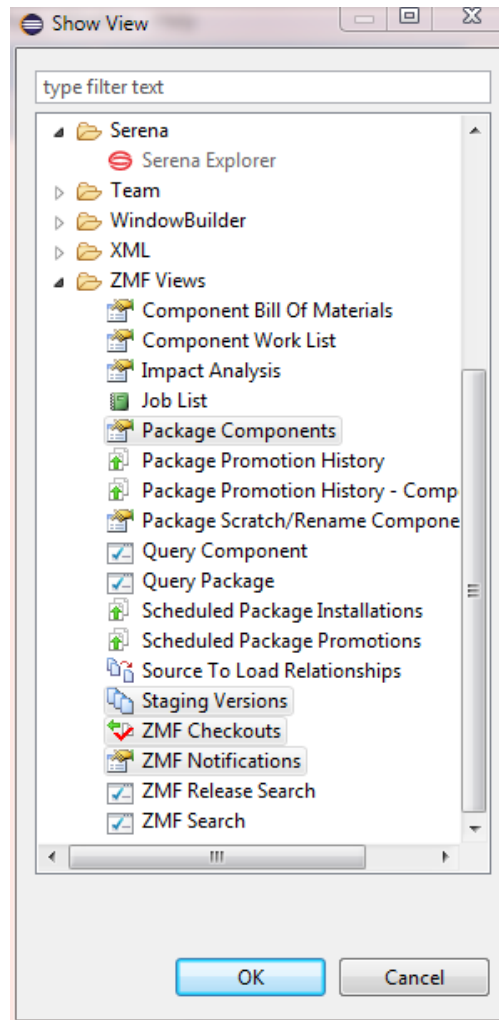
ZMF table views can be opened in any perspective.

To display one or more ZMF table views in the **Serena** perspective, perform the following steps.

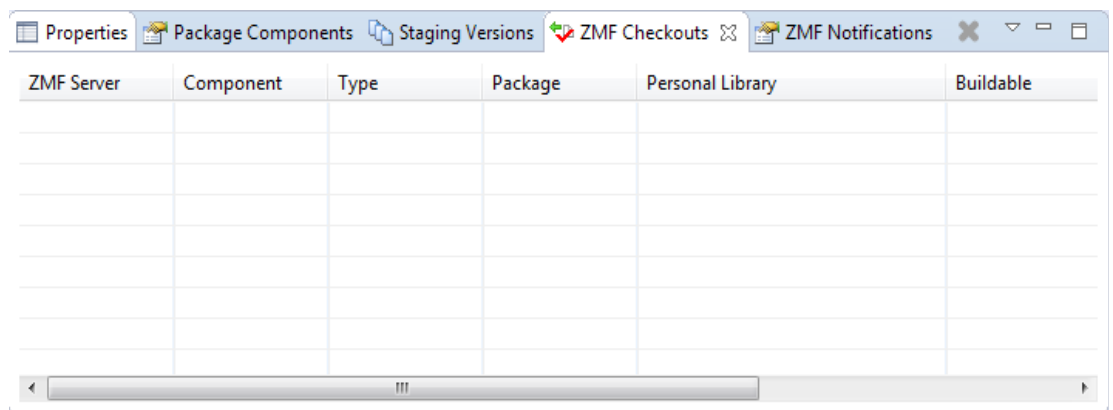
- 1 Switch to the **Serena** perspective.
- 2 From the workbench **Window** menu, select **Show View | Other**.



- 3 When the **Show View** dialog box displays, type "ZMF" in the filter text box or scroll to the bottom of the view navigation tree to locate the **ZMF Views** folder.
- 4 Expand the **ZMF Views** node in the tree. A list of ZMF table views is shown below it.



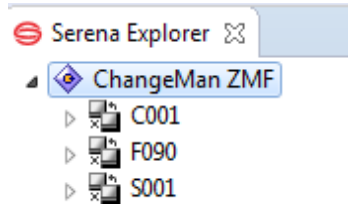
- 5 Select one or more ZMF table views as desired, then click **OK**.
- 6 The selected table views display together, each with its own tab, in the lower right pane of the perspective. No data will be displayed at this time.





# Accessing ChangeMan ZMF Repositories

ZMF for Eclipse connects your workbench development environment to one or more mainframe software repositories managed by ChangeMan ZMF. Repository servers are listed in the **Serena Explorer** view under the top-level **ChangeMan ZMF** node.



Individual repository names are assigned during connection setup and need not match the site names defined to ChangeMan ZMF on the mainframe. See the *ChangeMan ZMF for Eclipse Installation and Configuration* document for more information.

The following repository access functions are invoked from the **Serena Explorer** view:

- [Logging On to a ChangeMan ZMF Server](#)
- [Logging Off from a ChangeMan ZMF Server](#)
- [Accessing ChangeMan ZMF Repositories](#)

## Logging On to a ChangeMan ZMF Server

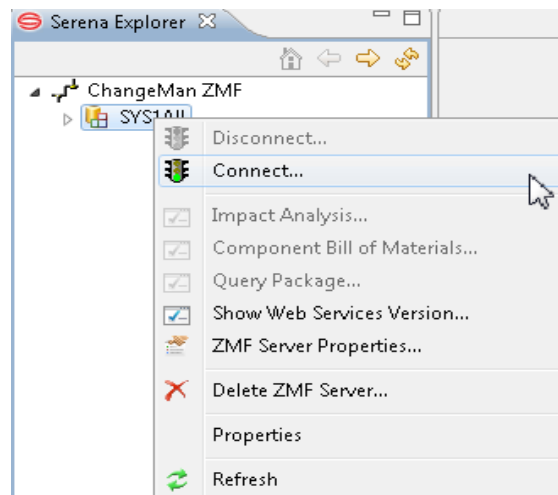
Automatic Logon Prompts

ZMF for Eclipse automatically prompts you to log on to a ZMF server when you expand the server's node in the **Serena Explorer** view. You may also be prompted to log in when switching between the z/OS native and z/OS Unix file systems.

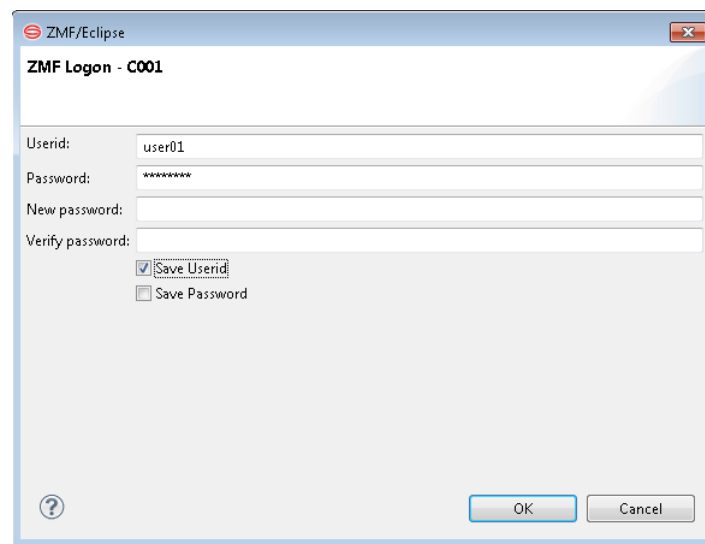
Contextual Menu Logon

In addition, you can log on from a contextual menu at any time. To log on to a ZMF server from a contextual menu:

- 1 Open the **Serena** perspective in the workbench.
- 2 In the **Serena Explorer** navigation view, right-click on the desired ZMF server to bring up its contextual menu, then select **Connect**.



- 3 The **ZMF Logon** dialog displays the name of the server you are accessing and prompts for your mainframe logon credentials.



Type your mainframe user ID and password in the **Userid** and **Password** fields. Password phrases (as well as standard passwords) are supported.



**NOTE** You should not enter a value in the **New Password** and **Verify New Password** fields unless you wish to change your password on the mainframe.

If you wish to save your mainframe logon credentials on your local computer for automatic reuse with this dialog, you can check **Save Userid**, **Save Password**, or both.



**IMPORTANT!** Do NOT save your mainframe logon password on a shared machine. Check **Save Password** only if you are the sole authorized user of your PC and access to the PC is protected. If **Save Userid** or **Save Password** is not allowed for your installation, the selection is ignored.

- 4 Click **OK** to log on to the host.
- 5 If logon is successful, you will see the expansion of the selected ZMF Server to show z/OS Datasets, z/OS Jobs, z/OS USS Files, ZMF Applications and (if ERO is licensed) ZMF Releases.



**TIP** If you get an error message at logon and an explicit retry from the contextual menu does not help, restart your Web application server (Tomcat or WebSphere Application Server) in Windows from **Start | Control Panel | Administrative Tools | Services**. Then try to log on again.

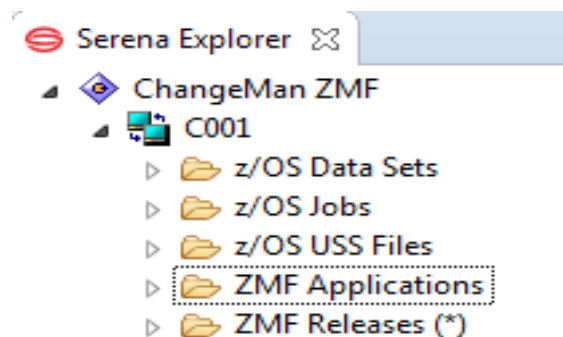
## Logging Off from a ChangeMan ZMF Server

Logging off from a ChangeMan ZMF repository is very similar to logging on.

- 1 In the **Serena Explorer** navigation view, right-click on the desired ChangeMan ZMF site to bring up its contextual menu, then select **Disconnect**.
- 2 The expansion of the ZMF Server will disappear.

## z/OS Resources Available on the Server

After successful logon, the **Serena Explorer** navigation view shows five subordinate nodes for the ZMF server. Each node represents a different group of mainframe resources.



- **Data Sets** — Shows personal development libraries associated with your TSO user ID in the z/OS native file system on the ZMF server.  
See the following topics for details:
  - ["Working with z/OS Data Set Libraries" on page 37](#)
  - ["Working with z/OS Data Set Members" on page 46](#)
- **z/OS Jobs** — Shows output from all jobs (such as compile jobs) that you submitted to JES for execution on the mainframe and which are still held in the job queue.  
See ["Working with z/OS Jobs" on page 55](#) for details.
- **z/OS USS Files** — Shows personal development libraries associated with your TSO user ID in the z/OS Unix System Services (USS) Hierarchical File System (HFS).  
See the following topics for details:
  - ["Working with z/OS Unix HFS Folders" on page 50](#)

- "Working with z/OS Unix HFS Files" on page 53
- **ZMF Applications** — Folders show software repository resources in ChangeMan ZMF, sorted by application ID. Only applications that you are authorized to view are shown.  
See the following topics for details:
  - "Working with ZMF Applications" on page 57
  - "Working with ZMF Packages" on page 62
  - "Working with ZMF Components" on page 67
- **ZMF Releases** — Shows releases (only if licensed for ERO) in ChangeMan ZMF, sorted by release name and optionally filtered by chosen filter.  
See the following topics for details:
  - "Working with ZMF Applications" on page 57

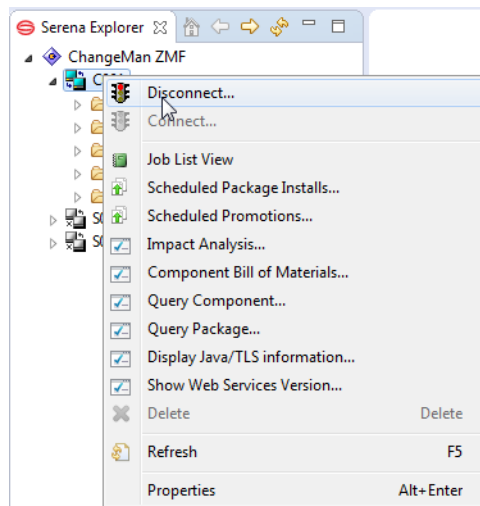


**NOTE** If one of the mainframe resource nodes does not appear for your ZMF server, that resource type may not be enabled in the ZDDOPTS command library for ZMF. If you need that node enabled in ZMF for Eclipse, ask your ZMF administrator.

## ZMF Server Functions

ChangeMan ZMF repository functions with potentially global scope can be requested at the ZMF server level after logon. Like the basic logon and logoff functions, they are invoked from the server's contextual menu.

ZMF Server  
Contextual Menu



ZMF Server  
Functions

The following ZMF functions are available at the ZMF server level:

- **Job List View** - Displays a tab with a view of current jobs on the ZMF server. Job Name, Owner and ZMF Server can be changed to alter the view.
- **Impact Analysis** — The **Impact Analysis** function identifies all the higher-level software components that will be affected by changes to one or more subordinate, reusable software modules (such as a subroutine). An impact analysis can span multiple applications. It is the inverse of the **Component Bill of Materials** function.

See ["Component Impact Analysis" on page 168](#) for step-by-step instructions.

- **Component Bill of Materials** — The **Component Bill of Materials** function identifies all the subordinate, reusable software modules (such as subroutines) referenced by one or more higher-level software components. A bill of materials query can span multiple applications. It is the inverse of the **Impact Analysis** function.

See [Chapter 5, "Component Bill of Materials" on page 165](#) for step-by-step instructions.

- **Query Component** — The **Query Component** function lists information about selected components, which is displayed in the **Query Component** view.

See ["Query Component Functionality" on page 171](#) for details.

- **Query Package** — The **Query Package** function lists all change packages on the server that meet your search criteria. A package query can span multiple applications.

See ["Querying Packages" on page 228](#) for step-by-step instructions.

- **Display Java/TLS Version** — Displays version information about Java and TLS, including Java Version, Java Home, available TLS Protocols, and available TLS Cipher Suites.

- **Show Web Services Version** — Displays the version of ZMF Web Services used by your web application server to connect to the selected ZMF server.

- **Properties** — Displays server properties such as Name, Address, ZMF and XCH TCP/IP ports associated with the selected ZMF server.

Refer to the *ChangeMan ZMF for Eclipse Installation and Configuration* document for more information.

## Working with z/OS Data Set Libraries

### Navigating Data Sets

**Data Sets Node** The **Serena Explorer** view provides access to data sets in the native z/OS file system under the **Data Sets** node. These data sets are accessed using z/OS file management outside the control of the ChangeMan ZMF software repository. For example, your PDS personal development libraries are listed here.

**Default Data Set Filter** When you expand the **Data Sets** node in the **Serena Explorer** view, a list of your currently defined data set viewing filters displays. If you have not defined any custom viewing filters, only the **Default Data Set Filter** node is shown. The **Default Data Set Filter** shows all data sets with a High-Level Qualifier (HLQ) equal to your TSO user ID.



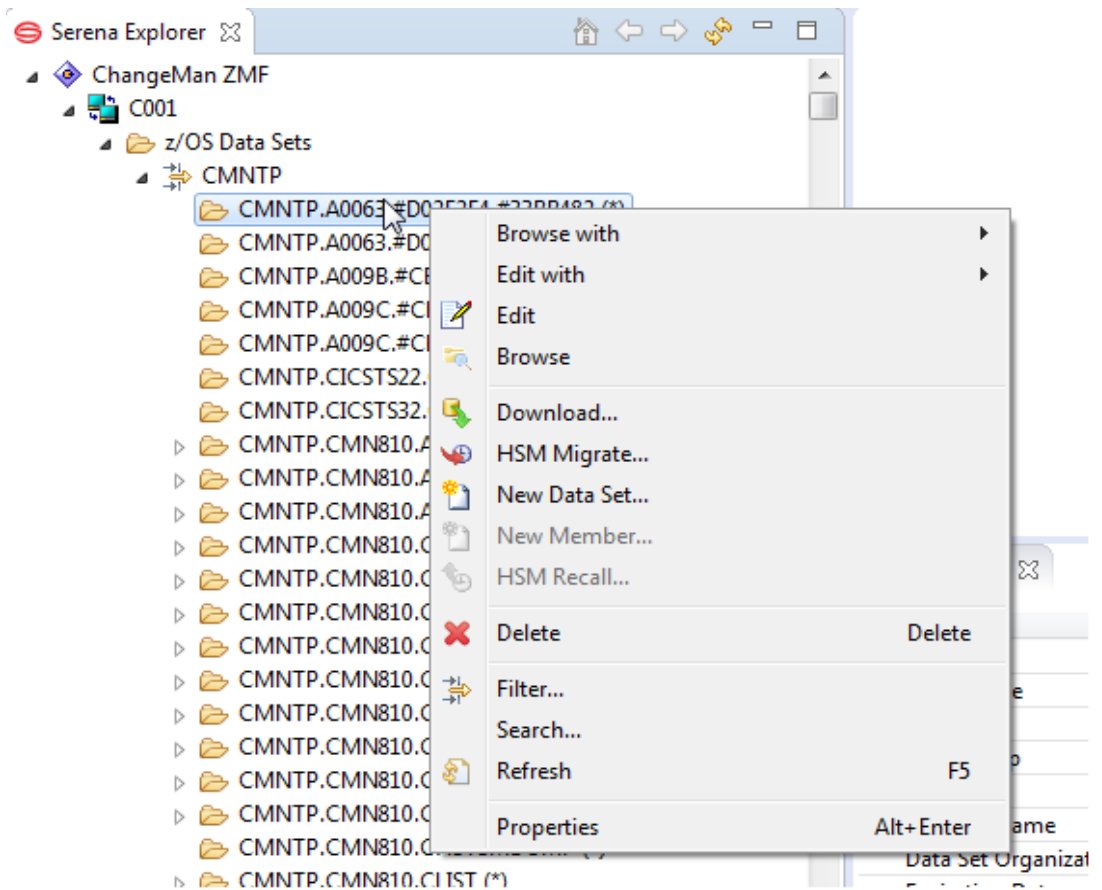
**TIP** Until you become familiar with the interrelationships between filters in ZMF for Eclipse and filters in the native workbench perspectives, Serena suggests you leave the default filters as-is and create new filters as needed for more selective views.

### z/OS Data Set Library Functions

The functions available for data set libraries differ from those for individual data set members. Contextual menus for libraries and members differ accordingly. The current properties of a selected object also affect the contents of a contextual menu.

Right-click on a data set name to bring up the data set library contextual menu. For example:

#### z/OS Data Set Contextual Menu



#### ZMF Functions for z/OS Data Sets

The following mainframe functions are available in the workbench for data sets as a whole (the first 4 are only available for sequential datasets):

- Browse with - gives you a dialog to choose what tool to use to browse
- Edit with - gives you a dialog to choose what tool to use to edit
- Edit - invoke the default text editor to edit
- Browse - invoke the default text editor to browse
- [Downloading a Data Set](#)
- [Migrating a Data Set Offline](#)
- [Allocating a New Data Set](#)
- [Creating a New Data Set Member](#)
- [Recalling a Migrated Data Set](#)
- [Deleting a Data Set](#)
- Filter
- [Searching](#)
- [Refreshing a Data Set](#)

For functions that apply to individual data set members, see ["Working with z/OS Data Set Members" on page 46](#).



**TIP** The order of data set names is different to that seen in ISPF due to the differences between EBCDIC and ASCII sort orders.

If the data set you're looking for is not shown, you may need to modify or create a data set viewing filter. See ["Filtering z/OS Data Sets" on page 77](#) for details.

## Refreshing a Data Set

The **Refresh** option on the contextual menu for a z/OS data set refreshes the list of members in the data set based on the current members present in the dataset.

## Allocating a New Data Set

The **New Data Set** option on the contextual menu for a z/OS data set brings up the **Allocate Data Set** dialog:

Field	Value
Data Set Name:	
Data Set Type:	PDS
Space Units:	TRK
Record Format:	FB
Primary quantity:	2
Record Length:	80
Secondary quantity:	5
Block Size:	6000
Directory Blocks:	
Data Class:	
Generic unit:	3390
Storage Class:	COMMON
Volume serial:	SRSM31
Management:	STANDEF
Extended Attributes:	

If you right-clicked on the name of a data set or data set that resides on an Extended Attribute Volume (EAV) to bring up the z/OS Data Set contextual menu, the fields of the **Allocate Data Set** dialog are initially populated with the saved preference values for the attributes. Fields of the **Allocate Data Set** dialog are:

- Data Set Name
- Data Set Type
- Record Format

- Record Length
- Block Size
- Data Class
- Storage Class
- Management
- Space Units
- Primary Quantity
- Secondary Quantity
- Directory Blocks
- Generic Unit
- Volume Serial
- Extended Attributes

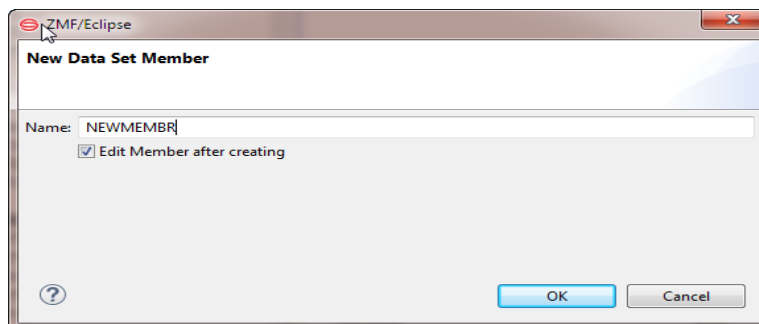
Fill in/overtyping the values for these fields as appropriate for the data set that you are allocating and click OK to allocate the data set. Note that your userid must have the required authority to create the dataset requested, otherwise you will see an error message at the top of the Allocate Dataset window "SER9008E Unauthorized access (ALTER): *datasetname*".

**Note** that this function enables you to set extended attribute values for a data set that resides on an EAV.

## Creating a New Data Set Member

You can create a new partitioned data set (PDS) member on the mainframe using the **New Member** function. To do this, perform the following steps.

- 1 From the parent data set's contextual menu, select the **New Member** option.
- 2 When the **New Data Set Member** window displays, enter the following information:
  - a **Name** — Type a member name of 8 characters or less in length, the case is folded to upper case.
  - b **Edit member after creating** — Check this box to open the PDS member in a workbench editor after the member is created.



- 3 Click **OK**. Note that if the name you choose already exists, the error "Name already exists" will appear in the heading panel of this window.





**NOTE** The new data set member is *not* a component in a ChangeMan ZMF repository. To add the member to a ZMF repository, use **Checkin** to check it in to a change package.

See "Checking In and Building a Data Set Member" on page 49 for details.

## Deleting a Data Set

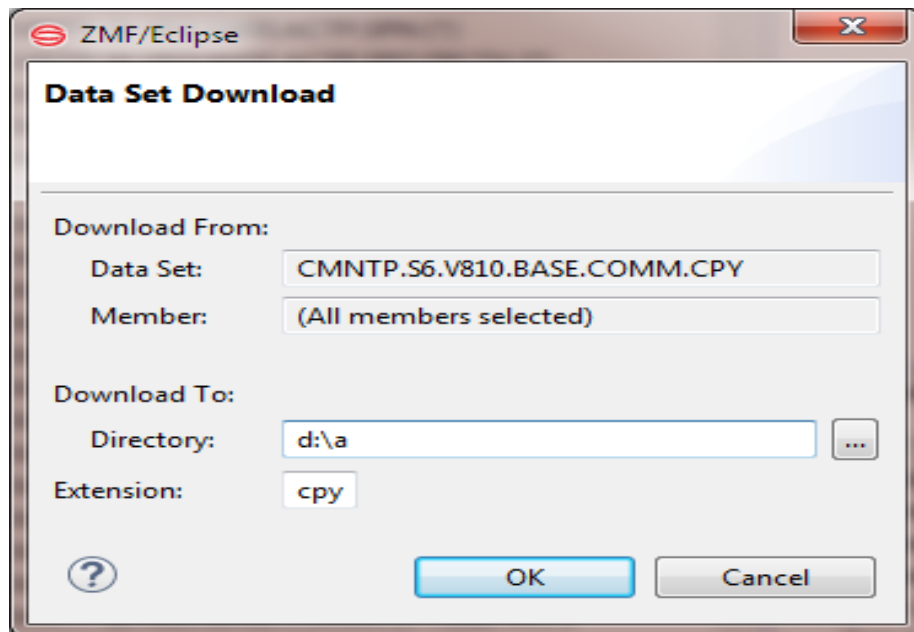
The **Delete** option on the contextual menu for a data set deletes the selected z/OS data set on the host. As an alternative, you may press the **Delete** key to delete a data set.

A dialog box asks the user to "Confirm Data Set delete" before executing the request. Click **OK** to proceed with deletion, or Cancel to leave alone.

## Downloading a Data Set

The **Download** option on the contextual menu for a data set copies all data set members as separate files to any directory you select. To download the members from a data set:

- 1 Select the **Download** option from the contextual menu for the data set library to be downloaded. The **Data Set Download** window displays.



- 2 The **Download From** section of the **Data Set Download** window is already populated and cannot be changed. Verify that this information is correct.
- 3 In the **Download To** section of the **Data Set Download** window, supply the desired target desktop directory information for the download.
  - a **Directory** — Supply the path to the target directory for the download.  
Click the ellipsis (...) button at right to navigate to another target directory.
  - b **Extension** — Type the filename extension that will be appended to all member file names created on the PC during the download. Note also that this is not limited to three characters, and case is as-is.

By default, the **Extension** text box shows the Windows file extension defined for the selected z/OS data set library type by library-to-file-type mapping parameters in the ZDDOPTS configuration library of ChangeMan ZMF. To override the default mapping, type a desired file extension in this field.

- 4 Click **OK**.
- 5 A progress bar displays in the bottom right corner of the window while data set members are downloading. If desired, the user can do the following:
  - Click the icon in the corner (hover the mouse and it will say "Shows background operations in Progress view") to observe progress as the download completes in the background.
  - Click the red box (hover the mouse and it will say "Cancel Operation") at the right of the progress bar to stop the download.
- 6 On completion, the progress bar will disappear.

## Migrating a Data Set Offline

After working with a data set, you may want to archive it offline to a Hierarchical Storage Management (HSM) system such as a tape library, rather than delete it. The **HSM Migrate** function lets you do this. The offline copy can be retrieved if needed.

To migrate a data set offline:

- 1 Select **HSM Migrate** from the contextual menu for the data set to be migrated.
- 2 When the confirmation dialog (Are you sure you want to migrate *dsname*) displays, verify that the data set name is correct.
  - a Click **OK** to proceed with the migration.

or

  - b Click **Cancel** to exit without migrating the data set.
- 3 A dialog notifies you when data set migration is initiated with a return code, reason code and a message SER4133I HMIGRATE issued for *datasetname*. Click **OK**.

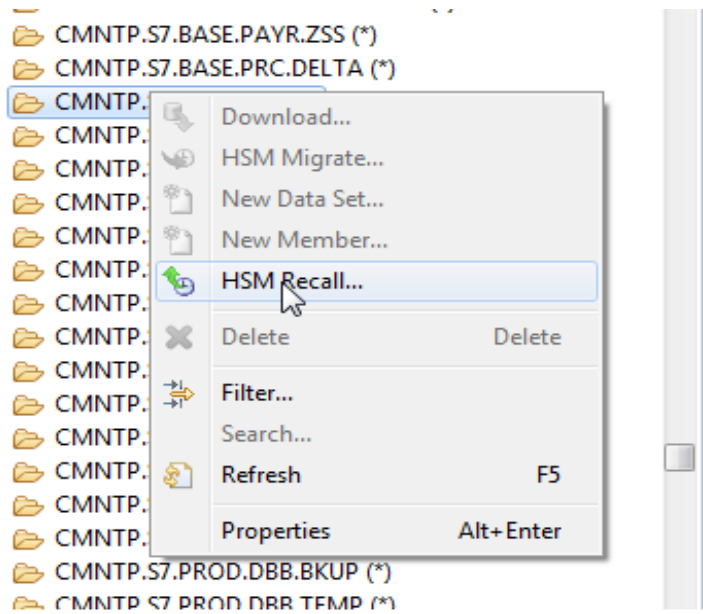
## Recalling a Migrated Data Set

Data Sets that are unused for a long period are often archived automatically to an offline Hierarchical Storage Management (HSM) system such as a tape library. ZMF for Eclipse, provides an option to recall such data sets from HSM storage to online storage.

Recalling a migrated data set can take several minutes even with an automated HSM system. If manual intervention is required, the recall process takes longer. ZMF for Eclipse allows you to keep working on other things while the recall is processed in the background.

To recall a migrated data set to online storage, perform the following steps.

- 1 Select **HSM Recall** from the contextual menu for the data set to be recalled.



- 2 When the confirmation dialog displays, verify that the data set name is correct. If the dataset is on tape (ML2) the context menu will say "HSM Recall from Tape...".
  - a Click **OK** to proceed with the recall. If the dataset is on ML2 (i.e. on Tape) it will also warn that "**migrated to Tape and may take extended time to recall**".
  - b Click **Cancel** to exit without recalling the data set.
- 3 A dialog notifies you the data set recall request(s) was submitted. Click **OK**.

## Searching

The Search facility enables you to search for the following ZMF-searchable objects:

- Data sets.
- Data set members.
- Data set filters.
- HFS filters.
- HFS files.
- HFS folders.
- Baseline data sets.
- Staging data sets.

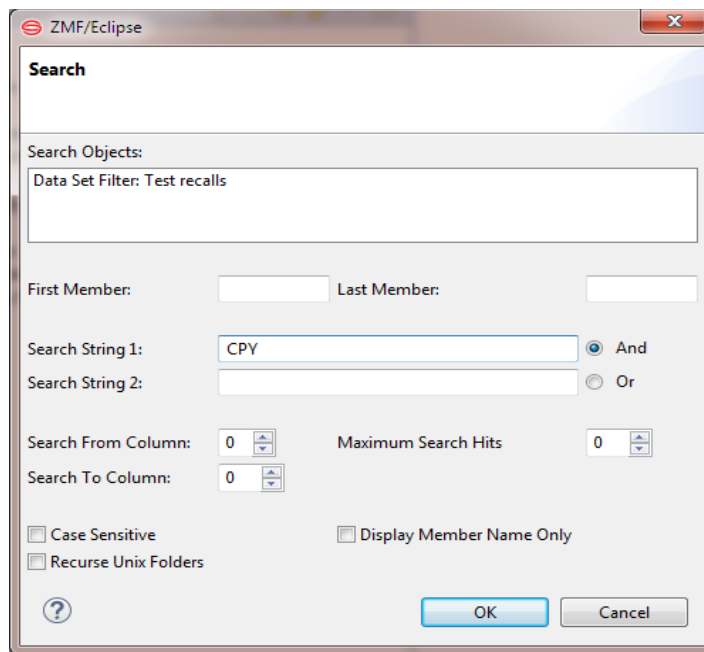
You can search any of these objects for any character string. You can further specify:

- Whether the search is case sensitive.
- The record positions to examine for the string.
- Two strings with a boolean OR or AND relationship.

To request a search:

- Right-click on the desired object in the Serena Explorer perspective. (A data set is chosen in this example.)
- Select **Search** from the pop-up menu. The **Search** dialog appears.

### Search Dialog



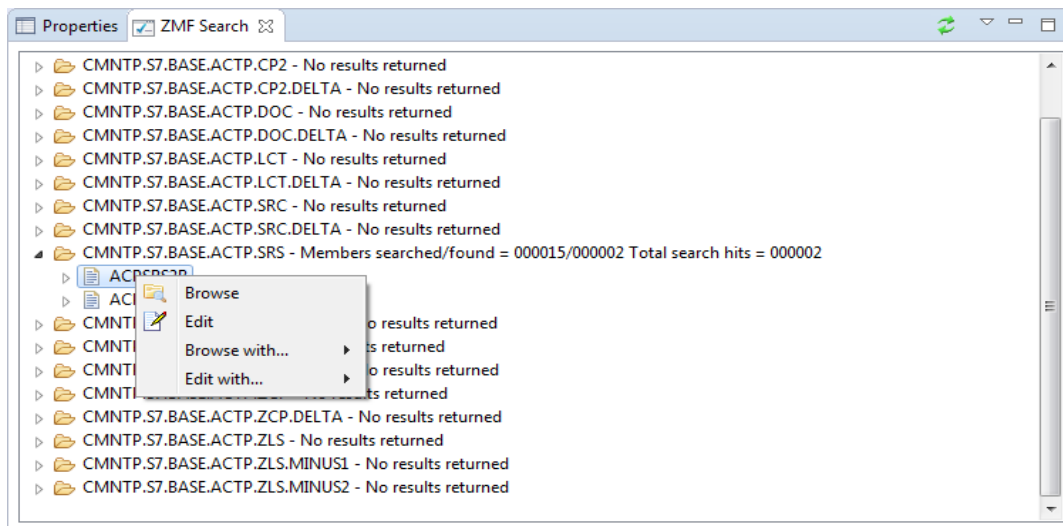
If data sets are migrated you will see a dialog that asks if you want to recall and process the data set (Yes), Skip it (No), or Cancel to stop the operation at that point. Results found to that point will be displayed if you click cancel. You can observe progress either in the window or you can send it to background and view in the progress pane if you have it open. You can click on the icon at the bottom right similar to the download process.

Fields on the Search dialog are:

Field	Description
First Member Last Member	<p>Type the First Member name and Last Member name to specify a range of members to search.</p> <ul style="list-style-type: none"> <li>■ To search all members, leave the First Member and Last Member fields blank.</li> <li>■ To search a single member, type its name in the First Member field. Leave the Last Member field blank.</li> <li>■ To limit the search to members that match a pattern, type a pattern (such as ABC*) in the First Member field. ChangeMan ZMF for Eclipse ignores the Last Member field value.</li> <li>■ You can type a range, such as MEMBERA for the First Member and MEMBERZ for the Last Member even if the members do not exist in the data set. All members that fall within the specified name range will be searched.</li> </ul>
Search String 1 Search String 2	<p>Type the string or strings to search for. Do not enclose a string containing embedded blanks or nonalphabetic characters in single quotes. Enclose a string that contains leading or trailing blanks in single quotes.</p>
And/Or	<p>If you specify a value for Search String 2, specify if you want Search String 1 and Search String 2 to be logically ANDed or ORed.</p>
Search From Column Search To Column	<p>Type the starting (From) and ending (To) column positions to search in each record. Type 7 for column 7, 36 for column 36, and so on. Type 0 in the From and To Column fields to search all positions in the record.</p>
Maximum Search Hits	<p>Type the number of output records to display:</p> <ul style="list-style-type: none"> <li>■ 0 Display all records that match the search criteria.</li> <li>■ <i>n</i> Display up to <i>n</i> records that have been retrieved.</li> </ul> <p>This value determines the number of target records that are included in the displayed output. (It does not limit the number of record hits that search retrieves.)</p>
Case Sensitive	<ul style="list-style-type: none"> <li>■ Check this box to search for a string exactly as you typed it.</li> <li>■ Leave the box unchecked to find all occurrences of a string, whether in upper, lower, or mixed case.</li> </ul>

Field	Description
Display Member Name Only	<ul style="list-style-type: none"> <li>Check this box to display only the name of a member where a hit occurs.</li> <li>Leave the box unchecked to display the names of the members and the data that meet the search criteria.</li> </ul>
Recurse Unix Folders	Search the current folder and its subfolders for the specified search string.

### Search View Results



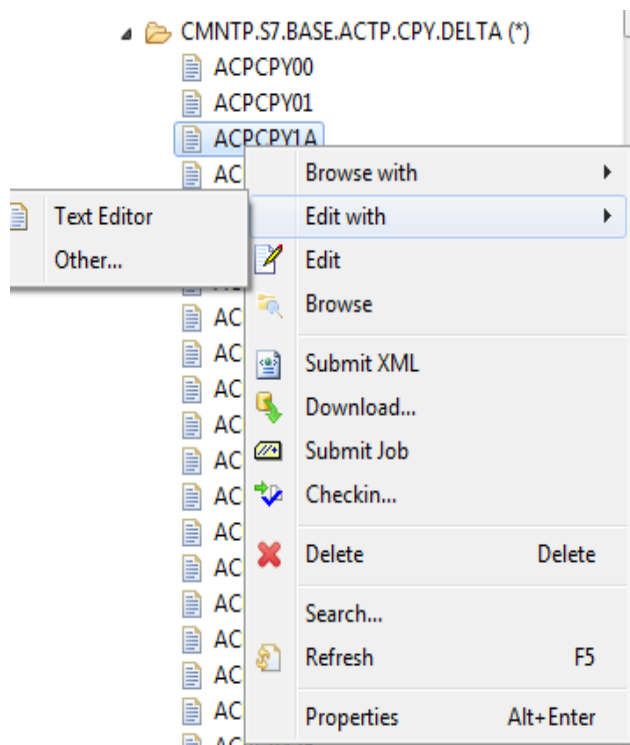
Results of the search are displayed. You can right-click on a member of file that is listed in the display to bring up a Browse/Edit menu:

- Select Browse or Browse with... to browse the selected member or file.
- Select Edit or Edit with... to edit the selected member or file.

## Working with z/OS Data Set Members

The **Serena Explorer** navigation view provides access to individual data set members as well as full data sets under the **Data Sets** node. Data set members are accessed using z/OS file management outside the control of the ChangeMan ZMF software repository.

To work with individual data set members, expand the **Data Sets** node, then the folder for an applicable data set filter, then the node for the data set library containing the member. Finally, right-click on the desired member to bring up its contextual menu.

Data Set Member  
Contextual MenuZMF Functions for  
Data Set Members

The functions available for individual data set members differ from those for data sets. The following ZMF functions are accessible from the data set member contextual menu:

- Browse with - gives you a dialog to choose what tool to use to browse
- Edit with - gives you a dialog to choose what tool to use to edit
- Edit - [Editing a Data Set Member](#) (or double click the member name)
- Browse - [Browsing a Data Set Member](#)
- Submit XML - [Submitting an XML Services Request](#)
- Download... - [Downloading a Data Set Member](#)
- Submit Job - [Submitting a Job for Execution](#)
- Checkin... - [Checking In and Building a Data Set Member](#)
- Delete - [Deleting a Data Set Member](#) (Del key)
- Search... - Search a member
- Refresh - [Refreshing a Data Set Member](#) (F5 key)
- Properties - Properties of a member (Alt-Enter keys)

## Editing a Data Set Member

The **Edit** option for a data set member on the z/OS host opens that member for update in a workbench editor. A tab in the workbench editors area displays the name of the member, while its contents appear in the main region of the tabbed pane. When you save changes in the editor, the data set member on the z/OS server is automatically updated (as long as you have the authority on z/OS to do so).

Copy and paste operations between an open z/OS data set and another desktop window, either inside or outside the workbench, is supported for compatible data types.

## Browsing a Data Set Member

The **Browse** option on the contextual menu for a data set member opens that member for read-only access in a workbench editor. A tab in the editors area of the workbench displays the name of the member, while the contents appear in the main body of the tabbed pane.



**TIP** Because **Browse** is a read-only function, you can browse a data set member's contents even if it is locked against updates. If you use the default editor, you can see that it is browse by the RO in the header, i.e. `//C001/RO/dsname/membername.type`

## Downloading a Data Set Member

The **Download** option on the contextual menu for a data set member downloads a copy of the selected member as a single file to any directory you select.

To download a data set member, perform the following steps.

- 1 Select the **Download** option from the contextual menu for the member to be downloaded. The **Data Set Download** window displays.
- 2 The **Download From** section of the **Data Set Download** window is already populated with the dataset name and member and cannot be changed. Verify that this information is correct.
- 3 In the **Download To** section of the **Data Set Download** window, supply the desired target desktop directory information for the download.
  - a **Directory** — Supply the path to the target directory for the download.  
Click the ellipsis (...) button at right to navigate to a target directory.
  - b **Extension** — Type the filename extension that will be appended to all member file names during the download.  
  
By default, the **Extension** text box shows the Windows file extension defined for the selected z/OS member library type by library-to-file-type mapping parameters in the ZDDOPTS configuration library of ChangeMan ZMF. To override the default mapping, type a desired file extension in this field.
- 4 Click **OK**.
- 5 A window will display progress, or you can select run in background, then a progress bar displays in the bottom right corner of the window while the member is downloading. If desired, you can do the following:
  - Click the icon in the corner (hover the mouse and it will say "Shows background operations in Progress view") to observe progress as the download completes in the background.
  - Click the red box (hover the mouse and it will say "Cancel Operation") at the right of the progress bar to stop the download.
- 6 On completion, the progress bar will disappear.



## Checking In and Building a Data Set Member

**Component Checkin** The **Checkin** option on the contextual menu for a data set member enables you to check in a component from a personal development library on the z/OS server to a change package managed by ChangeMan ZMF. The **Checkin** function in ZMF for Eclipse is similar to the **Stage** function in the ISPF interface to ChangeMan ZMF.

**Component Builds** If the checked in resource is buildable, ZMF for Eclipse automatically prompts you for build job specifications at checkin. Default job cards and build jobs for the associated application are displayed in the prompt. You can omit the build step if desired.

Checkin from a development library is permitted only if the following criteria are met:

- The library type of the member must match a library type defined for the application.
- The component in the change package must not be locked by another TSO user ID.
- The component in the change package must not be a generated component.
- The package must be in DEV status and its install date must be today's date or later.



**CAUTION!** If another version of the data set member already exists in the target change package but is not locked, **it will be overwritten**. No warning message is displayed.



**IMPORTANT!** See "[Checking In a Component](#)" on page 134 for step-by-step instructions concerning **Checkin**.

## Submitting a Job for Execution

**Executing Mainframe Jobs** Executable data set members on the mainframe may be submitted to z/OS for execution using the **Submit Job** option on the contextual menu for a data set member.

Output from the executing job is saved to the z/OS JES (Job Entry Subsystem) spool and can be viewed under the **z/OS Jobs** node in the **Serena Explorer** view.



**NOTE** There is no automated build support or job card insertion associated with the data set **Submit Job** function, as the job is being submitted from a personal development library outside the control of ChangeMan ZMF. You must include all required job cards and JCL statements with the submitted job.

## Submitting an XML Services Request

**XML Services Requests** Requests that invoke the ChangeMan ZMF XML Services API may be submitted for execution using the **Submit XML** option. Only single members or individual files may be submitted using this command. You can edit your XMLIN dataset member, to prepare it for submission, and save it, and then submit directly from the list of members.

**XML Services Replies** Output from the XML Services request is displayed (named as *membername.xml*) in an editor window in the Serena perspective of ZMF for Eclipse. The resulting data stream may be edited and saved on the desktop or in the repository as desired.

**XML Services Syntax** Refer to the *ChangeMan ZMF XML Services User's Guide* for information on coding ZMF XML Services requests.

## Deleting a Data Set Member

The **Delete** option on the contextual menu for a data set member deletes the selected member from the z/OS data set on the host. As an alternative, you may press the **Delete** key to delete a data set member.

A dialog box asks you if you to confirm the **Data Set Member delete** before executing your request. Click **OK** to proceed with deletion, or click **cancel**. The member list is refreshed immediately if you elect to delete.

## Refreshing a Data Set Member

The **Refresh** option on the contextual menu for a data set member refreshes the working copy of the data set member that is maintained by the workbench.

## Working with z/OS Unix HFS Folders

**z/OS USS Files Node** The **Serena Explorer** navigation view provides access to z/OS Unix System Services (USS) Hierarchical File System (HFS) folders and files under the **z/OS USS Files** node. Your HFS personal development libraries on the mainframe are listed here.

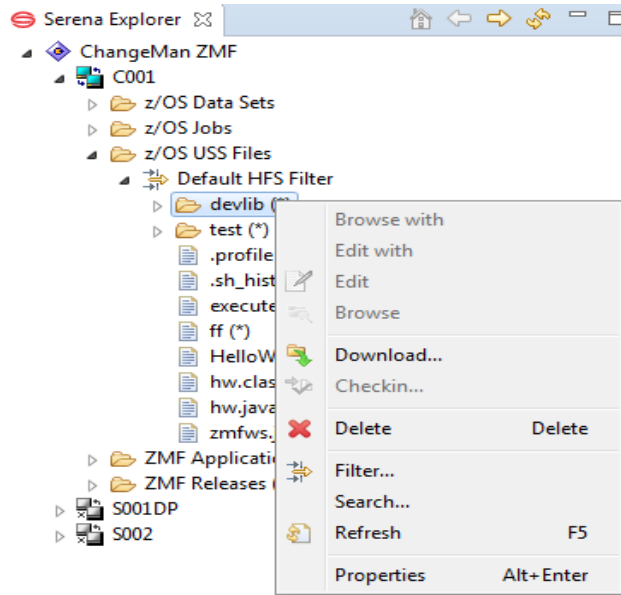
**Default HFS Filter** When you expand the **z/OS USS Files** node in the **Serena Explorer** view, a list of your currently defined HFS viewing folders displays. If you have not defined any custom viewing folders and associated filters, only the **Default HFS Filter** folder is shown.

Expand an HFS filter node to work with the HFS directories or folders that match the filter's selection criteria. By default, the **Default HFS Filter** shows all z/OS HFS folders in the user directory */u/userid/*, where *userid* is your TSO user ID on the mainframe. However, to minimize visual clutter, the top-level user directory */u/* and the next-level subdirectory named with your user ID are not actually shown.



**TIP** Until you become familiar with the interrelationships between filters in ZMF for Eclipse and filters in the native RDz perspectives, Serena suggests you leave the default folders and filters as-is and create new folders and filters as needed for more selective views.

Right-click on a folder name to bring up its contextual menu.

HFS Folder  
Contextual Menu

**NOTE** A Search function has been added to this contextual menu. Refer to ["Searching" on page 43](#) for a description of the Search function.

ZMF Functions for  
HFS Folders

The following functions are available for z/OS Unix HFS folders (that is, directories):

- [Refreshing a Unix HFS Folder](#)
- [Deleting a Unix HFS Folder](#)
- [Downloading a Unix HFS Folder](#)

## Refreshing a Unix HFS Folder

The **Refresh** option on the contextual menu for a z/OS HFS folder refreshes the list of subfolders and files in the folder. You may need to refresh this list in the client after creating a new file or folder on the z/OS server in another perspective.

## Deleting a Unix HFS Folder

The **Delete** option on the contextual menu for an HFS folder deletes the selected HFS folder and all its contents. As an alternative, you may press the **Delete** key.

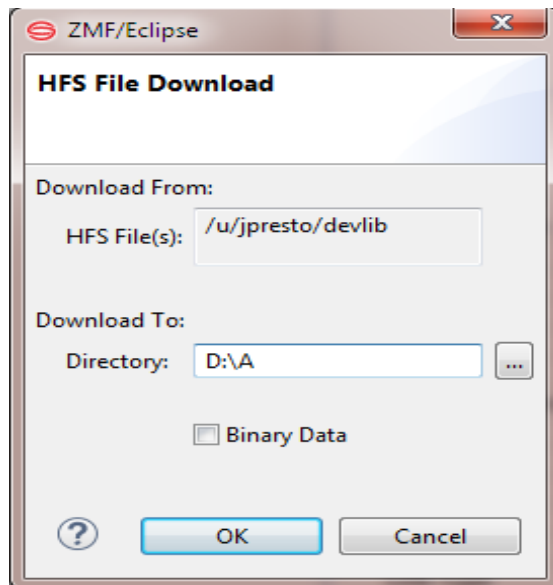
A dialog box asks you if you really want to delete the folder before executing your request. Click **OK** if you want to proceed with deletion. The delete will also delete any directories and files in the structure if they are present, and will go ahead successfully even if the user is using OMVS and is currently in one of those directories.

## Downloading a Unix HFS Folder

The **Download** option on the contextual menu for an HFS folder copies all subfolders and files in the folder to any directory you select on the desktop. The nested folder hierarchy is preserved.

To download the contents of an HFS folder to the desktop:

- 1 Select the **Download** option from the folder's contextual menu. The **HFS File Download** window displays.
- 2 The **Download From** section of the **HFS File Download** window is already populated and cannot be changed. Verify that this information is correct.



- 3 In the **Download To** section of the **HFS File Download** window, supply the desired target desktop directory for the download in the **Directory** text box.

Click the ellipsis (...) button to browse to and change to another target directory.

- 4 The **Binary Data** check box allows you to exclude binary files, such as executable code or graphics files, from EBCDIC-to-ASCII conversion on download. This setting applies to all subfolders and files within the downloaded folder.
  - **Do not** check this box if you are downloading source files, HTML or XML files, or other text files.
  - **Do** check this box if you are downloading compiled executables or graphics.



**IMPORTANT!** Do not mix text and binary file types in the same download.

- 5 Click **OK**.
- 6 A progress bar displays in the bottom right corner of the window while the download is running. If desired, you can do the following:
  - Click the icon in the corner (hover the mouse and it will say "Shows background operations in Progress view") to observe progress as the download completes in the background.
  - Click the red box (hover the mouse and it will say "Cancel Operation") at the right of the progress bar to stop the download.
- 7 On completion, the progress bar will disappear.

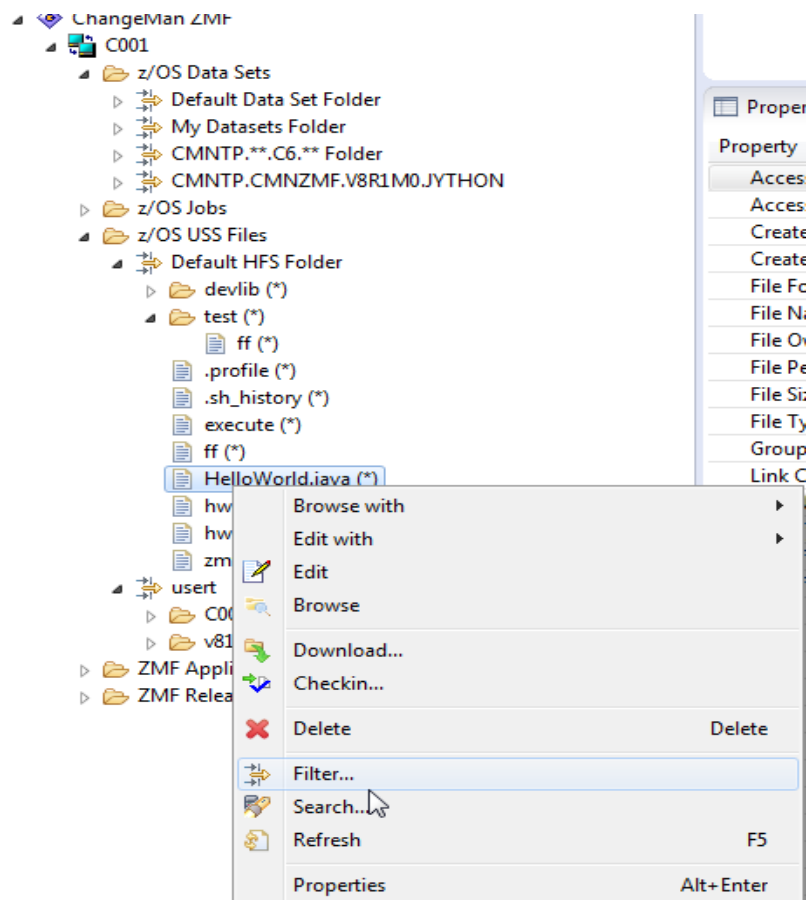
# Working with z/OS Unix HFS Files

To work with individual files in the z/OS Unix System Services (USS) Hierarchical File System (HFS), drill down the navigation tree hierarchy under **z/OS USS Files** in the **Serena Explorer** view until you locate the desired file. Right-click on the name of the file to bring up its contextual menu.

HFS File  
Function Summary

File functions accessible through this menu are **Browse with, Edit with, Edit, Browse, Download, Checkin** (to a ZMF change package), **Delete, Filter, Search, Refresh and Properties**.

HFS File  
Contextual Menu



ZMF Functions for  
HFS Files

The following functions are available for z/OS Unix HFS files:

- Browse with - Allows you to choose what editor to browse with.
- Edit with - Allows you to choose what editor to edit with.
- [Browsing a Unix HFS File](#)
- [Editing a Unix HFS File](#)
- [Downloading a Unix HFS File](#)
- [Checking In and Building a Unix HFS File](#)
- [Deleting a Unix HFS File](#)
- Filter - allows you to filter using wildcards.

- [Searching](#)
- [Refreshing a Unix HFS File](#)

## Refreshing a Unix HFS File

The **Refresh** option on the contextual menu for an HFS file refreshes the working copy of the data set member that is maintained by the workbench.

## Deleting a Unix HFS File

The **Delete** option on the contextual menu for an HFS file deletes the selected file on the mainframe. As an alternative, you may press the **Delete** key.

A dialog box asks you if you really want to delete the file before executing your request. Click **OK** to proceed with deletion.

## Downloading a Unix HFS File

The **Download** option on the contextual menu for an HFS file downloads a copy of the selected file to any directory you select on the desktop.

To download the contents of an HFS folder to the desktop:

- 1 Select the **Download** option from the folder's contextual menu. The **HFS File Download** window displays.
- 2 The **Download From** section of the **HFS File Download** window is already populated and cannot be changed. Verify that this information is correct.
- 3 In the **Download To** section of the **HFS File Download** window, supply the desired target desktop directory for the download in the **Directory** text box or click the ellipsis (...) button to browse to another target directory.
- 4 The **Binary Data** check box allows you to exclude binary files, such as executable code or graphics files, from EBCDIC-to-ASCII conversion on download. This setting applies to all subfolders and files within the downloaded folder.
  - **Do not** check this box if you are downloading source files, HTML or XML files, or other text files.
  - **Do** check this box if you are downloading compiled executables or graphics.
- 5 Click **OK**.
- 6 A progress bar displays while the file is downloading. If desired, you can do either of the following at any time during the download:
  - Click **Run in Background** to return to the Serena perspective while the download completes in the background.
  - Click **Cancel** to stop the download. If you have selected run in background and want to cancel, open the progress tab and click the red button on the right to cancel.

## Editing a Unix HFS File

The **Edit** option for an HFS file opens that file for update on the mainframe from a GUI workbench editor. A tab in the workbench editors area displays the name of the file, while its contents appear in the main region of the tabbed pane. When you save changes in the editor, the file on the z/OS server is automatically updated.

Copy and paste operations between an open z/OS file and another desktop window, either inside or outside the workbench, are supported for compatible data types (such as text).

## Browsing a Unix HFS File

The **Browse** option on the contextual menu for an HFS file opens the file for read-only access in a workbench editor. A tab in the editor area of the workbench displays the name of the member, while the contents appear in the main body of the tabbed pane.



**TIP** Because **Browse** is a read-only function, you can browse a file's contents even if it is locked against updates.

## Checking In and Building a Unix HFS File

Component  
Checkin

The **Checkin** option on the contextual menu for an HFS file enables you to check in a component from a personal development library on the z/OS server to a change package managed by ChangeMan ZMF. The **Checkin** function in Serena Explorer is similar to the "Stage from Development" function in ChangeMan ZMF.

Component  
Builds

If the checked in resource is buildable, ZMF for Eclipse automatically prompts you for build job specifications at checkin. Default job cards and build jobs for the associated application are displayed in the prompt. You can omit the build step if desired.

Checkin from a development library is permitted only if the following criteria are met:

- The filename extension must match a library type defined for the application.
- The component in the change package must not be locked by another TSO user ID.
- The component in the change package must not be a generated component.
- The package must be in DEV status and its install date must be today's date or later.



**CAUTION!** If another version of the data set member already exists in the target change package but is not locked, **it will be overwritten**. No warning message is displayed.



**IMPORTANT!** See ["Checking In a Component" on page 134](#) for step-by-step instructions concerning **Checkin**.

## Working with z/OS Jobs

The **Serena Explorer** navigation view provides access to z/OS job output, such as compiler listings or the SYSPRINT output from mainframe job execution, under the **z/OS**

**Jobs** node. The **z/OS Jobs** node is *not* used to submit jobs for execution. Mainframe jobs are submitted for execution through ZMF for Eclipse.

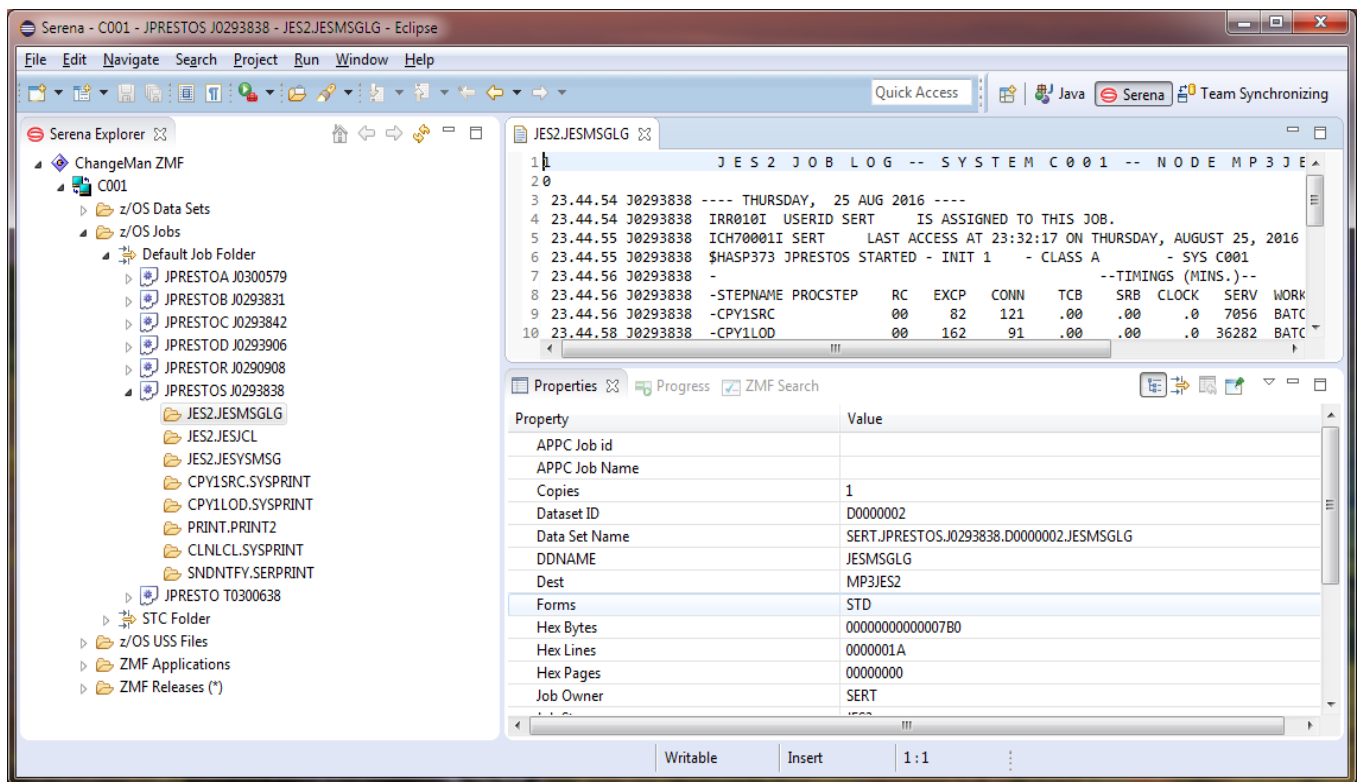
## Navigating Job Output

Job name and job owner determine where job output appears under the **z/OS Jobs** node. Output is listed by job name and organized by filters.

The **Default Jobs Filter** lists output for jobs that execute under your authority as the job owner or which include your user ID in the job name, as defined by the JCL in the job card used to submit the job. This is the typical situation for mainframe job execution.

Jobs which execute under the authority of other security entities require a custom job viewing filter to display under the **z/OS Jobs** node.

Expand the **z/OS Jobs** node to view a list of your job folders. Expand a folder to work with the jobs that match the folder's filter selection criteria. Expand a job node to see a list of viewable reports, listings, logs, and messages. Right-click on a node name to bring up its contextual menu. Double-click on an output file to browse it in a text editor.



## Browsing Job Output

To view the output of a job on a z/OS JES job queue in the Serena perspective:

- 1 In **Serena Explorer**, expand the **z/OS Jobs** node for the system whose job output you want to view. Then expand the node for the job filter containing the reports or listings of interest.
- 2 Expand the folder with the job name of interest to see a list of printable reports, listings, messages, and logs associated with the job.

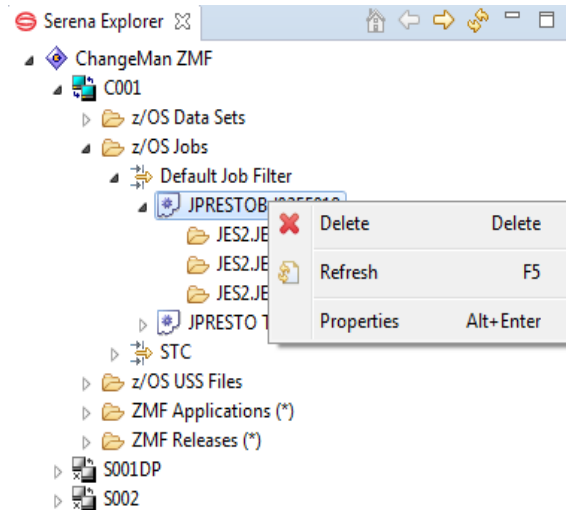


- 3 Double-click on a report, listing, or other output to browse it in a text editor.

## Canceling a Job and Deleting Job Output

To cancel a print job in the JES job queue and delete the associated job output, perform the following steps.

- 1 In **Serena Explorer** navigation view, expand the **z/OS Jobs** node for the server whose job output you want to view. Expand the node for the job filter containing the reports or listings of interest, then right-click on the name of the job you want to delete to bring up its contextual menu.



- 2 From the contextual menu, select **Delete**.
- 3 A dialog box gives you the option to confirm the **Delete** command or to cancel.
- 4 When the JES job has been purged from the spool, it also disappears from the list of jobs shown.

## Working with ZMF Applications

ChangeMan ZMF repository resources are shown under the **ZMF Applications** node in the **Serena Explorer** navigation view of the **Serena** perspective. Here, different ZMF functions can be invoked from the contextual menus for different resource types. When working with development libraries that reside on the mainframe, this is the preferred method of invoking ZMF functions in ZMF for Eclipse.

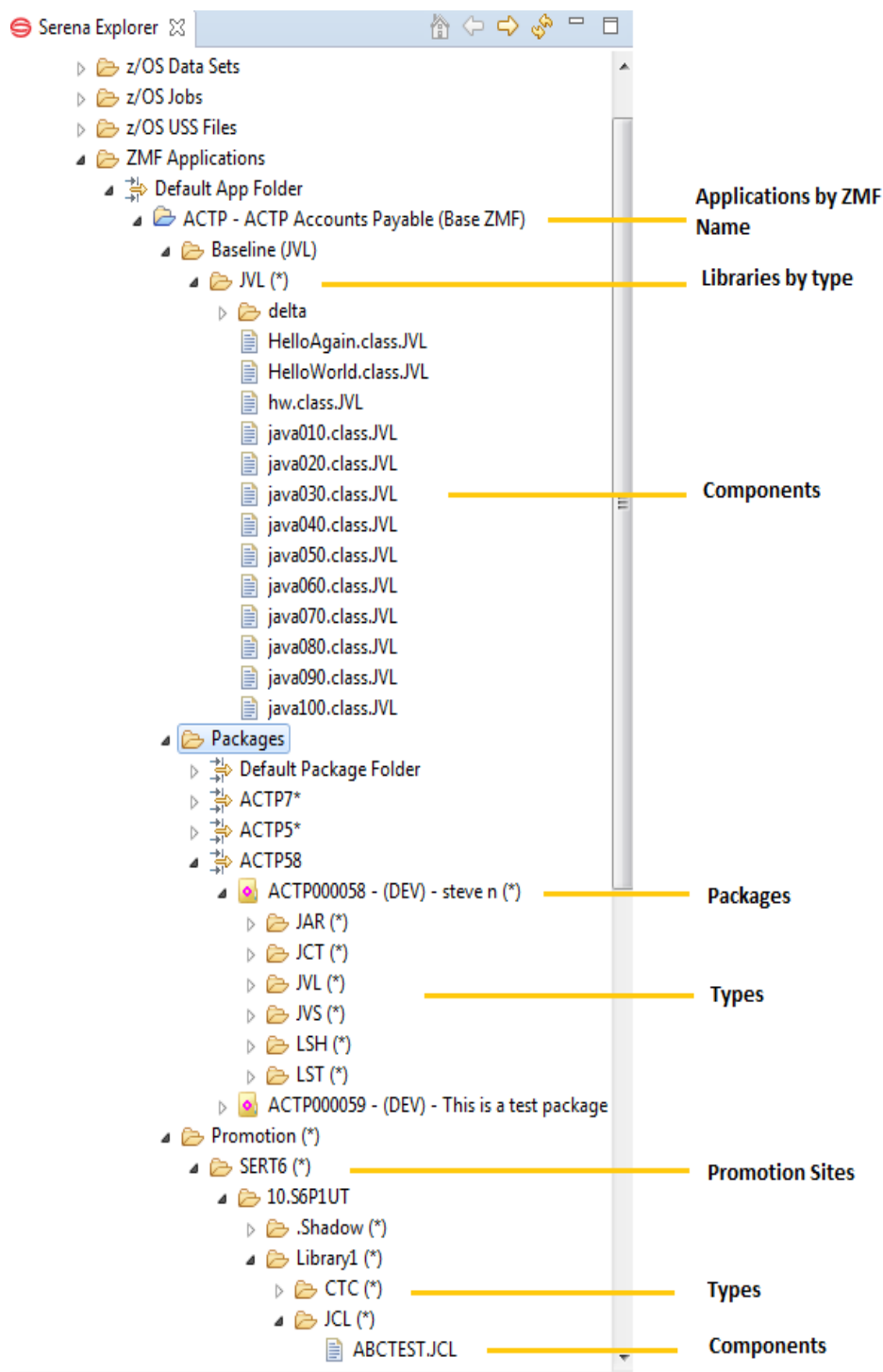


**TIP** When working with Java project resources that reside in a desktop workspace managed by Eclipse or RDz, the preferred method of invoking ZMF functions is from the **Package Explorer** navigation view in the Java perspective. The Team submenu of the contextual menu for the resource shows available ZMF functions.

See [Chapter 3, "Working with the Java Perspective" on page 95](#) for more information.

## Navigating ZMF Applications in Serena Explorer

ZMF Applications  
in Serena Explorer



ZMF repository resources in the **Serena Explorer** view are organized by ZMF application name first, then by collection type (baseline libraries, change packages, and promotion libraries), then by component type (source, load/binary, JCL, listing, and so forth — identified by mainframe library type in the case of z/OS native PDS libraries and by directory or folder in the case of z/OS Unix HFS files). Individual components are shown under the node for their type container.

## Navigation Hierarchy and Contextual Menus

Expand the **ZMF Applications** node to view a list of ZMF applications managed by the repository. Expand an application node to see associated resource containers for baseline libraries, change packages (and their associated staging libraries), and promotion libraries. Expand a resource container to see actual libraries by component type. Expand a library to see the individual components in that library. For all nodes, right-click on a node name to bring up its contextual menu.

## Browsing or Editing Components

Double-click on a component to open it in a workbench editor. Even compressed compiler listings stored in ZMF may be browsed in this way, as Serena supplies its own listing editor for Eclipse that decompresses listings on demand.

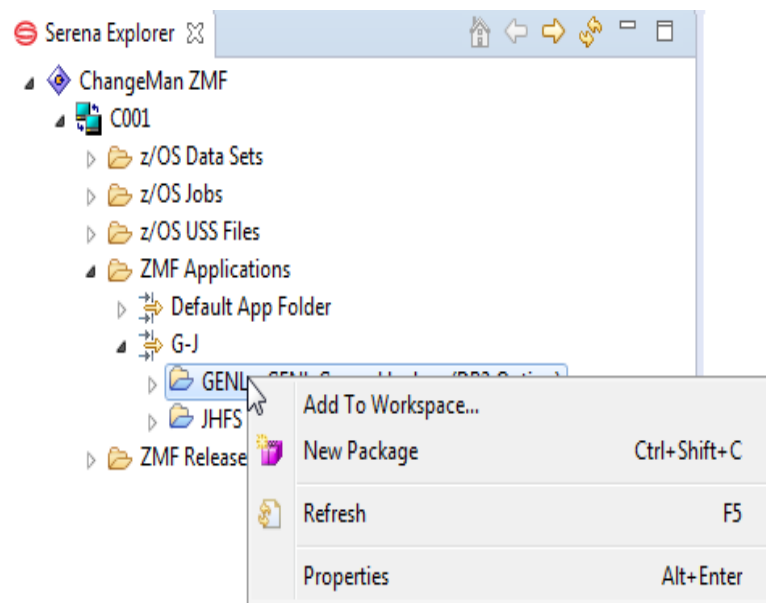
## Global Application Filters

The list of applications shown under the **ZMF Applications** node is filtered. Only one filter may be active at a time, however, so filter nodes are unnecessary and are not shown.

## Operations on ZMF Applications

ZMF Applications  
Contextual Menu

Operations that can be performed on ZMF applications from the workbench are shown in the contextual menu for the affected application. Right-click on the Serena Explorer node for a particular application to bring up its contextual menu.



ZMF Functions for  
ZMF Applications

The following operations are supported on ZMF applications:

- Add to Workspace** — Downloads an entire application to a desktop workspace in one step (in other words, performs a mass download of application components). Libraries or directories on the mainframe are mapped to folders in the workspace as directed by settings in the ZDDOPTS configuration file in ChangeMan ZMF.

See ["Adding an Application to a Desktop Workspace"](#) on page 60 for step-by-step instructions.

- **New Package** — Creates a new change package for the selected application. See ["Creating a Change Package"](#) on page 176 for step-by-step instructions.
- **Refresh** — Refreshes local application metadata and navigation structure based on information in the mainframe ZMF repository.

## Adding an Application to a Desktop Workspace

ZMF for Eclipse supports personal development libraries in the desktop workspace of the Eclipse or RDz workbench. Desktop development libraries let you take advantage of powerful GUI editors and the many open-source code libraries, code generators, developer toolkits, debuggers, documentation generators, and more that are available for modern languages like Java in a desktop environment. At the same time, ChangeMan ZMF retains central control over these software assets.

Mass Download  
of Application  
Libraries

The **Add to Workspace** function on an application's contextual menu invokes the mass download of multiple libraries and components from that application's baseline library to the desktop workspace managed by the workbench. The mapping between ZMF libraries and workspace directories or folders is defined in the ECLIPSE member of the ZDDOPTS parameter library for the ZMF repository where the application's baseline libraries reside.

### **Dynamic Workspace Integration with ZMF**

An application that is downloaded to the desktop using **Add to Workspace** is automatically enabled for ZMF change control under the covers. You can take advantage of this integration using the ZMF for Eclipse functions shown on the **Team** contextual menu in the Java perspective.



**IMPORTANT!** Desktop workspaces must be structured in accordance with the relevant ZMF library mappings in the ZDDOPTS parameter library of ZMF in order for automated application downloads to work as intended.

For information on how to set up library mappings between ChangeMan ZMF and your desktop workspace, refer to the *ZMF for Eclipse Installation and Configuration* document.

### **Mass Download Procedure**

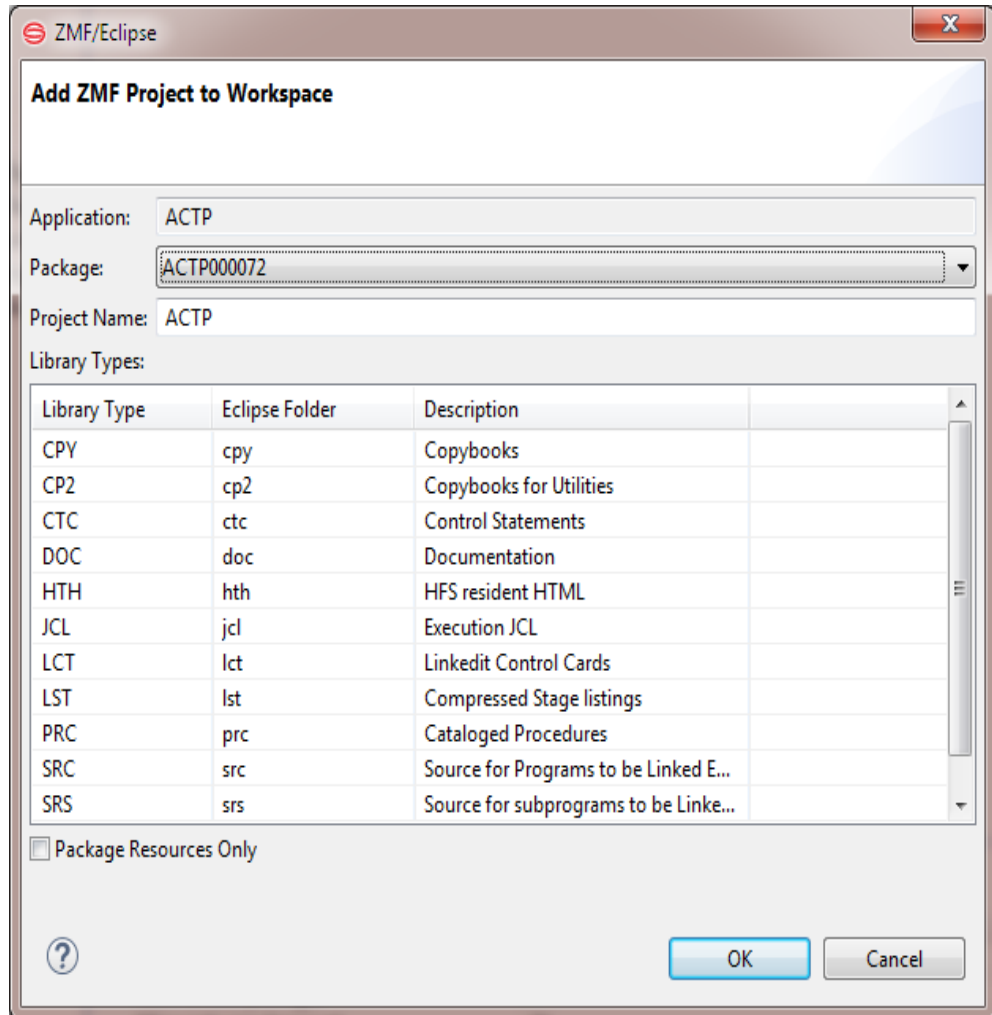
To download the entire contents of a ZMF application to a desktop workspace:

- 1 Open or switch to the Serena perspective.
- 2 Switch to a ZMF-specific desktop workspace. (Select **File | Switch Workspace.**)
- 3 If you do not already have a change package designated to receive component checkins for the project you are working on, create one now. See ["Creating a Change Package"](#) on page 176 for step-by-step instructions.
- 4 Right-click on the desired ZMF application to bring up its contextual menu and select **Add to Workspace.**

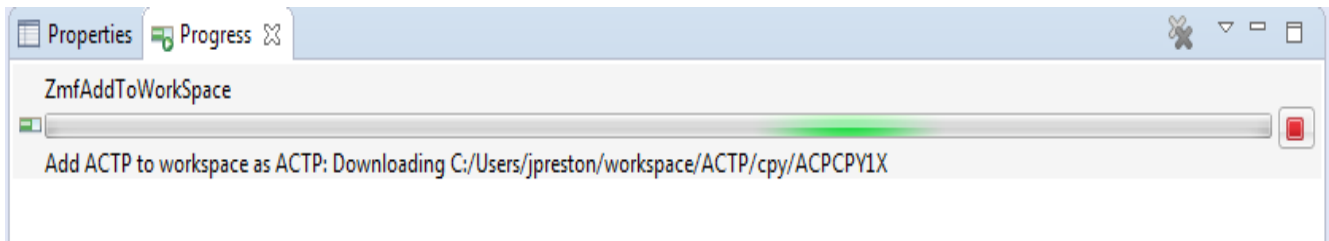


**NOTE** If this option is grayed out, the ZDDOPTS parameter library for ZMF has not been configured to support library mappings for mass downloads. Ask your ZMF administrator to set up the necessary configuration.

- 5 When the **Add ZMF Project to Workspace** window displays, select the name of the package to be used for automated component checkins and staging from development from the pull-down list. Click **OK**.

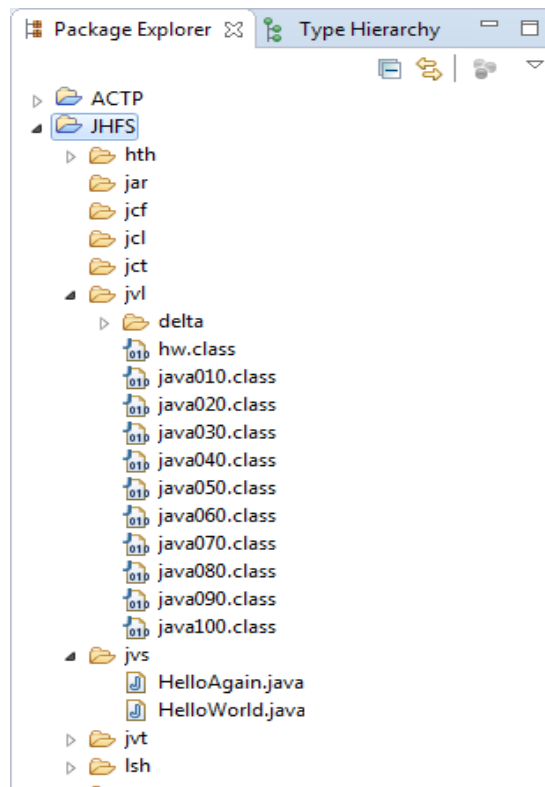


- 6 If you click on the bottom right corner of the Eclipse window, you can enable the Progress tab, and will see the following window.



- If you don't want to add the entire contents of this application to your desktop workspace after all, click the red button on the right to Cancel.
- 7 When the download is complete, the Progress tab will show the text "No operations to display at this time."

- 8 To see the results of the download, switch to the Java perspective and select the **Package** navigation view. You will see a new Java project with the name of the downloaded ZMF application. Expand the project node to see its contents.

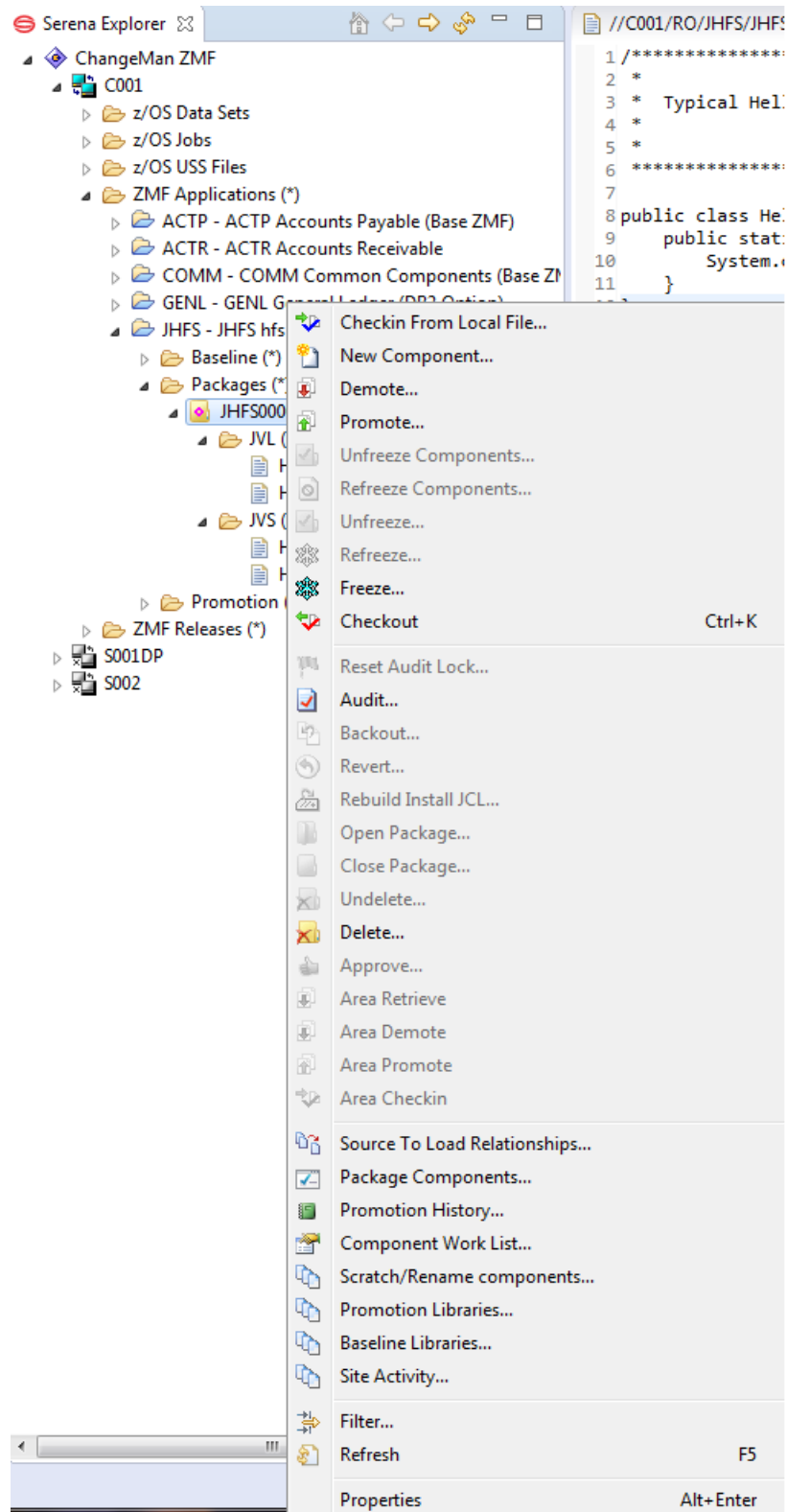


**NOTE** By default, all applications added to a workspace with **Add to Workspace** are configured for dynamic integration with ChangeMan ZMF. This means, for example, that whenever you edit a component in the workbench, ZMF for Eclipse will automatically check it out of baseline, check it into the package you selected at download time, and lock the component against changes by others.

See "[Dynamic Integration with ChangeMan ZMF](#)" on page 100 for more information.

## Working with ZMF Packages

ChangeMan ZMF package functions — such as promote, demote, backout, revert, freeze, audit, and the like — are invoked from the contextual menu for the desired package in the **Serena Explorer** navigation view of the **Serena** perspective.

Package  
Contextual Menu

Many of these same functions can also be invoked from the **Team** contextual submenu of the **Package Explorer** in the **Java** perspective. (See [Chapter 3, "Working with the Java Perspective"](#) on page 95 for more information about the Java perspective.)

## Invoking a ZMF Package Function in Serena Explorer

- 1 In the **Serena Explorer** view of the Serena perspective, expand the node for the ZMF server hosting the repository where the application of interest resides.
- 2 Expand the **ZMF Applications** node, then the node for the desired application, then the **Packages** node within that application. A list of packages displays in the **Serena Explorer** view.
- 3 Right-click on the desired package to bring up its contextual menu, then select the desired ZMF function.
- 4 Provide the requested information when prompted by the ZMF function wizard.
- 5 When finished, click **OK**.

## Package Contextual Menu Functions

The following ZMF package functions can be invoked from the package contextual menu:

- **Checkin from local file** - - Allows you to checkin a component from a local file. A dialog allows you to choose the file, and all the required parameters such as type, package etc, as well the build parameters.
- **New Component** - Allows you to create a new component and optionally edit it. You will be presented with a dialog that asks for the component name and type, and gives you the option to edit the new component.
- **Package Properties** — Edits the properties of an existing ZMF change package using this wizard. Properties that can be changed include package title, description, level (simple or participating), installation instructions, scheduler, user-defined variables, and install site details.

See [Chapter 6, "Editing Package Properties" on page 192](#) for step-by-step instructions.



**NOTE** The **Package Properties** function is specific to ChangeMan ZMF. It should not be confused with the Eclipse **Properties** function, which displays property pages for an object (such as a file or a server connection) managed locally by Eclipse.

- **Demote** — Demotes a change package from a ZMF promotion site and level, such as a specially configured test environment.

See [Chapter 6, "Demoting a Package" on page 209](#) for step-by-step instructions.

- **Promote** — Promotes a change package from development to a ZMF promotion site and level, such as a specially configured test environment.

See [Chapter 6, "Promoting a Package" on page 199](#) for step-by-step instructions.

- **Unfreeze Components** — After a change package has been frozen, this option selectively unfreezes particular components while continuing to enforce frozen status on the package as a whole. This option is typically used to make limited fixes to defects discovered during testing.

See [Chapter 6, "Unfreezing Package Components" on page 197](#) for step-by-step instructions.



- **Refreeze Components** — Refreezes package components that were previously unfrozen using **Unfreeze**.  
See [Chapter 6, "Refreezing Package Components" on page 198](#) for step-by-step instructions.
- **Unfreeze** — After selected package categories have been frozen, this option selectively unfreezes selected package categories.  
See [Chapter 6, "Unfreezing Package Categories" on page 195](#) for more information.
- **Refreeze** — Refreezes selected package categories after they have been unfrozen.  
See [Chapter 6, "Refreezing Package Categories" on page 196](#) for more information.
- **Freeze** — Freezes an entire change package to prevent further changes while the code is being tested. No components may be added to a frozen package.  
See [Chapter 6, "Freezing a Package" on page 194](#) for step-by-step instructions.
- **Checkout (Ctrl-K)** - Starts a dialog to checkout a component.
- **Reset Audit Lock** — Unlocks a package that has been locked for audit. This action permits developers to continue working on components in a change package while a preliminary package audit is under way.  
See [Chapter 6, "Resetting Audit Lock" on page 221](#) for step-by-step instructions.



**IMPORTANT!** Component changes during audit processing can result in object synchronization errors on the audit report. Consequently, this option is NOT recommended for use during the final pre-production package audit.

- **Audit** — Audits a change package to ensure all required components are present and dependencies are synchronized at the correct change level.  
See [Chapter 6, "Auditing a Package" on page 213](#) for step-by-step instructions.
- **Backout** — Backs out a change package that has been deployed into production and restores all production sites to their pre-change condition.  
See [Chapter 6, "Backing a Package Out of Production" on page 225](#) for step-by-step instructions.
- **Revert** — Reverts a frozen or backed out package to development status.  
See [Chapter 6, "Reverting a Package to Development Status" on page 227](#) for step-by-step instructions.
- **Rebuild Install JCL** — Rebuilds the JCL that control installation of a change package into production.  
See [Chapter 6, "Rebuilding Installation JCL" on page 224](#) for step-by-step instructions.
- **Open Package** - Changes the status of a Complex or Super package to OPN. Open status applies to Complex or Super change packages only. These packages are created as containers in order to group together two or more participating packages which share components and may contain inter-dependent changes. These complex/super packages will remain OPEN until all of their participating packages have been baselined.
- **Close Package** - Changes the status of a Complex or Super package to CLO.

- **Undelete** — Restores a memo-deleted change package that has not yet been physically deleted by the ChangeMan ZMF housekeeping job.  
See [Chapter 6, "Undeleting a Change Package" on page 191](#) for step-by-step instructions.
- **Delete** — Memo-deletes a change package. The package and all its components will be physically deleted the next time ChangeMan ZMF housekeeping is run.  
See [Chapter 6, "Deleting a Change Package" on page 190](#) for step-by-step instructions.
- **Approve** — Allows you to approve, reject, or register other approval-related actions against a change package scheduled for installation into production.  
See [Chapter 6, "Approving or Rejecting a Package for Installation" on page 222](#) for step-by-step instructions.
- **Area Retrieve** - removes components from area libraries.
- **Area Demote** - Demote an area.
- **Area Promote** - Promote an area.
- **Area Checkin** - Checkin an area.
- **Source to Load Relationships** — Displays the source-to-load relationships among components in a change package.  
See [Chapter 6, "Viewing Source-to-Load Relationships" on page 237](#) for step-by-step instructions.
- **Package Components** — Displays all components in a change package or any subset that meets your search criteria.  
See [Chapter 6, "Querying Package Components" on page 235](#) for step-by-step instructions.
- **Promotion History** — Displays the promotion history of a change package.  
See [Chapter 6, "Viewing Promotion History" on page 239](#) for step-by-step instructions.
- **Component Work List** — Displays the most recent users to change each component in a change package.  
See [Chapter 6, "Viewing the Component Work List" on page 237](#) for step-by-step instructions.
- **Scratch/Rename Components** — Displays the scratch and rename control record components in a change package.  
See [Chapter 6, "Viewing Scratch/Rename Components" on page 238](#) for step-by-step instructions.  
  
See also ["Removing Rename Records from a Package" on page 193](#) for step-by-step instructions and see [Chapter 6, "Removing Scratch Records from a Package" on page 193](#) for step-by-step instructions.
- **Promotion Libraries** — Lists the promotion libraries, sites, and levels currently associated with a selected change package.  
See [Chapter 6, "Viewing Promotion Libraries" on page 239](#) for step-by-step instructions.

- **Baseline Libraries** — Lists the baseline libraries and levels associated with a selected change package.  
See [Chapter 6, "Viewing Baseline Libraries" on page 240](#) for step-by-step instructions.
- **Site Activity** — Displays starting date and time stamp of activities that were performed at the sites where the selected package is or will be installed.  
See [Chapter 6, "Displaying Site Activities" on page 241](#) for step-by-step instructions.

## Working with ZMF Components

The **Serena Explorer** navigation view in the **Serena** perspective of ZMF for Eclipse provides navigation services in the workbench for mainframe objects. It also provides contextual menus for performing development and change control operations on components in the following ZMF repository environments:

- [Baseline libraries](#)
- [Package staging libraries](#)
- [Promotion libraries](#)

### ZMF Operations on Baseline Library Components

#### Invoking Baseline Component Operations

ChangeMan ZMF functions can be invoked on components in ZMF baseline libraries from the **Serena Explorer** navigation view of the **Serena** perspective. These operations are selected from the contextual menu of the desired component.

- 1 In the **Serena Explorer** view, expand the node for the ZMF server that hosts the repository of interest.
- 2 Expand the **ZMF Applications** node, then the folder for the desired application, then the **Baseline** node within that application, then the library or folder for the desired component type (source, JCL, compiler listing, and so on). A list of library components displays in the **Serena Explorer** navigation view.
- 3 Select one or more components to work with.

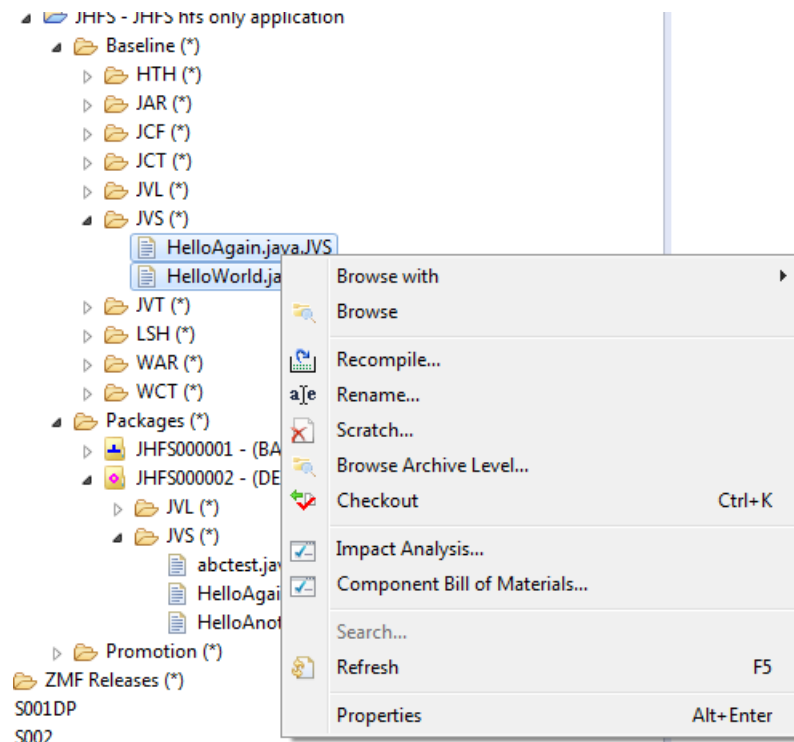


**NOTE** Not all ZMF functions can be performed on multiple components at once.

- 4 Right-click on the selection to bring up its contextual menu, then choose the desired ZMF function from the menu.
- 5 Provide the requested information when prompted by the ZMF function wizard.
- 6 When finished, click **OK**.

Baseline  
Component  
Contextual Menu

The contextual menu for baselined components in a ZMF repository is shown below.

ZMF Functions  
for Baseline  
Components

The following ZMF functions are provided for baseline components.

- **Browse With** — Allows you to select a text editor for viewing a component.
- **Browse** — Opens a selected mainframe component for read-only viewing in a workbench editor. Character code page conversion between mainframe and workbench file formats is performed automatically in the background.  
See ["Browsing a Component" on page 141](#) for step-by-step instructions.
- **Recompile** — Recompiles a source code module without performing a complete build.  
See ["Recompiling a Component" on page 150](#) for step-by-step instructions.
- **Scratch** — Sets up a versioned and reversible delete (or *scratch*) operation on a baselined component. Control records in a change package execute the scratch function when the package is baselined.  
See ["Scratching a Component under Change Control" on page 125](#) for step-by-step instructions.
- **Browse Archive Level** — Displays a dialog box where you may select a baseline level and a text editor for viewing a member. Stacked Reverse Delta (SRD) and PDS baseline library formats are supported.
- **Checkout** — Checks a component out of baseline into a staging library in a change package in the ZMF repository. Alternatively, it checks out the baseline component into a personal development library on the mainframe and associates it with a package to which it will be checked in (that is, staged from development) if any changes are made. Multiple components may be checked out at once.  
See ["Checking Out a Component" on page 129](#) for step-by-step instructions.

- **Rename** — Sets up a versioned and reversible rename of a baselined component. Control records in a change package execute the rename when the package is baselined.  
See ["Renaming a Component under Change Control" on page 127](#) for step-by-step instructions.
- **Component History** — Displays the component history list for the selected component. This history includes all the packages and promotion sites that have contained a version of the component, the timestamps associated with each change state, the build procedure executed on the component at checkin, and the users who have worked on that component.  
See ["Viewing Component History" on page 138](#) for step-by-step instructions.
- **Impact Analysis** — Performs a bottom-up query to find all higher-level components that reference or invoke a selected, subordinate component. For example, a reusable subroutine might be called by multiple higher-level programs, and you might want to discover which programs those are before making any changes to the subroutine.  
See ["Component Impact Analysis" on page 168](#) for step-by-step instructions.
- **Component Build of Materials** — Performs a top-down query to find all the subordinate components on which a selected, higher-level component has dependencies. For example, a higher-level program might call multiple subroutines, and you might want a list of these subroutines before you change the program.  
See ["Component Bill of Materials" on page 165](#) for step-by-step instructions.
- **Search** — Allows you to search baseline members for specified character strings.
- **Refresh** — Refreshes the component information displayed in the workbench from baseline library information in the ChangeMan ZMF repository.



**NOTE** More information about these functions of ChangeMan ZMF is available in the *ChangeMan ZMF User's Guide*.

## ZMF Operations on Package Components

Invoking Package  
Component  
Operations

ChangeMan ZMF functions can be invoked on components in ZMF change packages from the **Serena Explorer** navigation view of the **Serena** perspective. These operations are selected from the contextual menu of the desired component.

- 1 In the **Serena Explorer navigation** view, expand the node for the ZMF server that hosts the repository of interest.
- 2 Expand the **ZMF Applications** node, then the folder for the desired application, then the **Packages** node within that application, then the node for the desired package, then the library or folder for the desired component type (source, JCL, compiler listing, and so on). A list of library components displays in the **Serena Explorer** view.
- 3 Select one or more components to work with.



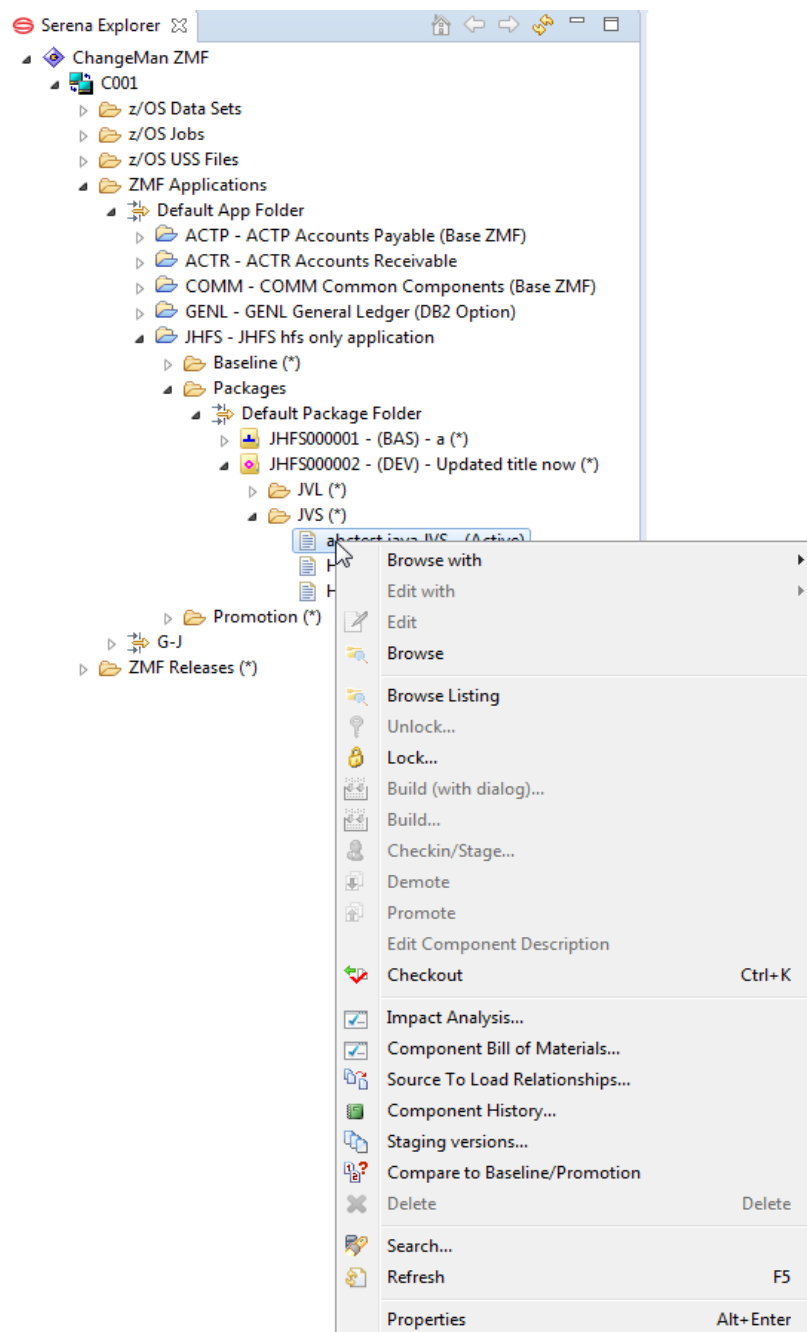
**NOTE** Not all ZMF functions can be performed on multiple components at once.

- 4 Right-click on the selection to bring up its contextual menu, then choose the desired ZMF function from the menu.

- 5 Provide the requested information when prompted by the ZMF function wizard and when ready, click **OK** to submit the function for execution.

Package Component Contextual Menu

The example **Serena Explorer** view below shows the component contextual menu for a source component in a package staging library.



ZMF Functions for Package Components

The following ZMF functions are supported for package components.

- **Browse with** allows you to choose what editor (in read-only mode) to browse the component with.
- **Edit with** allows you to choose what editor to edit the component with.

- **Edit** — In a package context, lets you modify package components residing on the mainframe using a workbench editor. Character code page translation between mainframe and workbench file formats takes place automatically in the background.  
See ["Editing a Component" on page 143](#) for step-by-step instructions.
- **Browse** — In a package context, lets you view package components residing on the mainframe from a read-only editor in the workbench. Components locked by another user may nevertheless be browsed. Character code page translation between mainframe and workbench file formats takes place automatically in the background.  
See ["Browsing a Component" on page 141](#) for step-by-step instructions.
- **Browse Listing** — Provides a navigation shortcut from the currently selected source member to a corresponding compiler listing or other printout saved in electronic form in a LST library in the same change package.  
See ["Browsing a Component Listing" on page 142](#) for step-by-step instructions.
- **Unlock** — Makes a component in a change package available to multiple developers concurrently for changes.  
See ["Locking and Unlocking Components" on page 141](#) for step-by-step instructions.
- **Lock** — Locks a package component to prevent changes by another developer while you are working on it.  
See ["Locking and Unlocking Components" on page 141](#) for step-by-step instructions.
- **Build (with Dialog)...** If you have 'Auto Submit Build' checked in preferences then you can bypass the auto build process by selecting this option, and use the full build dialog. See ["Building a Component" on page 143](#) for step-by-step instructions.
- **Build...** — Translates a package source component into an executable module in the same change package. Default build procedures are predefined in ChangeMan ZMF. Multiple components may be built at once.  
See ["Building a Component" on page 143](#) for step-by-step instructions.
- **Checkin/Stage...** - Starts a dialog to checkin - See ["Invoking the Checkin Function" on page 134](#) for more information.
- **Demote** - Starts a dialog to demote - see ["Procedure for Demoting a Package" on page 210](#) for more information.
- **Promote**- Starts a dialog to promote - see ["Procedure for Promoting a Package" on page 200](#) for more information.
- **Edit Component Description** - allows you to edit the component description via a dialog with fields with package, component and component type, and an input field for the description. If there is already a description present, you can edit it, otherwise input a new one.
- **Checkout** — In a package context, checks out the package component into a personal development library on the mainframe and associates it with the source package. It will be checked in to this same package automatically (that is, staged from development) if any changes are made to the component while it resides in the development library. See ["Checking Out a Component" on page 129](#) for step-by-step instructions.
- **Impact Analysis...** - Starts a dialog to execute a Component Impact Analysis. Performs a bottom-up query to find all higher-level components that reference or invoke a selected, subordinate component. For example, a reusable subroutine might be called by multiple higher-level programs, and you might want to discover which programs those are before making any changes to the subroutine. The inverse of the Component Bill of Materials function.

See [Chapter 5, "Component Impact Analysis" on page 168](#) for step-by-step instructions. See ["Impact Analysis Wizard Step-by-Step" on page 169](#)

- **Component Bill of Materials...** - Starts a dialog to invoke and view the Bill of Materials. Performs a top-down query to find all the subordinate components on which a selected, higher-level component has dependencies. For example, a higher-level program might call multiple subroutines, and you might want a list of all the subroutines involved before you change the program. The inverse of the **Impact Analysis** function.  
See [Chapter 5, "Component Bill of Materials" on page 165](#) for step-by-step instructions. See ["Invoking and Viewing the Bill of Materials" on page 165](#) for more information.
- **Source to Load Relationships...** — Displays all the dependencies between source modules and load modules in the current change package.  
See ["Source-to-Load Relationships" on page 164](#) for step-by-step instructions.
- **Component History...** — Retrieves the change history for a selected component. This history includes all the packages and promotion sites that contain or have contained a version of the component, the timestamps associated with each change state, the build procedure executed on the component at checkin, and the users who have worked on that component.  
See ["Viewing Component History" on page 138](#) for step-by-step instructions.
- **Staging Versions...** — Displays the **Staging Versions** table view, which lists all available backup versions of the selected component in the change package. The displayed versions are then accessible for browsing or comparison.
- **Compare to Baseline/Promotion** — Compares the current staging version of a package component against other versions in baseline or promotion libraries. Differences are displayed side-by-side under the **Compare** tab in the upper right pane of the perspective.  
See ["Comparing Components to Baseline or Promotion" on page 163](#) for step-by-step instructions.



**NOTE** The **Compare to Baseline/Promotion** function does not perform comparisons involving component staging versions. See ["Comparing Component Staging Versions" on page 162](#) if you wish to compare component staging versions.

- **Delete** — Immediately deletes a component from a change package. This is an immediate, local delete, not the versioned **Scratch** operation, which can only be performed on baselined components. Baselined component versions are not affected by the local delete function.  
See ["Deleting a Component" on page 124](#) for step-by-step instructions.
- **Search...** - Starts a search dialog where you can input up to two search strings (boolean AND/OR), start and end columns, maximum search hits, case sensitivity, recursion (for Unix Folders) and if to report member names only.
- **Refresh** — Refreshes the component information displayed in ZMF for Eclipse from metadata maintained in the ChangeMan ZMF repository.  
See ["Viewing the Component Staging Versions List" on page 159](#) for step-by-step instructions.



**NOTE** More information about these functions of ChangeMan ZMF is available in the *ChangeMan ZMF User's Guide*.



## ZMF Operations on Promoted Components

### Invoking Promoted Component Operations

ChangeMan ZMF functions can be invoked on components in ZMF promotion libraries from the **Serena Explorer** view of the **Serena** perspective. To work with a component in a promotion library, perform the following steps.

- 1 In the **Serena Explorer** view, expand the node for the ZMF server that hosts the repository of interest.
- 2 Expand the **ZMF Applications** node, then the node for the desired application, then the **Promotion** node within that application, then the node for the desired promotion site and level, then the node for the library or folder associated with the desired component type (source, JCL, compiler listing, and so on). A list of components displays in the **Serena Explorer**.
- 3 Select one or more components to work with.

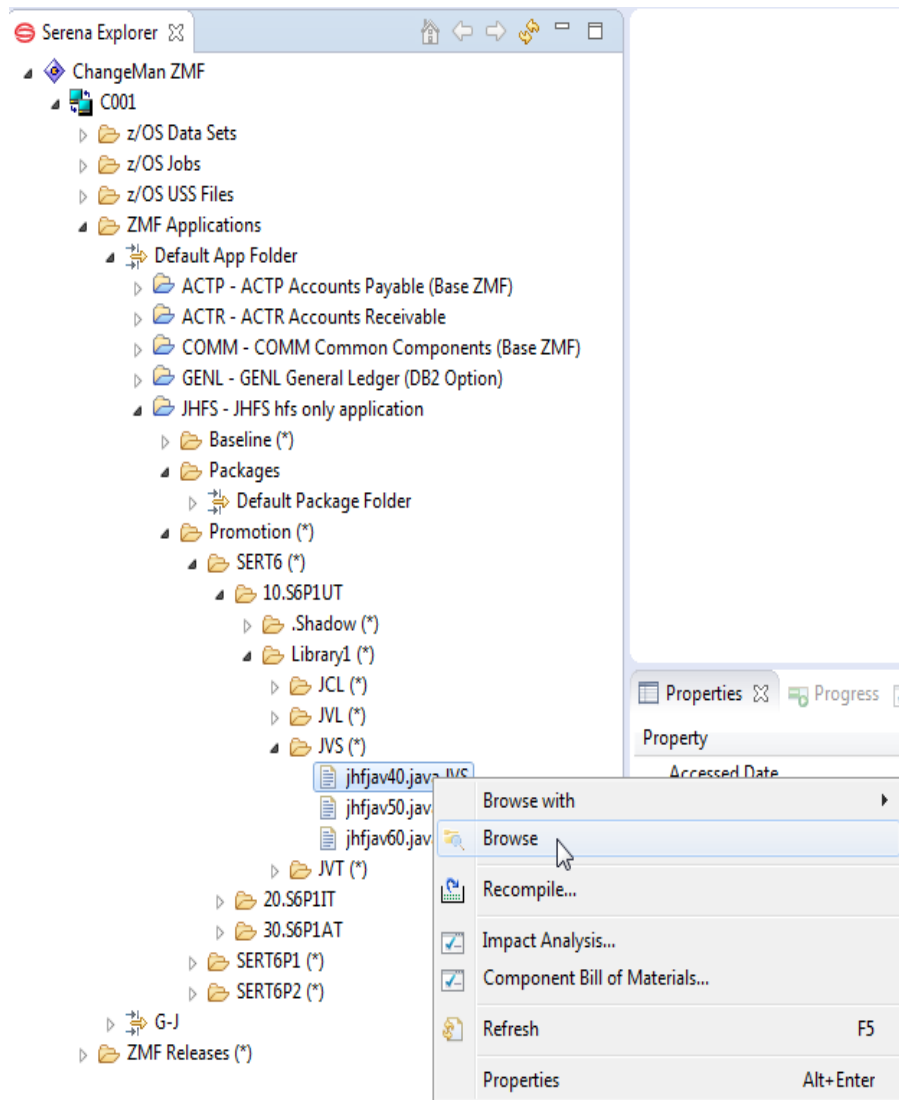


**NOTE** Not all ZMF functions can be performed on multiple components at once.

- 4 Right-click on the selection to bring up its contextual menu, then choose the desired ZMF function from the menu.
- 5 Provide the requested information when prompted by the ZMF function wizard.
- 6 Click **OK** to submit the request for execution.

### Promotion Component Contextual Menu

The example **Serena Explorer** view below shows the component contextual menu for a source component in a promotion library.



### ZMF Functions for Promotion Component

The following ZMF functions are supported for components in promotion libraries.

- **Browse with**
- **Browse** — In a promotion context, lets you view promoted components residing on the mainframe from a read-only editor in the workbench. Components locked by another user may nevertheless be browsed. Character code page translation between mainframe and workbench file formats takes place automatically in the background. See [Chapter 5, "Browsing a Component" on page 141](#) for step-by-step instructions.
- **Recompile** — Recompiles a source code module without performing a complete build. See [Chapter 5, "Recompiling a Component" on page 150](#) for step-by-step instructions.
- **Impact Analysis** — Performs a bottom-up query to find all higher-level components that reference or invoke a selected, subordinate component. For example, if a reusable subroutine is called by multiple higher-level programs, the impact analysis report would list those programs. The inverse of the **Component Build of Materials**. See [Chapter 5, "Component Impact Analysis" on page 168](#) for step-by-step instructions.
- **Component Bill of Materials** — Performs a top-down query to find all the subordinate components on which a selected, higher-level component has

dependencies. For example, if a higher-level program calls multiple subroutines, the build of materials report lists those subroutines. The inverse of **Impact Analysis**. See [Chapter 5, "Component Bill of Materials" on page 165](#) for step-by-step instructions.

- **Refresh** — Refreshes the component information displayed in ZMF for Eclipse from metadata tracked in the ChangeMan ZMF repository.



**NOTE** More information about these functions of ChangeMan ZMF is available in the *ChangeMan ZMF User's Guide*.

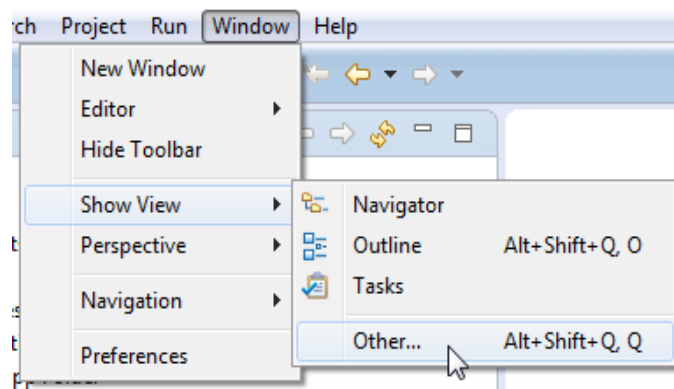
## ZMF Notifications

All notifications issued by ZMF to your mainframe user ID are captured by ZMF for Eclipse in the background and logged in the **ZMF Notifications** table view. These notifications keep you informed of ZMF events such as changes to package status, the outcome of a component build job, the results of a package audit, or requests for package approval.

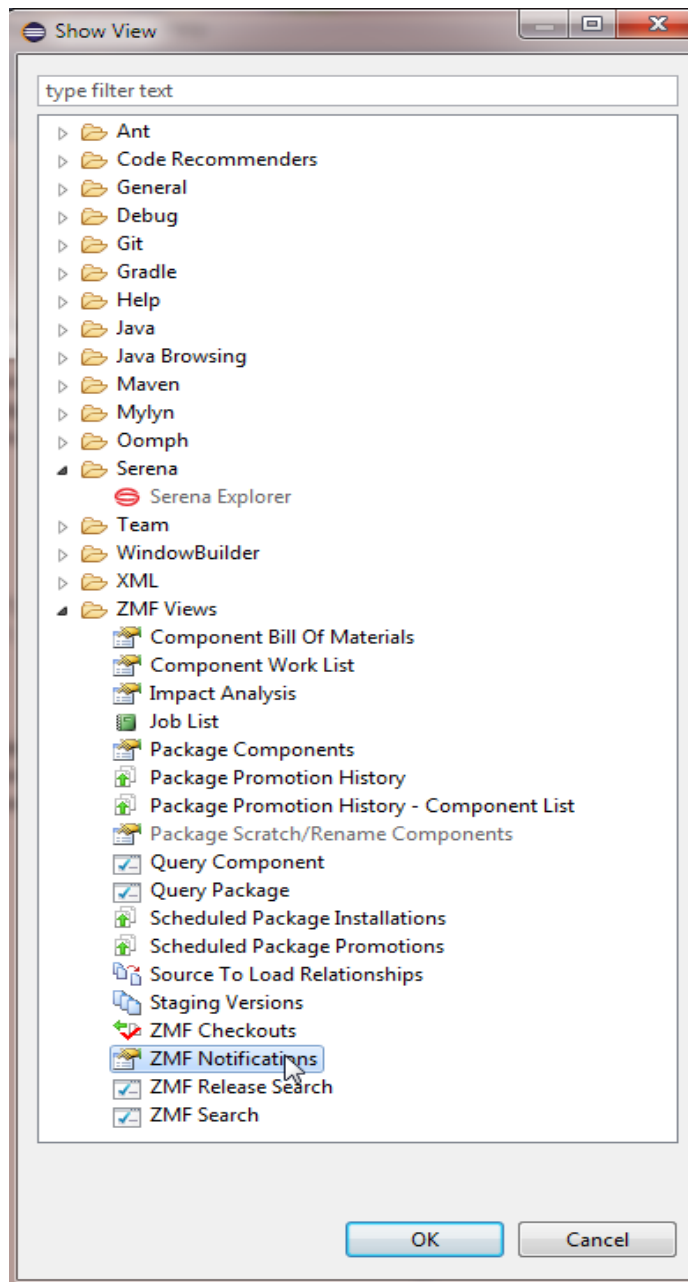
### Displaying the Notifications View

To display the **ZMF Notifications** table view, perform the following steps.

- 1 From the workbench **Window** menu, select **Show View | Other**.



- 2 When the **Show View** dialog box displays, type "ZMF" in the filter text box or scroll to the bottom of the view navigation tree to locate the **ZMF Views** folder.



- 3 Expand the **ZMF Views** folder, select **ZMF Notifications**, and click **OK**.
- 4 The **ZMF Notifications** tab is added to the set of table views in the lower right pane of the **Serena** perspective. Click on the tab to view your notifications.

Date	Source Job	Source ID	Message
2016/07/26 22:39:11	SERT6811	S0260933	Job JPRESTOC(J0261063) submitted
2016/07/26 22:39:41	SERT6811	S0260933	CMN98001 - JHFS000002 component HelloAnother FAILED ON 2016/07/26 @
2016/07/26 22:43:51	SERT6811	S0260933	Job JPRESTOD(J0261068) submitted
2016/07/26 22:44:17	SERT6811	S0260933	CMN408I - JHFS000002 Component NewTest.java ACTIVATED 2016/07/26 @
2016/07/26 22:54:56	SERT6811	S0260933	Job JPRESTOE(J0261074) submitted
2016/07/26 22:55:24	SERT6811	S0260933	CMN98001 - JHFS000002 component abctest.java FAILED ON 2016/07/26 @
2016/07/26 23:01:51	SERT6811	S0260933	Job JPRESTOF(J0261077) submitted
2016/07/26 23:02:20	SERT6811	S0260933	CMN408I - JHFS000002 Component abctest.java ACTIVATED 2016/07/26 @
2016/07/26 23:05:54	SERT6811	S0260933	Job JPRESTOG(J0261080) submitted
2016/07/26 23:06:49	SERT6811	S0260933	CMN408I - JHFS000002 Component abctest.java ACTIVATED 2016/07/26 @
2016/07/26 23:15:05	SERT6811	S0260933	Job JPRESTOH(J0261084) submitted
2016/07/26 23:15:31	SERT6811	S0260933	CMN408I - JHFS000002 Component HelloAnothertry.java ACTIVATED 2016/



**TIP** When creating or changing a connection of your workbench to a Web application server through **Window | Preferences | ZMF/Eclipse | Settings**, you can select the **Show Notify Message Dialog** so that a dialog box appears showing each message as it is received from ZMF. For more information see the *ChangeMan ZMF for Eclipse Installation and Configuration Guide*.

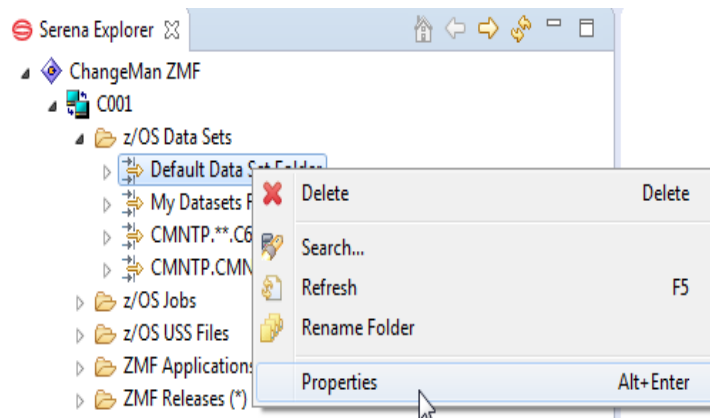
## Filtering z/OS Data Sets

Create, modify, or delete data set viewing filters for the **Serena Explorer** view from the contextual menu for the appropriate filter node.

- [Creating a New Data Set Filter](#)
- [Modifying a Data Set Filter](#)
- [Renaming a Data Set Folder](#)
- [Deleting a Data Set Folder](#)
- [Refreshing a Data Set Filter](#)

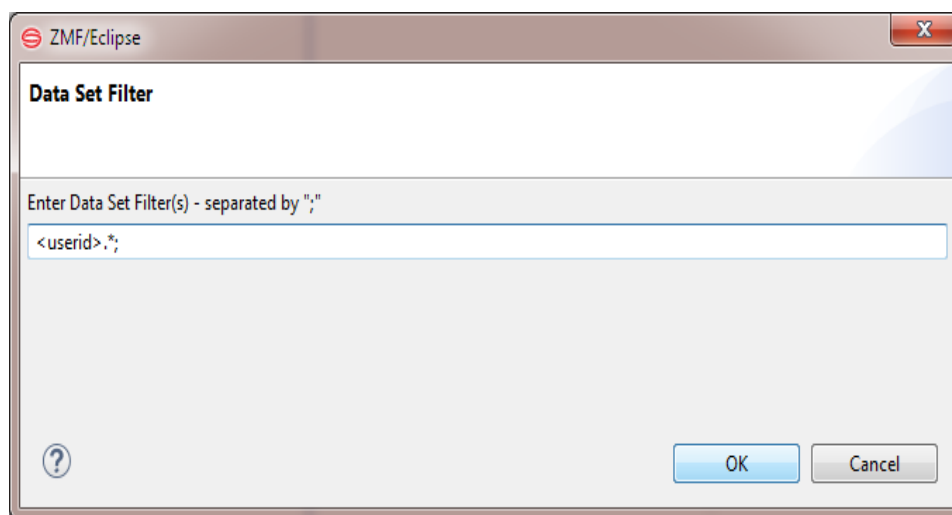
### Creating a New Data Set Filter

- 1 Expand the **z/OS Data Sets** node under the ZMF server whose data sets you want to filter.
- 2 Right-click on the **Default Data Set Filter** folder (or if you have already created others, then you can right click there also). When the contextual menu displays, select **Properties**.



- 3 The **Data Set Filter** window lists the defined data set filter(s) in the selection box. Edit to change or add a new data set filter to the list.

Data Set Filter Window



Data Set Filter String Syntax

The following syntax conventions apply to data set filter strings:

- **Filter strings are applied to data set names** in the z/OS native file system.
- **Data set names** consist of multiple "node" or "qualifier" names separated by periods. Node names may not exceed eight bytes in length, and the data set name as a whole, including periods, may not exceed 44 bytes. The leftmost (or highest level) node is typically the TSO user ID of the data set owner, while the rightmost node is a three-byte identifier for the library type (source code, load members, JCL, and so on). Data set names are not case sensitive.
- **Periods (.) in a filter string** mark node boundaries.
- **An asterisk (\*) in a filter string** is a wild card character that matches any number of characters within a single node of the data set name.
- **A double asterisk (\*\*) in a filter string** matches any number of characters across any number of nodes in a data set name.
- **A question mark (?) in a filter string** is a wild card that matches any one character in the position shown, relative to the beginning of the indicated node.

- **Literals in a filter string** match the identical character in the position shown, relative to the beginning of the indicated node.
  - **The <user id> variable in a filter string** matches the TSO user ID of the current ZMF for Eclipse user in the indicated node position.
  - **Multiple filter strings** are permitted within a data set filter. Filter strings are applied from top to bottom in the order listed in the **Filter Strings** selection box of the **Data Set Filters** window. They are combined with inclusive OR.
- 4 Click **OK** in the **Edit Filter String** dialog to commit your changes to the filter string.
  - 5 If additional filter strings are desired, click the lower **Add** button to the right of the **Filter Strings** selection box in the **Data Set Filters** window. The **Edit Filter String** dialog displays with a default filter string. Edit as needed and click **OK**.
  - 6 If fewer filter strings are desired, select the filter string to be deleted in the **Filter Strings** selection box and click the lower **Delete** button at right. Filter strings must be deleted one at a time.
  - 7 When all desired filter strings are complete and correct, click **Finish** to save your new data set filter and close the **Data Set Filters** window.

## Modifying a Data Set Filter

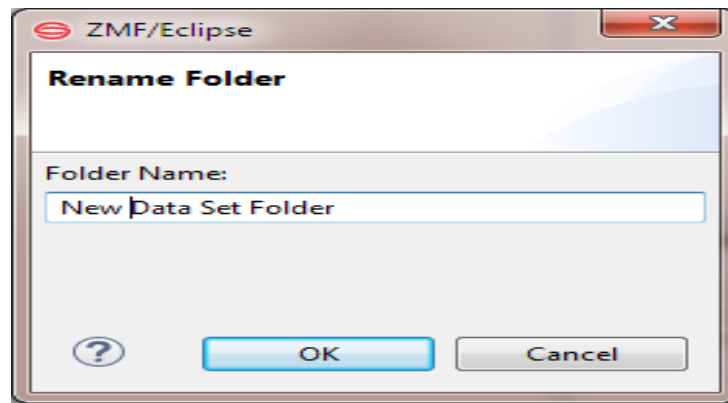
Modifying a data set filter in the **Serena Explorer** view works much like creating one. Refer to ["Creating a New Data Set Filter"](#) above to see screen images. To modify a filter:

- 1 Expand the **Data Sets** node under the ZMF server whose data sets you want to filter.
- 2 Under **Data Sets**, right-click on the folder you want to modify. When the contextual menu displays, select **Properties**.
- 3 When the **Data Set Filter** window displays, edit the selected filter. The procedure is the same as when creating a new data set filter. (For details, refer to ["Creating a New Data Set Filter"](#) above and perform Step 5 through Step 7.)

## Renaming a Data Set Folder

To rename a data set folder in the **Serena Explorer** view, perform the following steps.

- 1 Expand the **Data Sets** node under the ZMF server whose data sets you want to filter.
- 2 Under **Data Sets**, right-click on the folder you want to rename. When the contextual menu displays, select **Rename Folder**.
- 3 When the **Data Set Filters** windows displays, select the filter you want to rename from the **Data Set Filters** selection box in the top half of the window. Then click the topmost **Edit** button to the right of the selection box.
- 4 The **Rename Folder** window displays. Type the new folder name, then click **OK**.

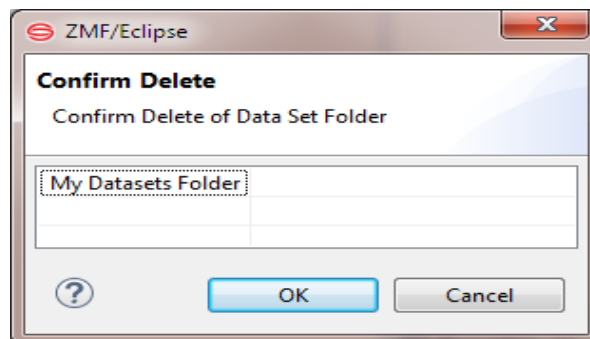


- 5 Click **OK** to save your changes and close the **Rename Folder** window.

## Deleting a Data Set Folder

To delete a data set folder in the **Serena Explorer** view, perform the following steps.

- 1 Expand the **Data Sets** node under the ZMF server whose data sets you want to filter.
- 2 Under **Data Sets**, right-click on the filter node you want to delete. When the contextual menu displays, select **Delete**.
- 3 When the **Confirm Delete** window displays, verify that is the folder you want to remove.



- 4 Click **OK** or **Cancel** to exit and close the **Confirm Delete** window.

## Refreshing a Data Set Filter

The contextual menu for the top-level **z/OS Data Sets** node provides a **Refresh** function that refreshes all filters for z/OS data sets. This function may be invoked after you create a new data set folder in order to see the result.



## Filtering z/OS Unix HFS Files

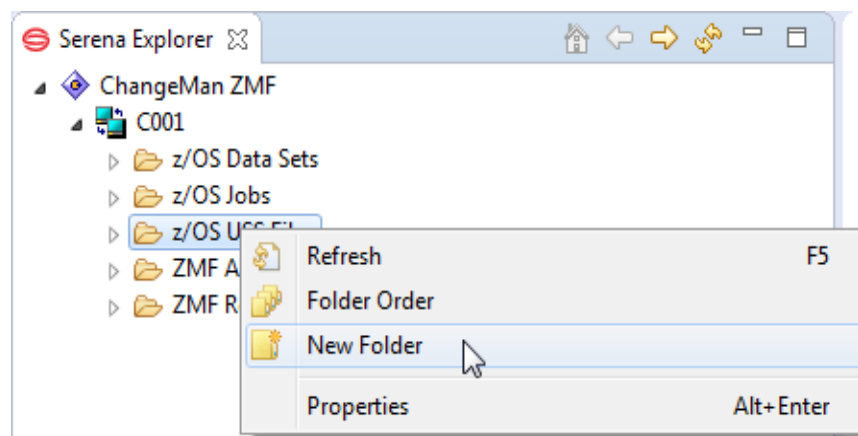
Create, modify, or delete z/OS Unix System Services (USS) Hierarchical File System (HFS) viewing filters for the **Serena Explorer** view from the contextual menu for the appropriate filter node.

- [Creating a New Unix HFS Folder](#)
- [Modifying a Unix HFS Filter](#)
- [Renaming a Unix HFS Filter](#)
- [Deleting a Unix HFS Filter](#)
- [Refreshing Unix HFS Filters](#)

### Creating a New Unix HFS Folder

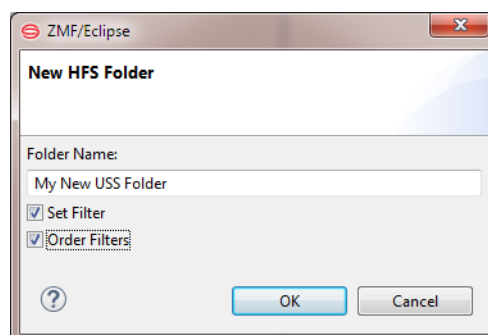
To create a new Unix HFS folder, perform the following steps.

- 1 Right click the **z/OS USS Files** node under the ZMF server whose Unix HFS folders and files where you want to add a new folder.
- 2 Click on the **New Folder** option on the contextual menu.

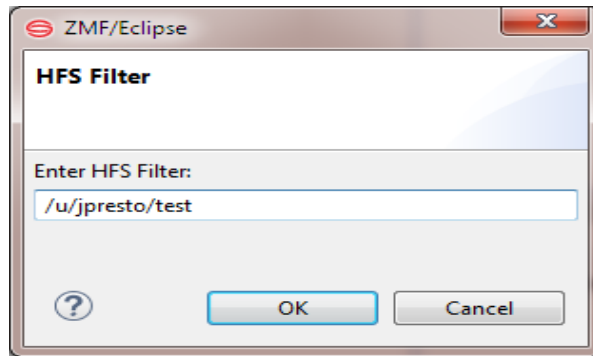


- 3 The **New HFS Folder** window asks you for the new folder name, and also optionally for the set filter and order filters dialogs to be run. Click **OK**.

New HFS Folder Window

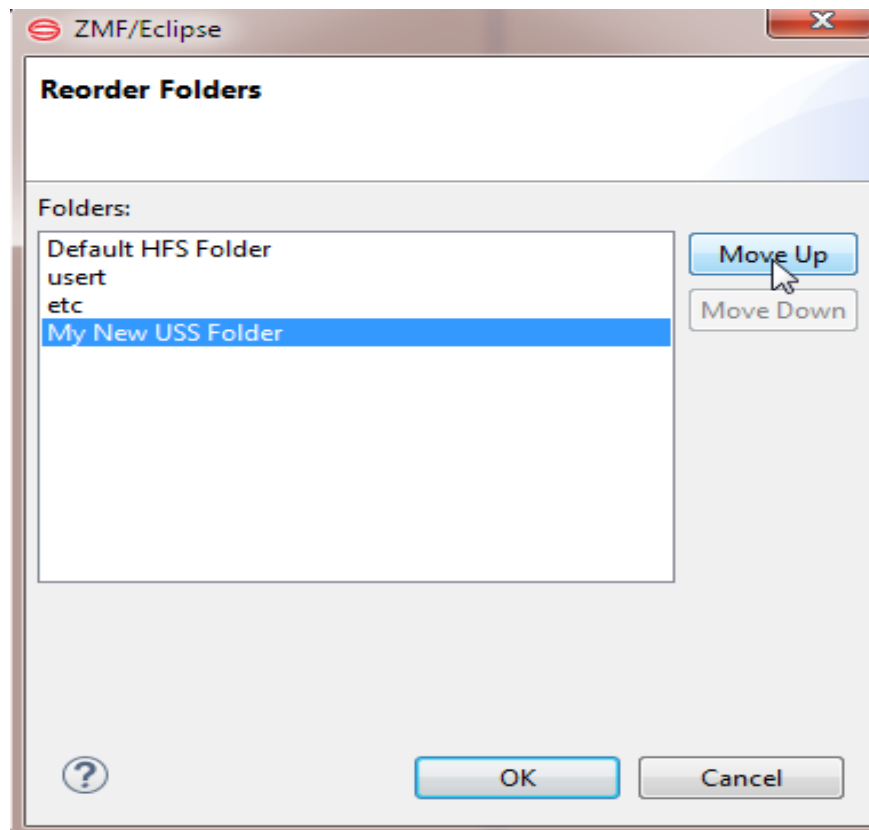


- 4 If you selected **Set Filter**, a dialog box prompts you for the name of the new HFS filter.



Type the new filter value and click **OK**.

- 5 If you selected **Order Filters**, a **Reorder Folders** window will allow you to rearrange the folders, simply highlight the one you want to move, and click on **Move Up** or **Move Down** as desired.



- 6 When finished click **OK** to save and exit.

#### HFS Filter String Syntax

The following syntax conventions apply to data set filter strings:

- **Filter strings are applied to HFS paths** in the z/OS Unix file system.
- **Path names are root relative** and consist of any number of directory or folder names separated by a *forward slash*. The topmost directory in the string must be preceded by a forward slash.

- **The asterisk (\*) and question mark (?) in an HFS filter string** are treated as literal values, not wildcard characters.
- **A period (.) in an HFS filter string** selects the contents of the current directory in the path. It may not be mixed with literals.
- **Literals in a filter string** match the identical character in the position shown, relative to the beginning of the indicated node.
- **The <userid> variable in a filter string** matches the TSO user ID of the current ZMF for Eclipse user in the indicated position.
- **Multiple filter strings** are not permitted in a HFS filter.

## Modifying a Unix HFS Filter

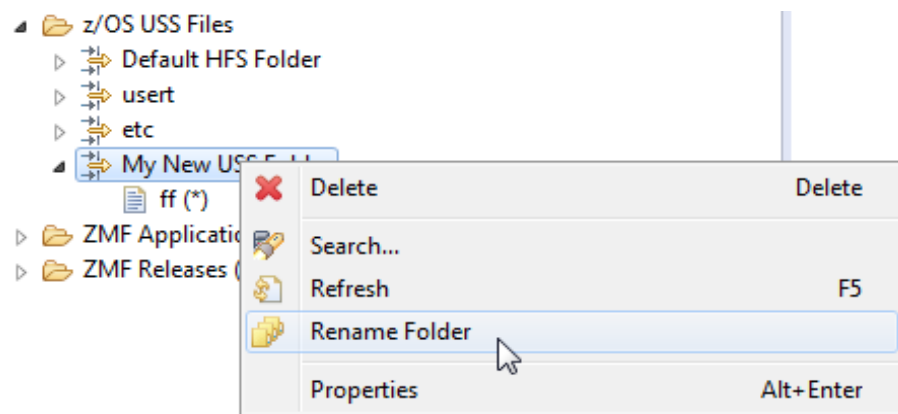
To modify an HFS filter in the **Serena Explorer** view, perform the following steps.

- 1 Expand the **z/OS USS Files** node under the ZMF server whose HFS folders and files you want to filter.
- 2 Under **z/OS USS Files**, right-click on the folder you want to modify. When the contextual menu displays, select **Properties**.
- 3 The procedure is the same as when creating a new HFS filter.

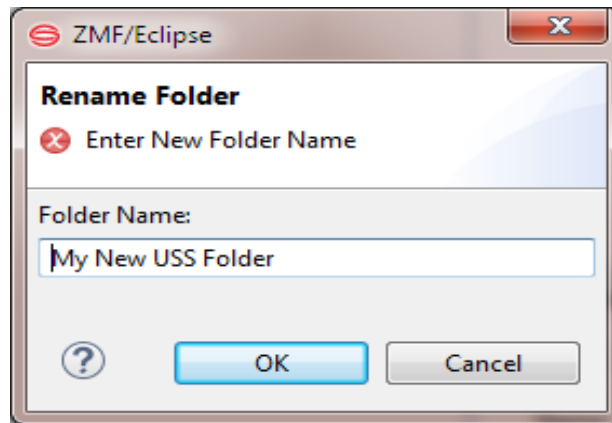
## Renaming a Unix HFS Filter

To rename an HFS folder in the **Serena Explorer** view, perform the following steps.

- 1 Expand the **z/OS USS Files** node under the ZMF server whose files you want to filter.
- 2 Right-click on the folder you want to rename.
- 3 Select the **Rename Folder** menu option.



- 4 The **Rename Folder** window displays. Type the new folder name, then click **OK**.



## Deleting a Unix HFS Filter

To delete an HFS folder in the **Serena Explorer** view, perform the following steps.

- 1 Expand the **z/OS USS Files** node under the ZMF server desired.
- 2 Right-click on the folder you want to delete. When its contextual menu displays, select **Delete**.
- 3 When the **Confirm Delete** window displays, select OK to confirm the deletion.

## Refreshing Unix HFS Filters

The contextual menu for the top-level **z/OS USS Files** node provides a **Refresh** function that refreshes the list of folders for z/OS Unix HFS files. This function may be needed after you create a new HFS filter.

## Filtering z/OS Jobs

By default, the viewing filter for z/OS job output shows jobs submitted under the authority of your TSO user ID, or jobs that include your user ID as part of the job name on a JOB card. Jobs which execute under the authority of security entities other than your TSO user ID (such as ZMF build jobs) or jobs with their own application names in the job card JCL require a custom filter to display under the **z/OS Jobs** node of **Serena Explorer**.

Create, modify, or delete job viewing filters for the **Serena Explorer** view from the contextual menu for the appropriate filter node.

- [Creating a New Job Filter](#)
- [Modifying an Existing Job Filter](#)
- [Renaming a Job Filter](#)
- [Deleting a Job Filter](#)

- Refreshing Job Filters

## Creating a New Job Filter

- 1 Right Click the **z/OS Jobs** node under the ZMF server.
- 2 Select **New Folder**.
- 3 The **New Job Folder** window displays.

New Job Folder Window

- 4 Type the name of the new filter in the text box and optionally select Set Filter and Order Filters, and click **OK**.
- 5 If you checked the **Set Filter** option, the **Job Filter Details** window is displayed.

Job Name	Job Owner
*	JPRESTO

- 6 When the **Job Filter Details** dialog displays, edit the default search patterns as needed. Values entered in the text boxes of the dialog map to operands in parentheses in the filter strings shown in the **Job Filters** window.
  - **Job Name** — Enter a search pattern for the job name or application name.
  - **Job Owner** — Enter a search pattern for the security entity that owns the job.

Job Filter String  
Syntax

The following syntax conventions apply to job filter strings:

- **The asterisk (\*) in a job filter string** is a wildcard character that matches any number of characters of any value in the position shown.
  - **Literals in a filter string** match identical characters in the positions shown.
  - **The <userid> variable in a filter string** matches the TSO user ID of the current ZMF for Eclipse user in the indicated position.
- 7 Click **OK** in the **Job Filter Details** dialog to commit your changes to the filter string.
  - 8 If additional filter strings are desired, click the upper **New Job Filter** button to the right of the window. The **Job Filter Details** dialog displays with a default filter string. Edit as needed and click **OK**.
  - 9 If fewer filter strings are desired, select the filter string to be deleted in the **Filter Strings** selection box and click the lower **Delete Filter** button at right. Filter strings must be deleted one at a time.
  - 10 When all desired filter strings are complete and correct, click **OK** to save your new data set filter and close the **Job Filter Details** window.
  - 11 If you selected the option to **Reorder Filters** then the Reorder Folders window will display, and you may then move the folders up or down.

## Modifying an Existing Job Filter

To modify a job filter in the **Serena Explorer** view, perform the following steps.

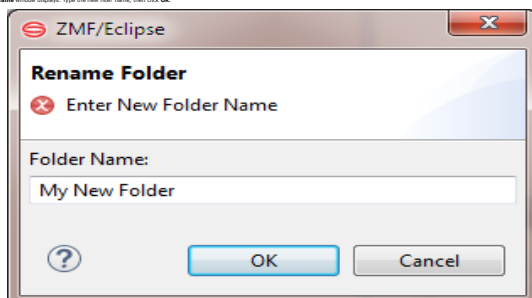
- 1 Expand the **z/OS Jobs** node under the ZMF server.
- 2 Under **z/OS Jobs**, right-click on the folder node you want to modify. When the contextual menu displays, select **Properties**.
- 3 When the **Job Filter Details** window displays, edit as described earlier.

## Renaming a Job Filter

To rename a job filter in the **Serena Explorer** view, perform the following steps.

- 1 Expand the **z/OS Jobs** node under the ZMF server.
- 2 Right-click on the folder you want to rename. When its contextual menu displays, select **Folder Rename**.
- 3 The **Rename Folder** window displays. Type the new filter name, then click **OK**.

The Edit Filter Name window displays. Type the new filter name, then click OK.



- 4 Click **OK** to save your changes and close the **Rename Folder** window.

## Deleting a Job Filter

To delete a job filter in the **Serena Explorer** view, perform the following steps.

- 1 Expand the **z/OS Jobs** node under the ZMF server.
- 2 Right-click on the folder you want to delete. When its contextual menu displays, select **Delete**.
- 3 When the **Confirm Delete** window displays, first verify that you have the right folder, then click the **OK** button to delete .

## Refreshing Job Filters

The contextual menu for the top-level **z/OS Jobs** node provides a **Refresh** function that refreshes the list of folders for z/OS JES jobs. This function may be needed after you create a new job folder.

# Filtering ZMF Applications

The view of ZMF applications shown in **Serena Explorer** is filtered. Multiple application folders may be defined. The default, uses a filter to show all ZMF applications in the selected repository that your TSO user ID is authorized to access.



**NOTE** You cannot modify an application filter to show applications that your TSO user ID is not authorized to access. Contact your ZMF administrator if you need access to applications that you currently cannot view in ZMF for Eclipse.

Work with application and package filters from the **ZMF Applications** contextual menu.

- [Modifying an Application Folder](#)
- [Filtering Package Views](#)

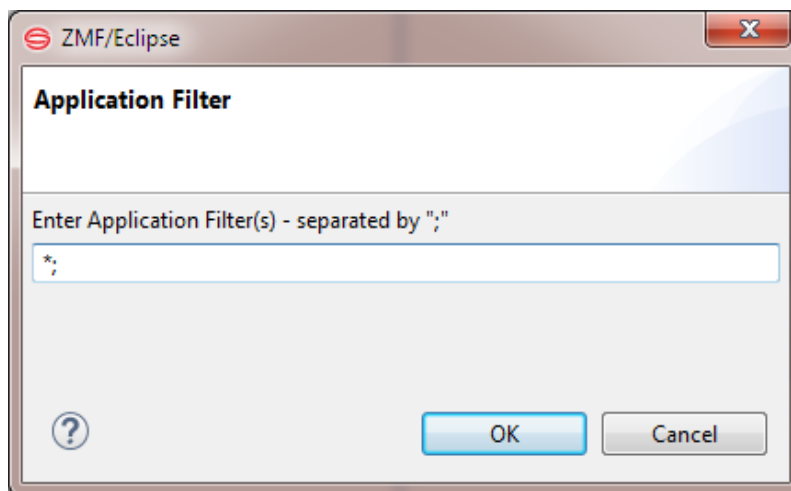
## Modifying an Application Folder

To modify an application viewing filter, perform the following steps.

- 1 Right-click on the **ZMF Applications** node to bring up its contextual menu, then select the **Properties** option.

- The **Application Filter** window displays.

Application/  
Package Filters  
Window



By default, the application filter contains a single filter string with a wildcard character that matches all ZMF application IDs. Multiple filter strings are permitted. Filter strings are applied from top to bottom in the order listed in the **Application Filter** text box. They are combined with inclusive OR.

Filter string(s) may be added, deleted, or modified in the text box.

Application Filter  
String Syntax

The following syntactic conventions apply to application filter strings:

- **An asterisk (\*) in a filter string** is a wild card character that matches any number of characters in the application name.
  - **A question mark (?) in a filter string** is a wild card that matches any one character in the position shown, relative to the beginning of the application name.
  - **Literals in a filter string** match the identical character in the position shown, relative to the beginning of the application name.
- a Click **OK** in the **ZMF Application Filter** dialog to commit your changes to the filter string. You will return to the **Application/Package Filters** window.
- If additional filter strings are desired, simply add them here separated by semi-colons ";" as desired.
  - Click **OK** to save your new Application Filter(s).

## Filtering Package Views

By default, all packages for an application are included in the **Serena Explorer** navigation view. However, you can filter these packages for viewing based on their properties in ChangeMan ZMF. Filterable properties include current package status, package level, package type, creator IDs, work request ID, department, install date range, create date range, package number, site name, and approval entity.

To modify the package filters for an application filter, perform the following steps.

- In the **Serena Explorer** view of the Serena perspective, expand the node for the ZMF system where the packages of interest reside.



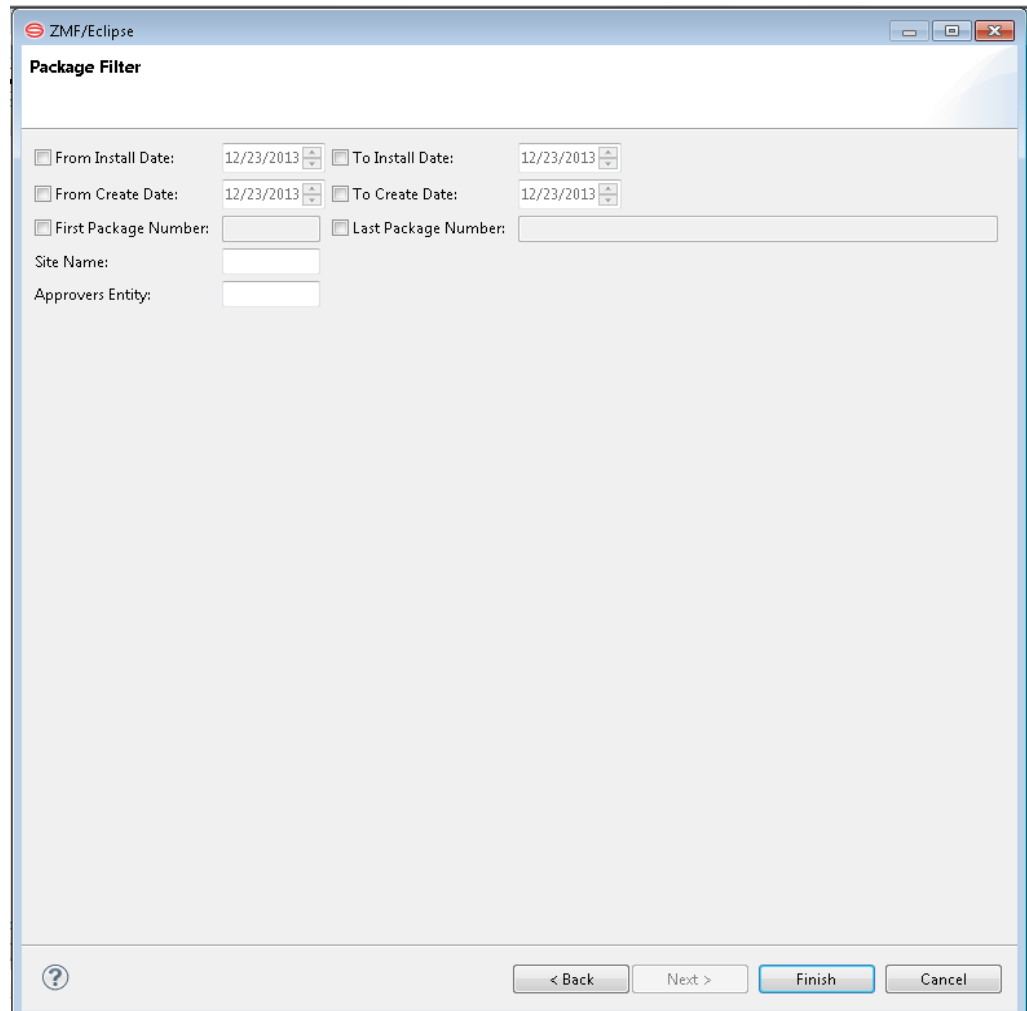
- 2 Right-click on the **ZMF Applications** node to bring up its contextual menu, then select **Properties**.
- 3 When the **Application/Package Filters** window displays, select an application filter in the **Application Filters** text box and click the **Package Filter** button.
- 4 The first page of the **Package Filter** wizard displays.

The following package property options may be modified for the selected application filter:

Field	Description
Package Status	Select or deselect the package statuses to be displayed.
Package Level	Select or deselect the package levels to be displayed.
Package Type	Select or deselect the package types to be displayed.
Creator's id List	Enter one or more TSO user IDs of package creators, delimited by semicolons.

Field	Description
Work Request	Enter a work request ID.
Department	Enter a department name.

- 5 When you are finished modifying the above options, click **Next** to display the second page of the **Package Filter** wizard.



- 7 The following options on this page may be modified:

Field	Description
From Install Date To Install Date	Lets you filter packages based on their scheduled installation date. You can filter by absolute date or by a floating window of dates.
From Create Date To Create Date	Lets you filter packages based on their creation date. You can filter by absolute date or by a floating window of dates.

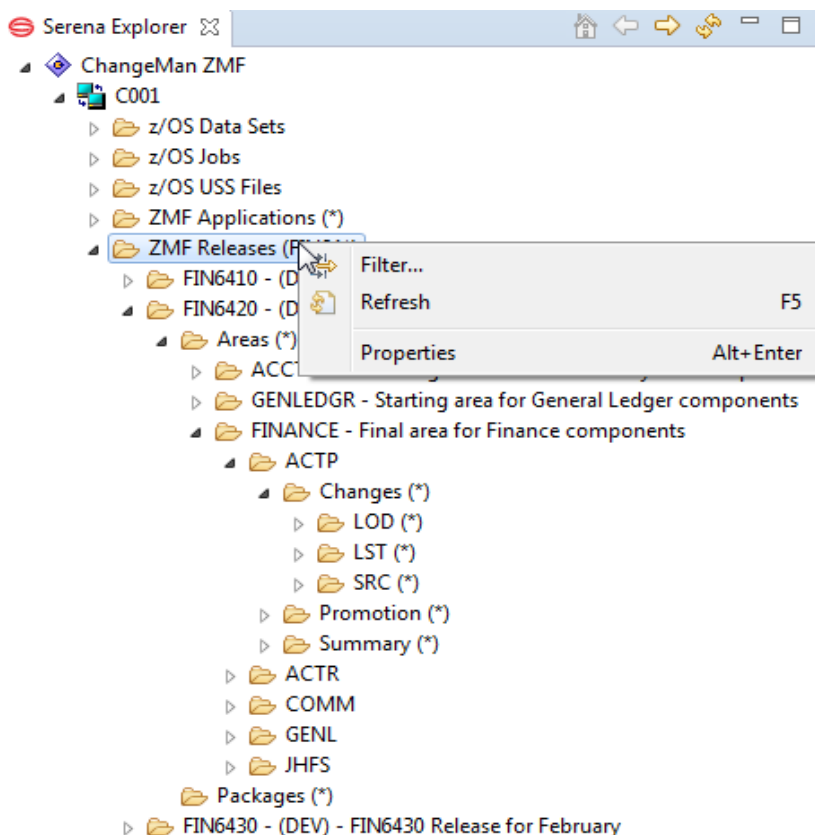
Field	Description
First Package Number Last Package Number	Lets you specify a range of package numbers to include in the filtered view. The range may be open-ended, including all package numbers up to and including a certain value, or all packages of a certain number or higher.
Site Name	Enter a site name.
Approvers Entity	Enter an approval entity.

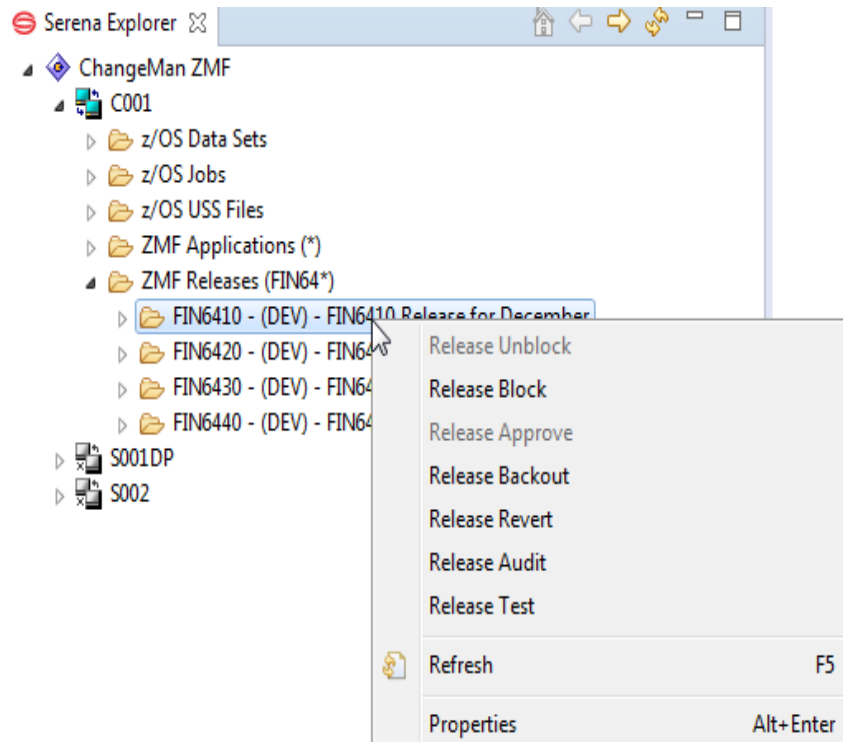
- 8 When all desired package filtering options are complete and correct, click **Finish** to save your new package filter.

## Working with ZMF Releases

If you are licensed for ERO, the **Serena Explorer** navigation view in the **Serena** perspective of ZMF for Eclipse provides the ability to work with ZMF Releases directly. More information is available in the ERO User's Guide.

### The contextual menu for Releases.



**The contextual menu for a Release.**

**Release Unblock** - Unlock the release for further changes.

**Release Block** - Lock down the release and its areas in preparation for an install.

**Release Approve** - All install approvers must enter their approvals before the release will install. When the last approval is entered, the release status is changed to APR.

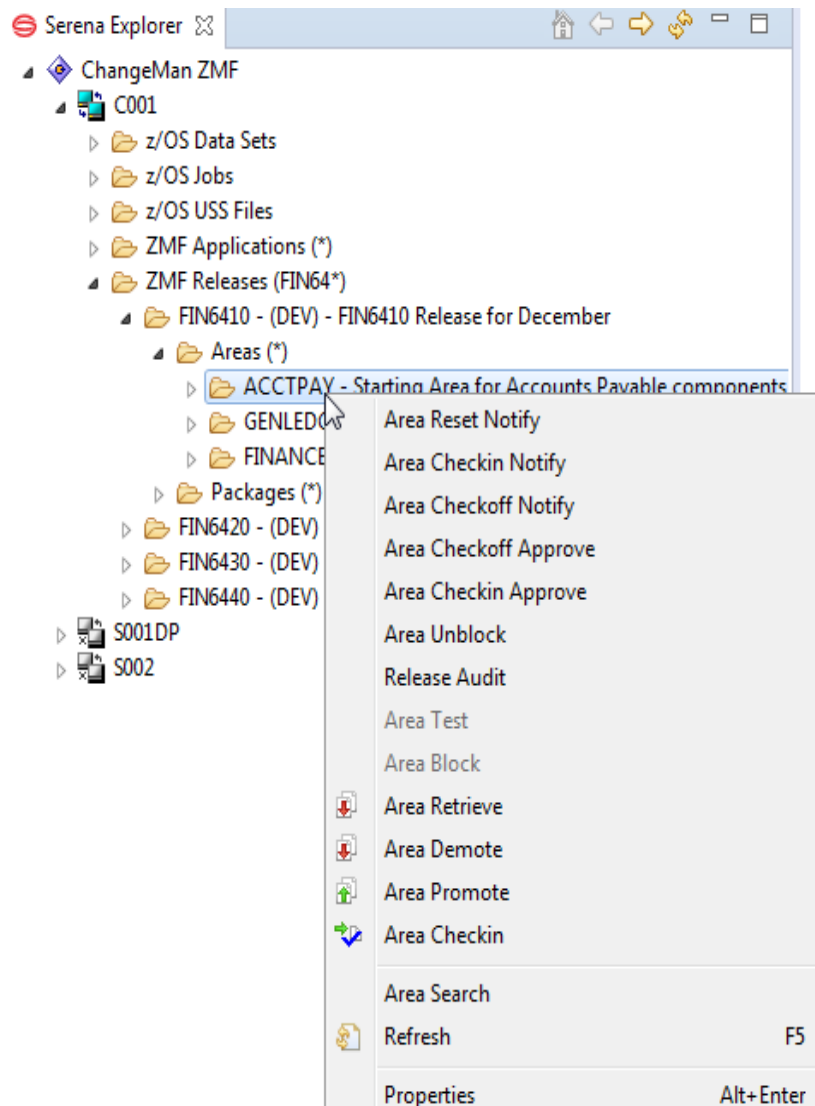
**Release Backout** - Release backout first verifies that all packages attached to the release are in a state that permits package backout. Then release backout submits package backout jobs from the X node libraries for the packages attached to the release. After all packages have been backed out, the packages and the release are in BAK status.

**Release Revert** - Release revert clears all release install approvals, unblocks the release, and changes the status of the release from APR or BAK to DEV status.

**Release Audit** - Will start a dialog to submit a batch job to audit the Release Area, asking you for the Release Area, Auto Resolve Scope, Ignore higher areas (Yes/No or Conditional) and include related applications.

**Release Test** - Will start a dialog to perform a Release Test, optionally checking cleanup of empty packages, Components from different packages, and not checked in components.

### The contextual menu for Release Areas



**Area Reset Notify** - Resets the check-in approvers for the selected area in the release.

**Area Checkin Notify** - Notifies the check-in approvers for the selected area in the release. Each approver will receive a message that their approval is awaited. If already done then you will receive a message that check-in approvers already notified.

**Area Checkoff Notify** - If the area is blocked, then it will notifies the check-off approvers that their approval is awaited, otherwise advises the area is not blocked.

**Area Checkoff Approve** - Grants permission to check-in the contents of area libraries to the next area.

**Area Checkin Approve** - Opens a release area for area check-in.

**Area Unblock** - Will unblock an Area that is blocked. Greyed out if not blocked.

**Release Audit** - A dialog will ask for Release Area, Auto Resolve Scope, Ignore Higher Areas, and Include Related Applications. Then it will submit a batch job to execute a Release Audit. You will need to look at the job output in the z/OS Jobs section above.

**Area Test** - Compares the contents of an area to the contents of packages attached to the release to find mismatches.

**Area Block** - Will block an Area that is not blocked. Greyed out if already blocked.

**Area Retrieve** - Starts a dialog that asks you the Application, then gives you component selection criteria, then allows you to select the components desired.

**Area Demote** - Starts a dialog that asks you the Application, then select a site, and submits a job.

**Area Promote** - Starts a dialog that asks you the Application and a time to schedule if desired, then select a site, and submits a job.

**Area Checkin** - Starts a dialog that asks you the Application, Eligible Components Only, Exclude Superseded Components, and Specify Component Checkin Criteria, then select the components, and submits a job.

**Source To Load Relationships...** through to **Site Activity...** - see the later parts of the section "[Package Contextual Menu Functions](#)" on page 64

## Chapter 3

---

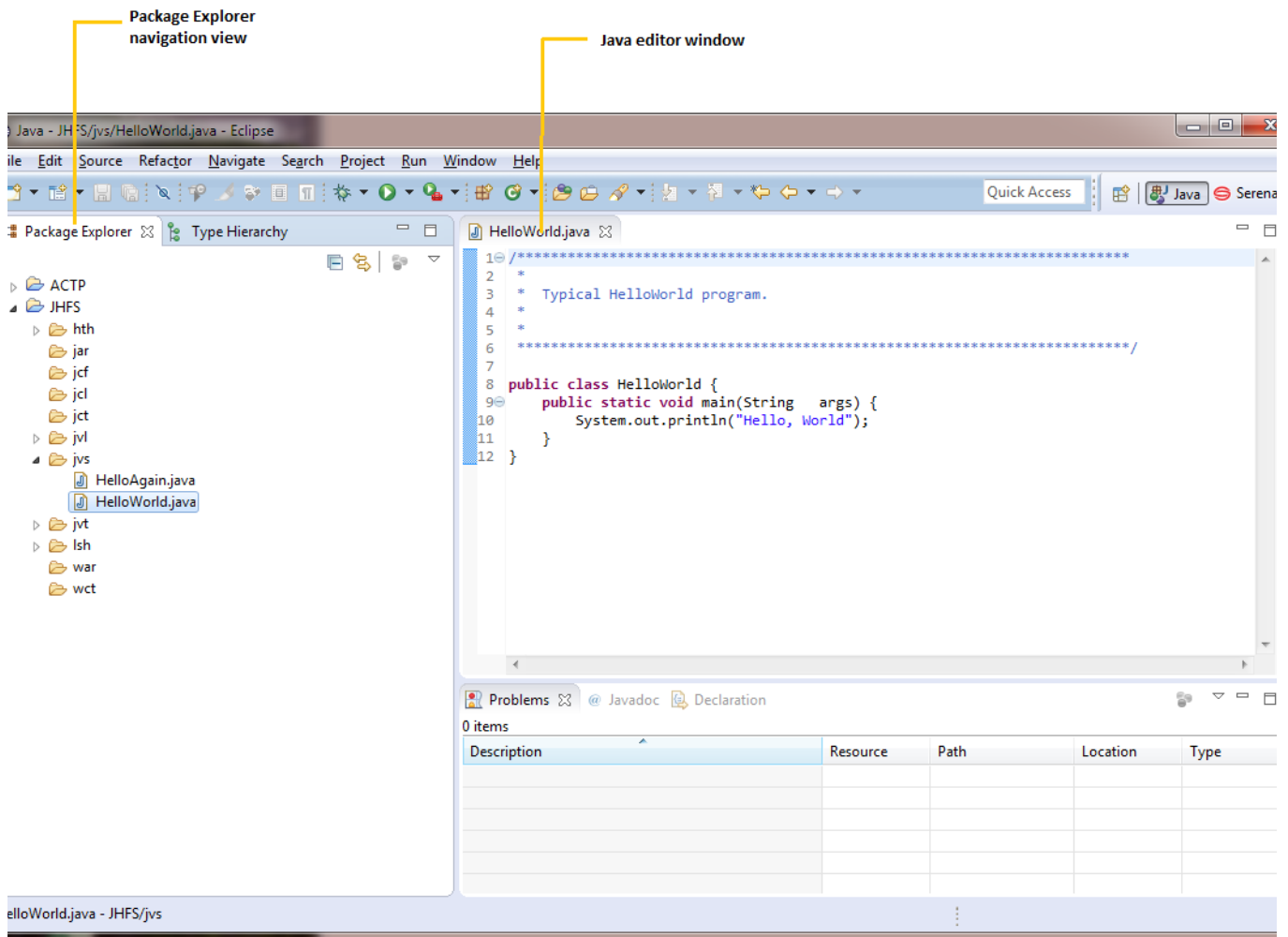
# Working with the Java Perspective

Java Perspective Overview	96
Dynamic Integration with ChangeMan ZMF	100
ZMF Functions in the Package Explorer	101

---

## Java Perspective Overview

The Java Perspective The default Java perspective in Eclipse and RDz displays the **Package Explorer** view in the left pane, one or more tabbed editor windows in the upper right pane, and various utility views in the pane at the lower right. The active view is highlighted.



### The Package Explorer Navigation View

The main view in the **Java** perspective is the **Package Explorer** navigation view. The **Package Explorer** view provides hierarchical navigation among Java projects and development resources in the desktop workspace.

- **Expand or collapse a node** in the hierarchy by clicking on its plus or minus symbols.
- **Open the contextual menu for a resource** by right-clicking on its name.
- **Perform actions on a resource** by choosing commands from its contextual menu.



## Working with the Java Perspective

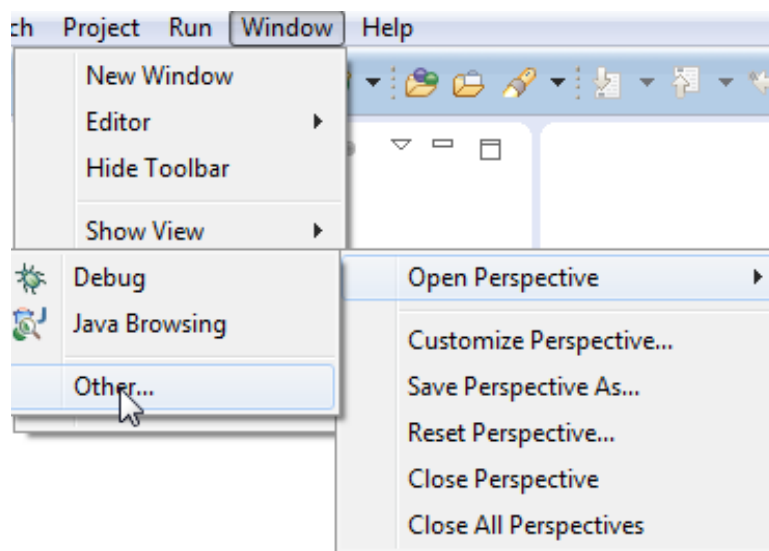
The **Java** perspective is native feature of the Eclipse workbench and is not described in detail here. However, you should become familiar with the following essentials.

- [Opening the Java Perspective](#)
- [Switching Between Perspectives](#)
- [Enabling ZMF Functions in the Team Menu](#)

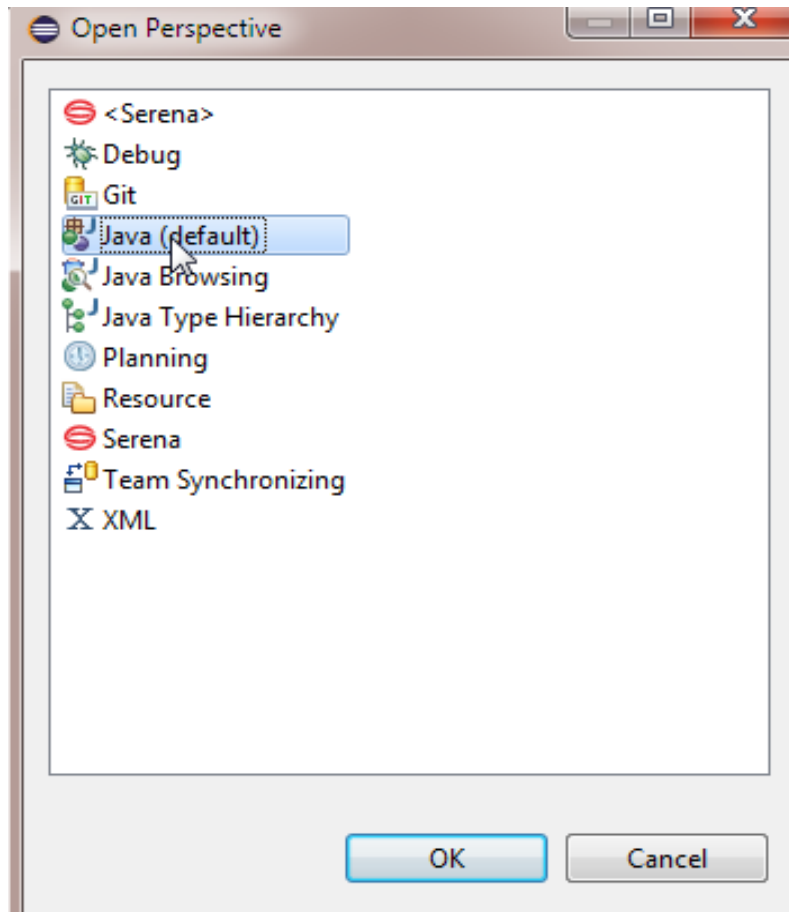
## Opening the Java Perspective

To open the Java perspective in the workbench, perform the following steps.

- 1 From the workbench **Window** menu, select **Open Perspective | Other**.



- 2 When the **Open Perspective** dialog appears listing the "other" perspective choices, select the **Java** perspective.



- 3 Click **OK**. The Java perspective displays, showing the **Package** navigation view.

## Switching Between Perspectives

Switch between multiple open perspectives in the workbench by clicking on a perspective button in the upper right of the workbench window.

- **Switch to the Java perspective** from other perspectives by clicking on the **Java** button in the upper right of the workbench window.



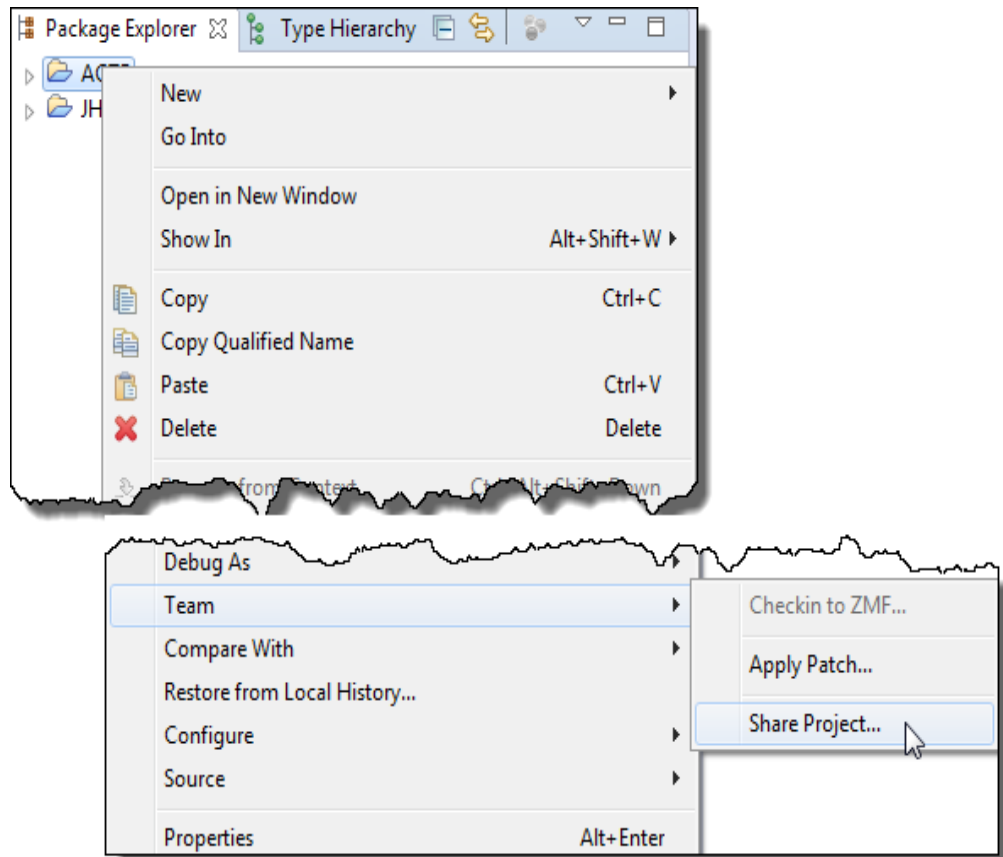
- **Switch to the Serena perspective** from the **Java** perspective or elsewhere by clicking on the **Serena** button.
- **In RDz only:**
  - **Switch to the RDz z/OS Projects perspective** from the Java perspective or elsewhere by clicking on the **z/OS Projects** button.
  - **Switch to the RDz Remote System Explorer perspective** from the Java perspective or elsewhere by clicking on the **Remote System Explorer** button.

## Enabling ZMF Functions in the Team Menu

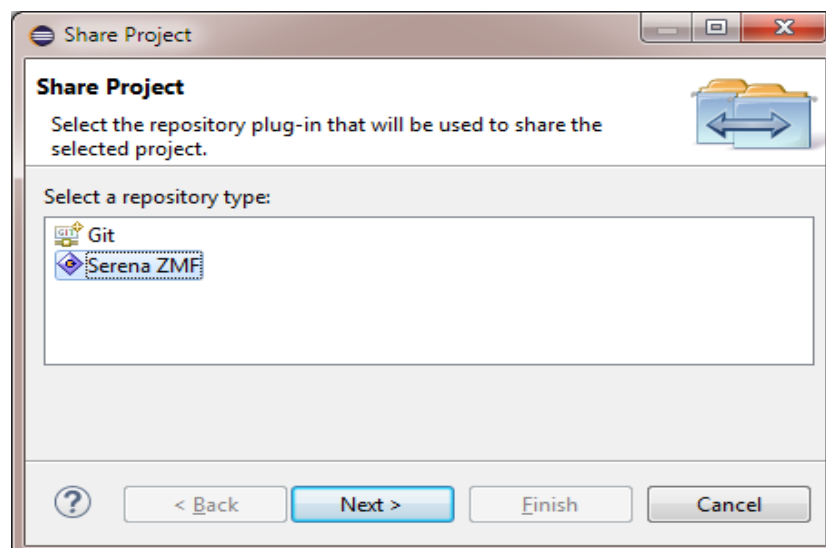
### Sharing Projects with ZMF

Java projects must be shared with a ChangeMan ZMF repository through the **Team** menu of the **Package Explorer** before you can access ZMF repository functions from the Java perspective. To share a project with ZMF:

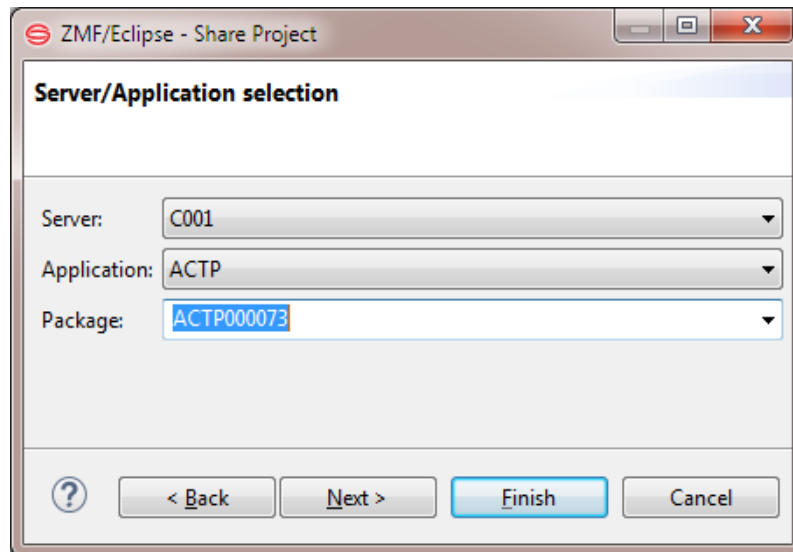
- 1 In the **Package Explorer** view, right-click on the project you want to share with ZMF. Then in its contextual menu, select **Team | Share Project**.



- 2 When prompted to choose a repository type, select **Serena ZMF** and click **Next**.



- When prompted to select a particular ZMF server and application, enter the following:



- **Server** — Select the desired ZMF repository (or site) from the drop-down list.
- **Application** — Select the ZMF application associated with your project.
- **Package** — Select the ZMF change package associated with your project (optional). If a package is not selected now, you will be prompted to select one at checkin later.



**NOTE** Be sure the application you select allows checkin from a personal development library. If it does not, you will not be able to check in your code later.

- Optionally, click **Next** to verify the mappings between the project folder names used in your Eclipse project and the library types defined in the ZMF repository.



**NOTE** Folder-to-library mappings can not be changed from the client. However, you can verify that the folder names in your Eclipse project are those that ZMF expects. This is necessary for the correct operation of **Checkin** and **Checkout**.

- Click **Finish** to enable sharing.

## Dynamic Integration with ChangeMan ZMF

A ChangeMan ZMF application or change package can be linked dynamically to a Java project. When dynamic integration is enabled, common actions in the Java perspective (such as saving an edited a component) automatically trigger appropriate change management activities (such as component checkout or checkin and staging from development) in ChangeMan ZMF.

## Enabling and Disabling Dynamic Integration with ZMF

A Java project is enabled for dynamic integration with ZMF in one of two ways:

- Select the **Share Project** option from the **Team** contextual menu in the **Package Explorer** view of the Java perspective.
- Create a Java project using the **Add to Workspace** function in the contextual menu for a specific ZMF application in the **Serena Explorer** view of the Serena perspective.

Dynamic integration with ZMF can be turned off for a project by selecting the **Unshare Project** option from the **Team** contextual menu in the **Package Explorer** view of the Java perspective.

## Dynamic ZMF Functions in the Java Perspective

The following ZMF functions are triggered automatically in shared projects while dynamic integration is enabled:

- **Logon** to the associated ZMF server is triggered by selecting a shared project in the **Package Explorer** view.
- **Checkout of a component** from ZMF baseline, staging that component from development to a change package, and optionally locking the component is triggered by changing the contents of a component in any editor within the Java perspective.
- **Mass checkin of all components** in a Java project to a newly created change package, with automatic mapping between workspace directories and ZMF libraries or folders, is performed when you select **ZMF Checkin** for a project (rather than a component) in the **Team** menu.

## ZMF Functions in the Package Explorer

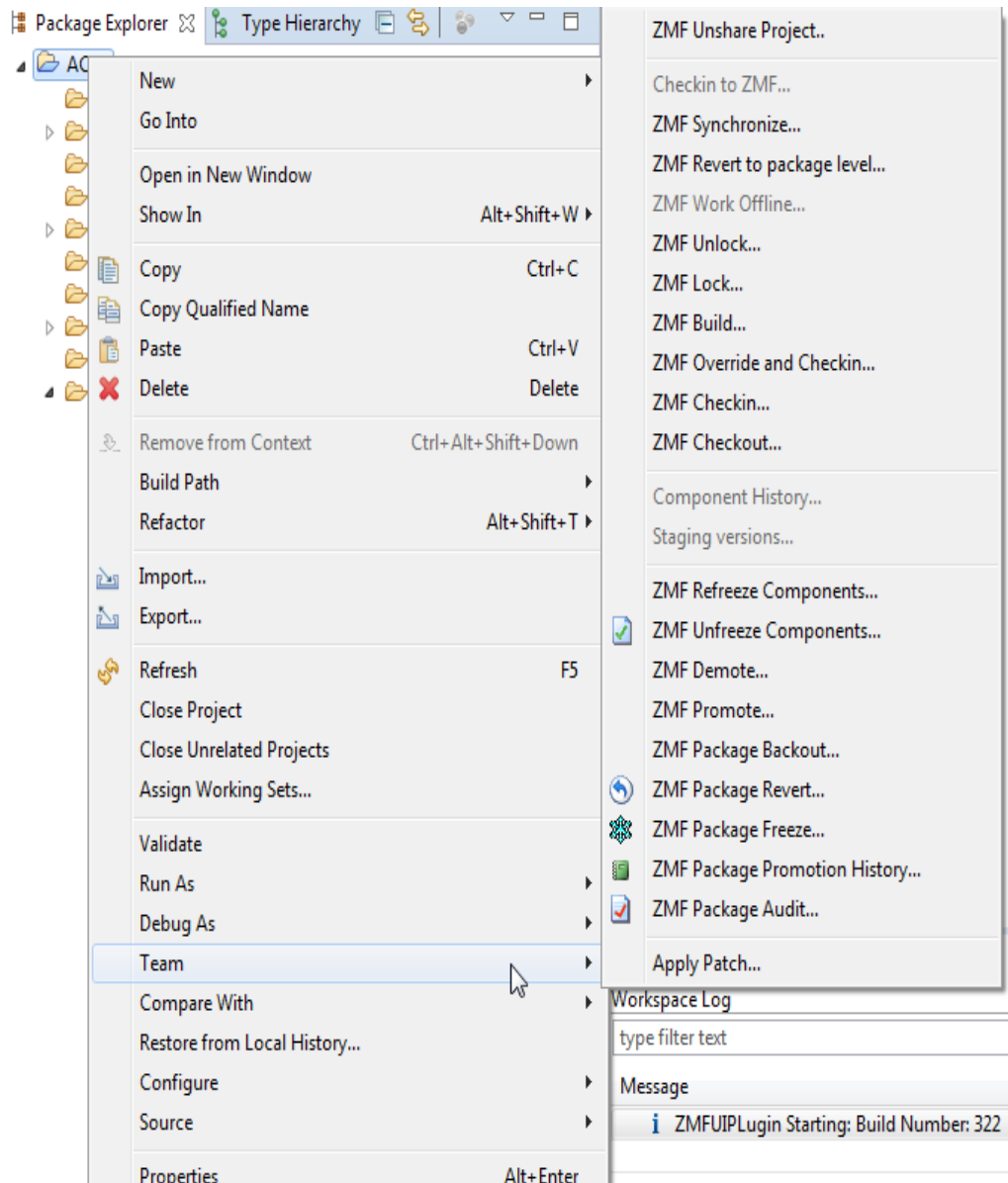
Team Menu and  
ZMF Functions

ChangeMan ZMF functions can be invoked in the Java perspective only for Java projects that are shared with a ZMF repository. These functions are invoked from the **Team** contextual submenu of the **Package Explorer** navigation view. Different ZMF functions are available for different Java objects:

- "Project-Level ZMF Functions"
- "Folder-Level ZMF Functions"
- "Component-Level ZMF Functions"

## Project-Level ZMF Functions

Project-Level ZMF functions for a shared Java project display in the **Team** contextual submenu.



Project-Level  
ZMF Functions

The following ZMF-specific functions are enabled for Java projects:

- **ZMF Unshare Project** — Unshares a Java project with ChangeMan ZMF and disables dynamic integration between the workbench and the ZMF repository for that project. Automatic checkouts and checkins to a change package will no longer take place every time you change a component in this project. See "[Dynamic Integration with ChangeMan ZMF](#)" on page 100 for more information about sharing a Java project with ZMF.
- **ZMF Synchronize...** - Asks you to Confirm Open Perspective and if you want to open this perspective now.
- **ZMF Revert to Package Level** — Reverts all components in this project (that is, reverts their corresponding components in a ZMF change package) to the promotion level of the change package as a whole in ChangeMan ZMF.

See [Chapter 6, "Reverting a Package to Development Status" on page 227](#) for more information on reverting a package.

- **ZMF Unlock** and **ZMF Lock** — Unlocks or locks all components in this project (that is, unlocks or locks their corresponding components in a ZMF change package) against change by other users.  
 See [Chapter 5, "Locking and Unlocking Components" on page 141](#) for step-by-step instructions.
- **ZMF Build** — Instructs ZMF to build all components in this project (that is, build their corresponding components in a ZMF change package).  
 See [Chapter 5, "Building a Component" on page 143](#) for step-by-step instructions.
- **ZMF Override and Checkin...** - Starts a dialog to checkin with override.
- **ZMF Checkin...** — Checks in all components in this project to a corresponding change package in a ChangeMan ZMF repository.  
 See [Chapter 5, "Checking In a Component" on page 134](#) for step-by-step instructions.
- **ZMF Checkout...** — Checks out all components in this project (that is, checks out their corresponding components in a ZMF change package) from the change package to a personal development library on the mainframe. The personal development library may reside in the native z/OS Partitioned Data Set (PDS) file system or in the z/OS Unix System Services (USS) Hierarchical File System (HFS).  
 If the personal development library is mapped to an RDz subproject, the **Checkout** operation has the effect of copying checked out components to that RDz subproject.  
 See [Chapter 5, "Checking Out a Component" on page 129](#) for step-by-step instructions.
- **ZMF Unfreeze Component** and **ZMF Refreeze Component** — Unfreezes or refreezes selected components within a frozen change package. Unfrozen components in the package are enabled for editing in the workbench. Refrozen components in the package are locked against editing in the workbench.  
  
 See [Chapter 6, "Unfreezing and Refreezing Package Components" on page 197](#) for step-by-step instructions.
- **ZMF Demote** — Demotes the ZMF change package associated with the Java project from a mainframe promotion site and level (for example, a testing environment).  
 See [Chapter 6, "Demoting a Package" on page 209](#) for step-by-step instructions.
- **ZMF Promote** — Promotes the ZMF change package associated with the Java project to a mainframe promotion site and level (for example, a testing environment).  
 See [Chapter 6, "Promoting a Package" on page 199](#) for step-by-step instructions.
- **ZMF Package Backout** — Backs out the ZMF change package associated with a Java project from all production sites and restores those sites to their pre-change states.  
 See [Chapter 6, "Backing a Package Out of Production" on page 225](#) for step-by-step instructions.
- **ZMF Package Revert** — Reverts a frozen or backed out change package associated with a Java project to development (DEV) status and enables further changes.  
 See [Chapter 6, "Reverting a Package to Development Status" on page 227](#) for step-by-step instructions.
- **ZMF Package Freeze** — Freezes the ZMF change package associated with a Java project, and all its components, against changes during testing or review.  
 See [Chapter 6, "Freezing a Package" on page 194](#) for step-by-step instructions.
- **ZMF Package Promotion History** — Displays the promotion history for the change package associated with a Java project.

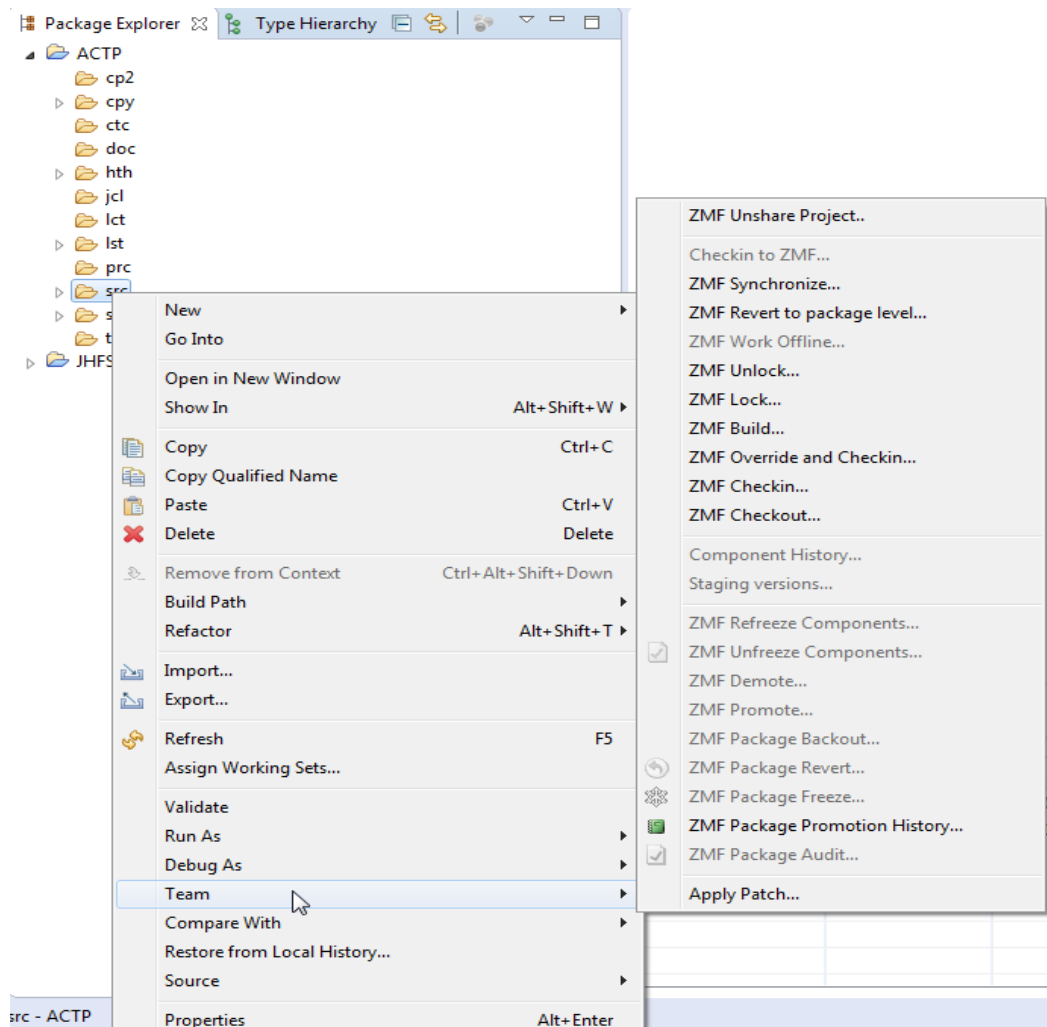
See [Chapter 6, "Viewing Promotion History" on page 239](#) for step-by-step instructions.

- **ZMF Package Audit** — Audits the ZMF change package associated with a Java project for component completeness and synchronization. See [Chapter 6, "Auditing a Package" on page 213](#) for step-by-step instructions.
- **Apply Patch** - Starts a Patch Input Specification dialog, On the first page you can select the file or folders on which to apply the patch and whether the patch is loaded from an external file or from the clipboard.



## Folder-Level ZMF Functions

Folder-Level Team Menu A subset of project-level ZMF functions are also available for folders within a Java project. These are invoked from the **Team** contextual menu of the Java perspective, as shown.



Folder-Level ZMF Functions

The following ZMF-specific functions are enabled for Java folders:

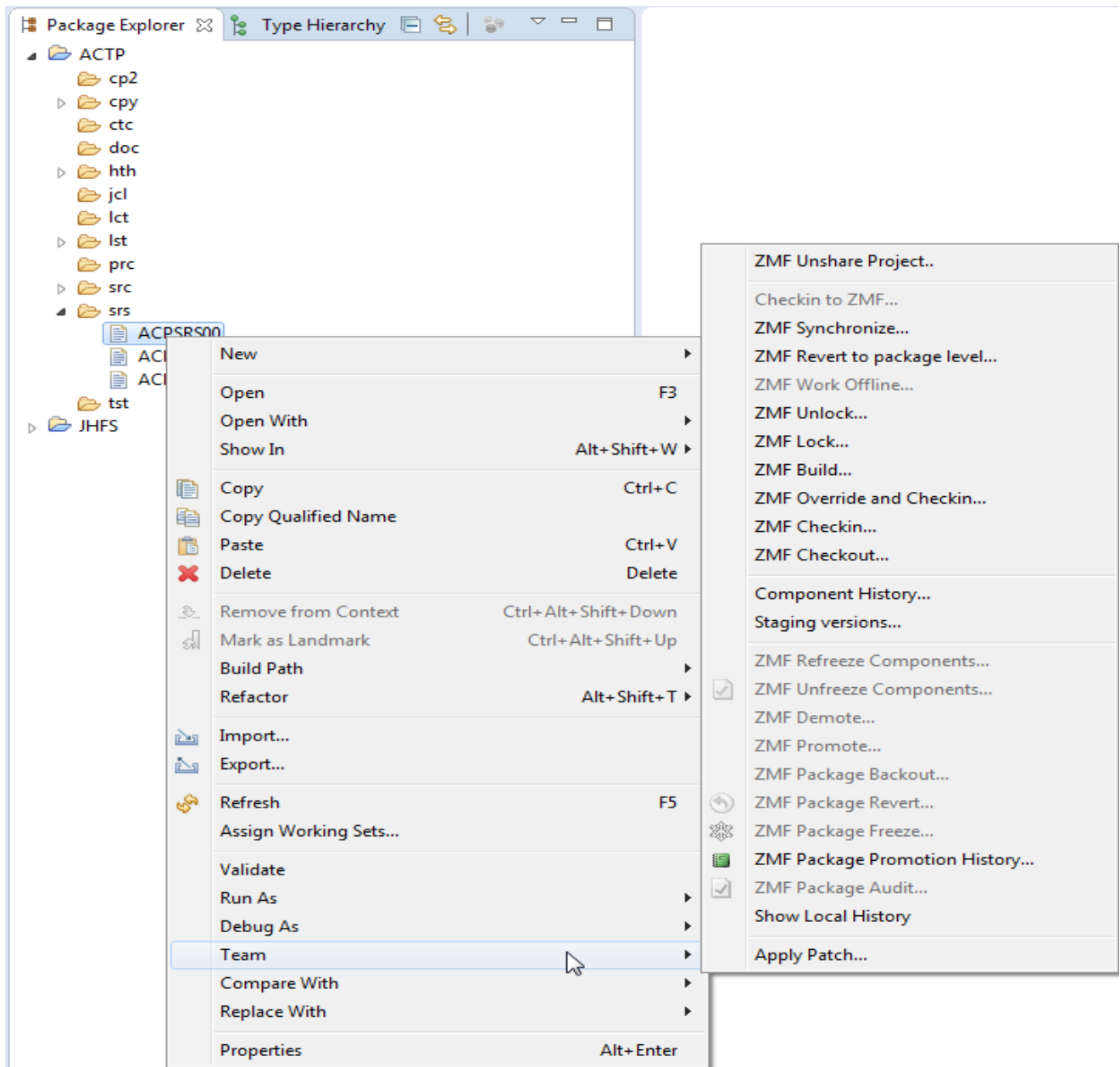
- **ZMF Unshare Project** — Unshares a Java project with ChangeMan ZMF and disables dynamic integration between the workbench and the ZMF repository for that project. Automatic checkouts and checkins to a change package will no longer take place every time you change a component in this project. See "[Dynamic Integration with ChangeMan ZMF](#)" for more information.
- **ZMF Synchronize...** - Asks you to Confirm Open Perspective and if you want to open this perspective now.
- **ZMF Revert to Package Level...** — Reverts all components in this folder (that is, reverts their corresponding components in a ZMF change package) to the promotion level of the change package as a whole in ChangeMan ZMF. See [Chapter 6, "Reverting a Package to Development Status" on page 227](#) for related information on reverting a change package to development status.
- **ZMF Unlock** and **ZMF Lock** — Unlocks or locks all components in this project (that is, unlocks or locks their corresponding components in a ZMF change package) against change by other users.

See [Chapter 5, "Locking and Unlocking Components" on page 141](#) for step-by-step instructions.

- **ZMF Build** — Instructs ZMF to build all components in this project (that is, build their corresponding components in a ZMF change package).  
See [Chapter 5, "Building a Component" on page 143](#) for step-by-step instructions.
- **ZMF Override and Checkin...** - Starts a dialog to checkin with override.
- **ZMF Checkin** — Checks in all components in this project to a corresponding change package in a ChangeMan ZMF repository.  
See [Chapter 5, "Checking In a Component" on page 134](#) for step-by-step instructions.
- **ZMF Checkout** — Checks out all components in this project (that is, checks out their corresponding components in a ZMF change package) from the change package to a personal development library on the mainframe. The personal development library may reside in the native z/OS Partitioned Data Set (PDS) file system or in the z/OS Unix System Services (USS) Hierarchical File System (HFS).  
If the personal development library is mapped to an RDz subproject, the **Checkout** operation has the effect of copying checked out components to that RDz subproject.  
See [Chapter 5, "Checking Out a Component" on page 129](#) for step-by-step instructions.
- **ZMF Package Promotion History** — Displays the promotion history for the change package associated with a Java project.  
See [Chapter 6, "Viewing Promotion History" on page 239](#) for step-by-step instructions.

## Component-Level ZMF Functions

Component-Level Team Menu Certain ZMF functions apply to components in a Java project. They are invoked from the **Team** contextual menu of the Java perspective, as shown below.



Component-Level ZMF Functions The following ZMF-specific functions are enabled for Java project components:

- **ZMF Unshare Project** — Unshares a Java project with ChangeMan ZMF and disables dynamic integration between the workbench and the ZMF repository for that project. Automatic checkouts and checkins to a change package will no longer take place every time you change a component in this project. See "[Dynamic Integration with ChangeMan ZMF](#)" on page 100 for more information.
- **ZMF Synchronize...** - Asks you to Confirm Open Perspective and if you want to open this perspective now.

- **ZMF Revert to Package Level** — Reverts all components in this folder (that is, reverts their corresponding components in a ZMF change package) to the promotion level of the change package as a whole in ChangeMan ZMF.  
See [Chapter 6, "Reverting a Package to Development Status" on page 227](#) for related information on reverting a change package to development status.
- **ZMF Unlock** and **ZMF Lock** — Unlocks or locks all components in this project (that is, unlocks or locks their corresponding components in a ZMF change package) against change by other users.  
See [Chapter 5, "Locking and Unlocking Components" on page 141](#) for step-by-step instructions.
- **ZMF Build** — Instructs ZMF to build all components in this project (that is, build their corresponding components in a ZMF change package).  
See [Chapter 5, "Building a Component" on page 143](#) for step-by-step instructions.
- **ZMF Override and Checkin...** - Starts a dialog to checkin with override.
- **ZMF Checkin** — Checks in all components in this project to a corresponding change package in a ChangeMan ZMF repository.  
See [Chapter 5, "Checking In a Component" on page 134](#) for step-by-step instructions.
- **ZMF Checkout** — Checks out all components in this project (that is, checks out their corresponding components in a ZMF change package) from the change package to a personal development library on the mainframe. The personal development library may reside in the native z/OS Partitioned Data Set (PDS) file system or in the z/OS Unix System Services (USS) Hierarchical File System (HFS).  
  
If the personal development library is mapped to an RDz subproject, the **Checkout** operation has the effect of copying checked out components to that RDz subproject.  
See [Chapter 5, "Checking Out a Component" on page 129](#) for step-by-step instructions.
- **Component History** — Displays the component history list, which shows all change packages associated with the selected components.  
See [Chapter 5, "Viewing Component History" on page 138](#) for step-by-step instructions.
- **Staging Versions** — Displays a list of backup copies of the selected, staged components in a package associated with the Java project. Backup copies may be browsed or compared from the **Staging Versions** table view.  
See [Chapter 5, "Viewing the Component Staging Versions List" on page 159](#) for step-by-step instructions.
- **ZMF Package Promotion History** — Displays the promotion history for the change package associated with a Java project.  
See [Chapter 6, "Viewing Promotion History" on page 239](#) for step-by-step instructions.

## Chapter 4

---

# Working with RDz Perspectives

z/OS Projects Perspective Overview	110
RDz Projects and Subprojects	110
Using the Smart Editor	111
Checking a Component into ZMF from an RDz Project	118

## z/OS Projects Perspective Overview

**z/OS Projects Perspective** Rational Developer for z Systems provides two workbench perspectives: the **z/OS Projects** perspective (for development and debugging work with RDz projects) and the **Remote Systems** perspective (for connecting to one or more remote z/OS systems and working with personal development libraries on those systems). Neither perspective is supplied with the open-source version of the Eclipse Workbench.

Although the main perspective for working with ChangeMan ZMF from the workbench is the new Serena perspective, developers of mainframe software will typically spend much of their time in one of these RDz perspectives. For their convenience, the ZMF for Eclipse plug-in duplicates certain ZMF-specific functions in the contextual menu for individual components in the **z/OS Projects** navigation view.

Otherwise, the integration between ChangeMan ZMF and RDz is passive rather than active. Any personal development library on the mainframe is visible to RDz without special action on the part of the ZMF for Eclipse plug-in. ZMF-specific functions involving these libraries are managed from the Serena perspective independently of RDz. But native RDz functionality is used to associate these development libraries with RDz projects. Once part of an RDz project, the same development libraries that rely on ZMF for change control and build management can also take advantage of RDz's rich suite of development and debugging tools for mainframe applications.

## RDz Projects and Subprojects

You can link your personal development libraries on the z/OS mainframe to any desired RDz project. When you do this, ZMF for Eclipse permits integrated ZMF checkouts to and checkins from RDz projects. This lets you take advantage of the mainframe development tools of RDz while working with ZMF-managed software assets.

### Navigating Logical Projects and Physical Libraries

RDz projects and subprojects are defined in the z/OS Projects perspective and are displayed in the **z/OS Projects** navigation view. They are logical rather than physical collections of resources that organize a development project for work with the mainframe development and debugging tools of RDz. Projects and subprojects point to the physical resources that are displayed in the **Remote Systems** view.

Because RDz projects and subprojects are *not* physical containers for files, you must map these logical resources to physical libraries before you can perform physical operations on them. For example, if you want to check out a ZMF component into an RDz project, you must first link one or more physical containers to that project so the checkout operation can have a physical target for the component.



**IMPORTANT!** You cannot check out a ZMF component directly into an RDz subproject. A project must be linked to a physical container, such as a PDS library or an HFS folder, so that checkout to a subproject can copy the checked out component into a library or folder to which the subproject is pointing.

## RDz Project Structure

RDz projects consist of subprojects, which point to libraries, which contain files or members that serve as software components in your application.

- **MVS Subprojects** contain native z/OS PDS libraries
- **z/OS Unix Subprojects** contain z/OS Unix HFS folders and files.

RDz projects allow you to work with PDS and HFS files collectively. However, many operations — such as component build procedures — cannot be applied to these different file systems in the same way. For this reason, most work with RDz projects occurs at the subproject level. Libraries and files or members, for example, are linked to subprojects, not projects, and the file system of the resource must match the file system supported by the subproject type.

Personal development libraries on the z/OS server must be connected to an RDz project before you can work with them in RDz. Once this is set up, you can use ZMF for Eclipse to check out components from a ZMF baseline library into a personal development library and its associated RDz project at the same time.

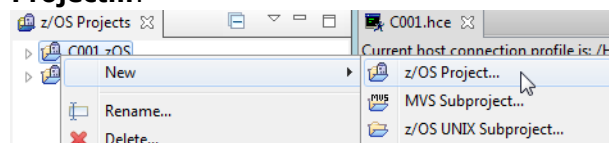
PDS development libraries in the native z/OS file system must be linked to an RDz project through an MVS subproject. HFS development libraries in the z/OS Unix System Services (USS) file system must be linked to an RDz project through a z/OS Unix subproject.

## Using the Smart Editor

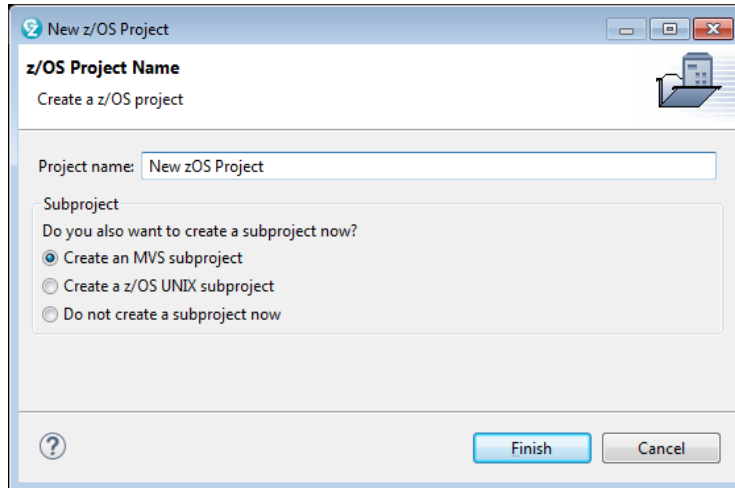
### Set up a z/OS Project and Subproject in RDz

**Project and Subproject** Before using the Smart Editor feature you must set up a z/OS Project and Subproject in RDz to use the Smart Editor.

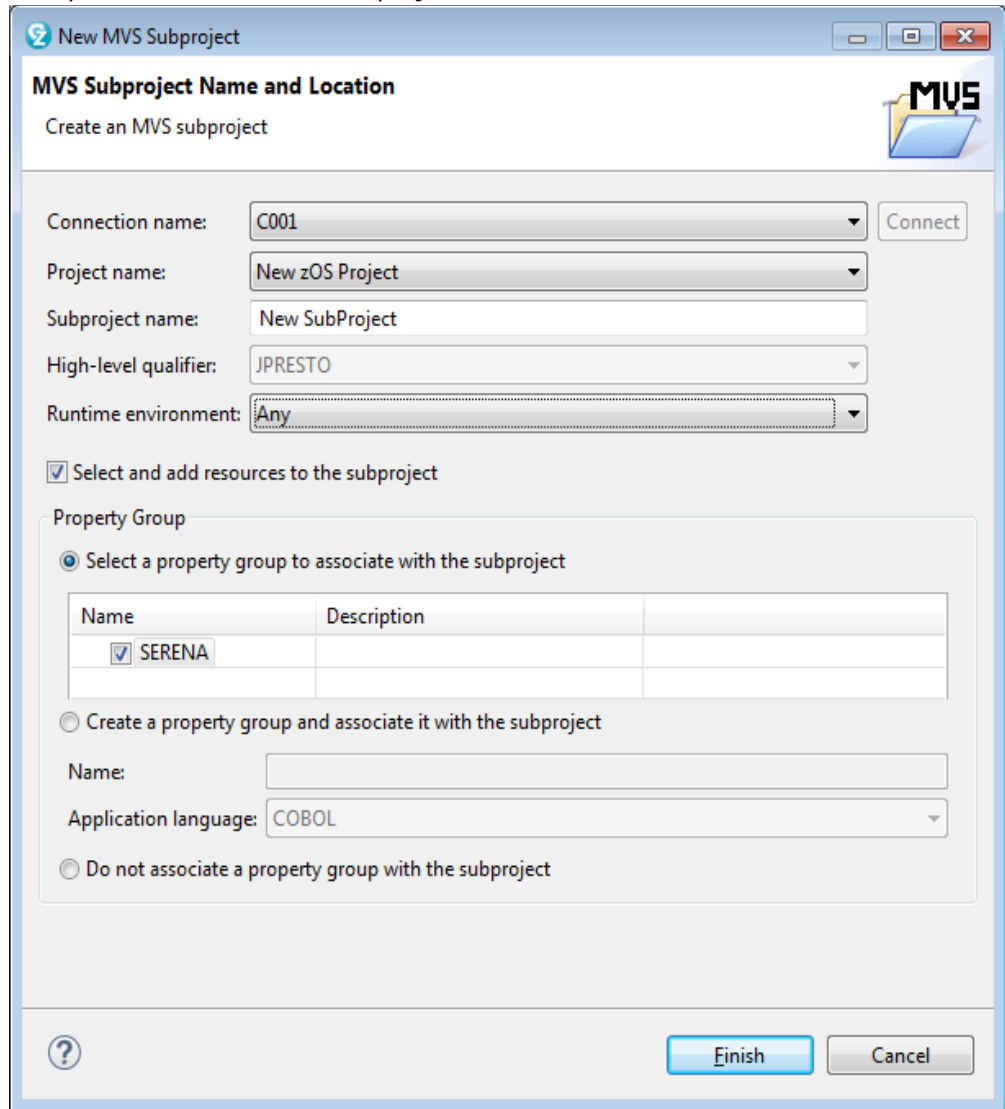
In the z/OS Projects perspective right click an existing project and select **New | z/OS Project...**



Then choose a Project Name:



Then you must decide on a subproject name and location:

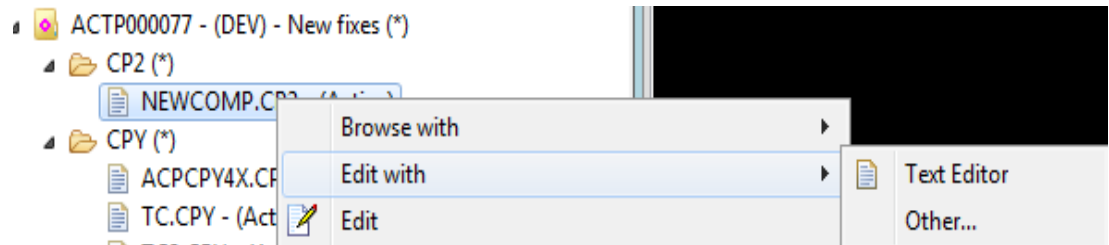


A Property Group must be defined to use the integration otherwise the user will get a 'SubProject does not have Property Group' error.



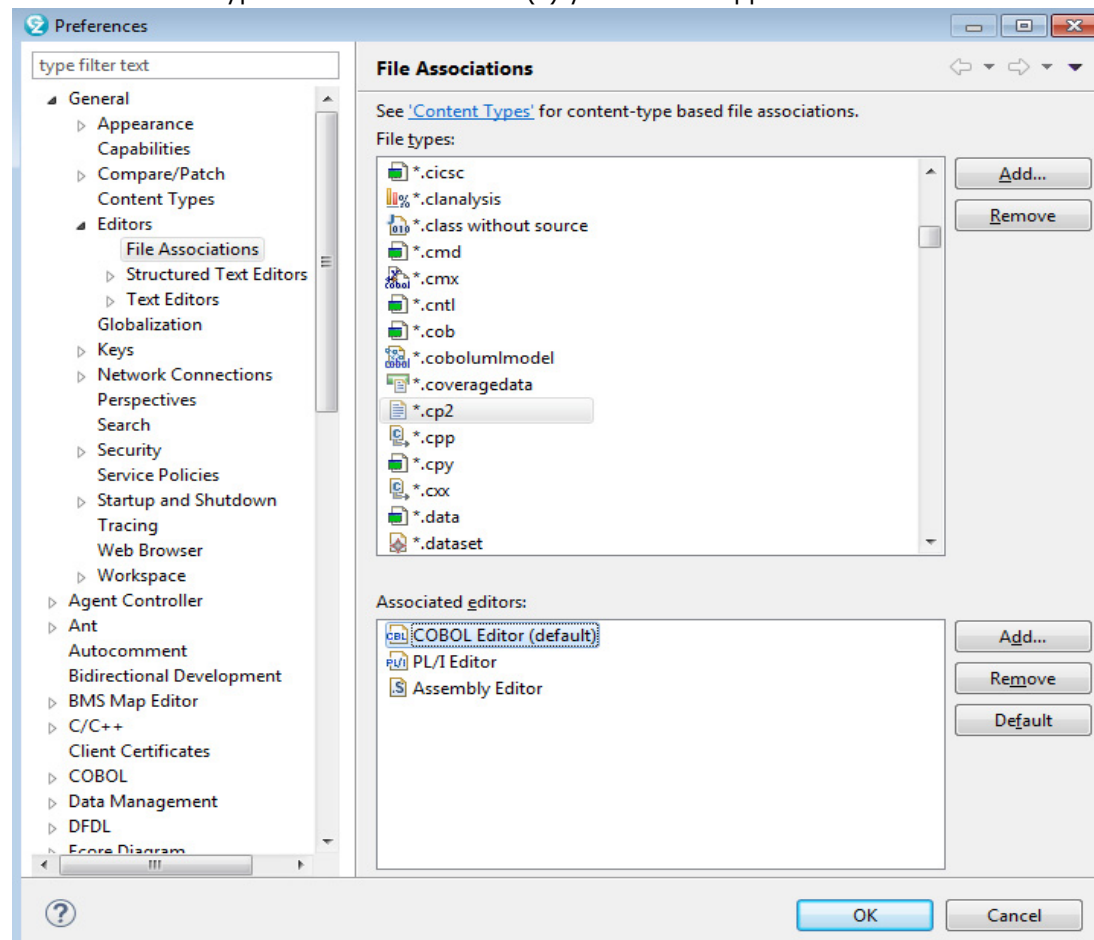
## Associate File Types

**Associate File Types** You can associate file types in RDz to enable the associated editors that appear directly on the menu rather than choosing from the 'Other' editors.



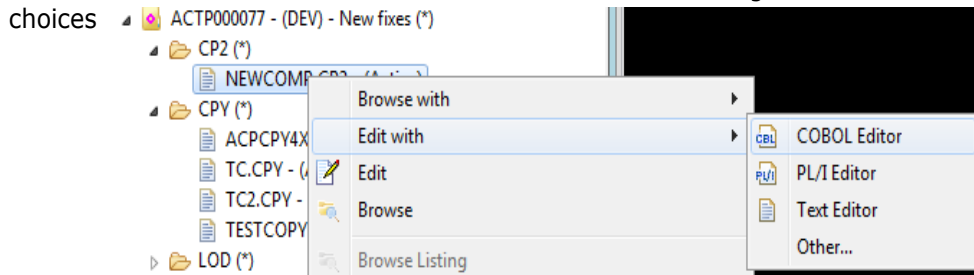
Window/  
Preferences/  
General/Editor

To do this, you use the **Window | Preferences | General | Editor | File Associations** and find the File type and select the editor(s) you want to appear:



Also note that the ZDDOPTS SYSLIB member should be defined so that property group overrides will work. See the ChangeMan ZMF for Eclipse 8.1.2 Installation and Configuration manual.

Menu of editor choices This will result in the menu of editor choices looking like this:

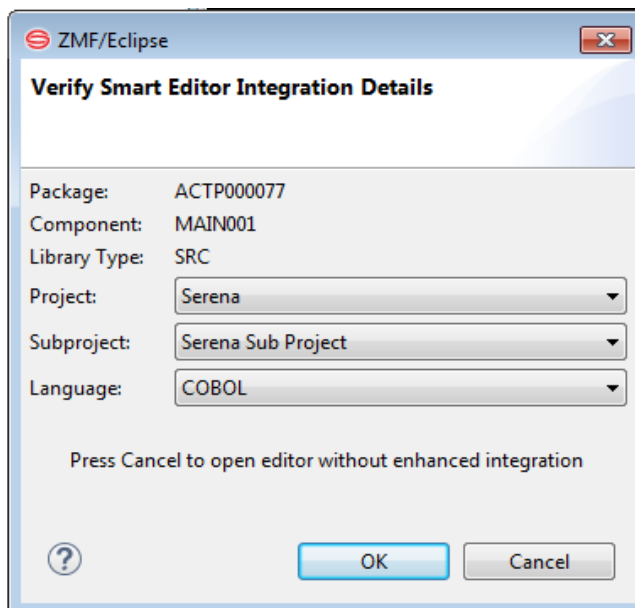


This allows your preferred editor to be chosen without using the **Other** option.

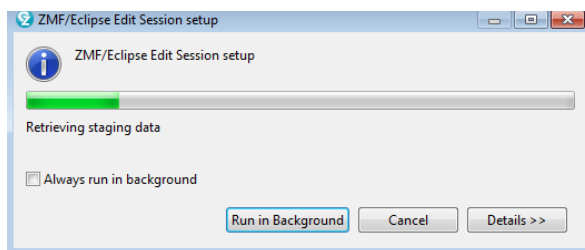
## Smart Editor Integration

Verify Smart Editor Integration Details

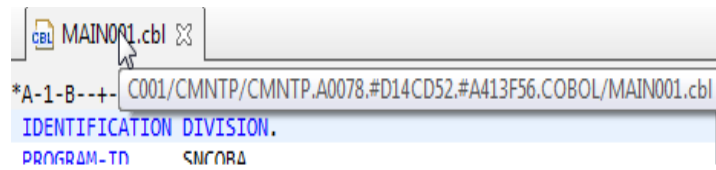
When you invoke the editor, for example the COBOL Editor, from within the Serena Perspective then you will be prompted for the details to use for the Smart Editor Integration. RDz Project, Subproject and Language:



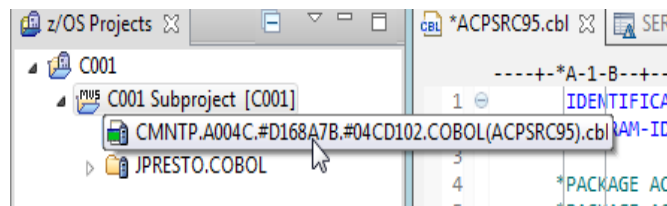
The Project and Subproject must already be defined and the default value will be taken from the Smart Editor Integration settings in the user's ZMF/Eclipse preferences section. If you click OK then the Smart Editor session will be set up, a z/OS Project temporary dataset will be created for the duration of your edit session, or click cancel to skip enhanced integration



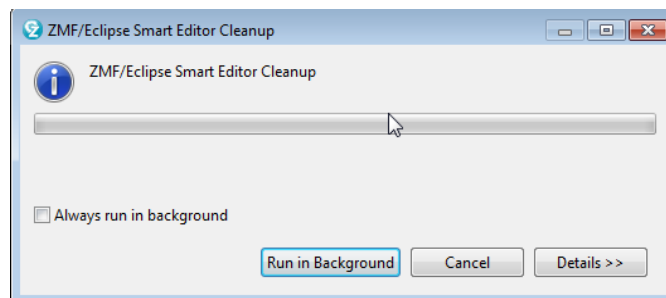
When the Smart Editor is running, the ZMF details can be seen by hovering the cursor over the component name:



If you look at data sets in the z/OS Projects you will see the temporary data set listed (with most of its properties shown Unavailable):

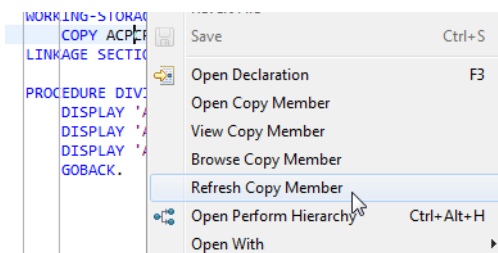


This data set will be removed when finished editing, and the Smart Editor session is automatically cleaned up:



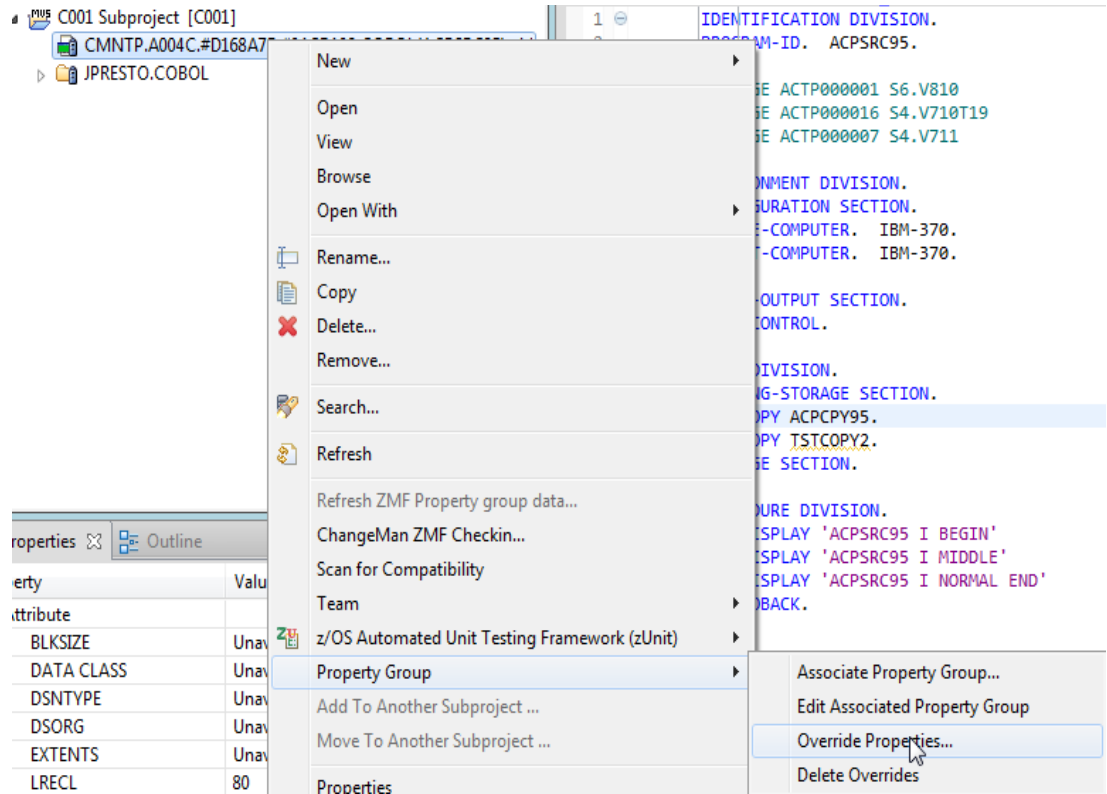
## Troubleshooting

If you are in the Smart Editor you will be able to see the contents of copybooks used from the SYSLIB concatenation (defined in the ZDDOPTS) by holding the cursor over the copybook name, but if a copybook has been changed since the edit session was started, you can right click the copybook name and select **Refresh Copy Member** to see the updated copybook member.

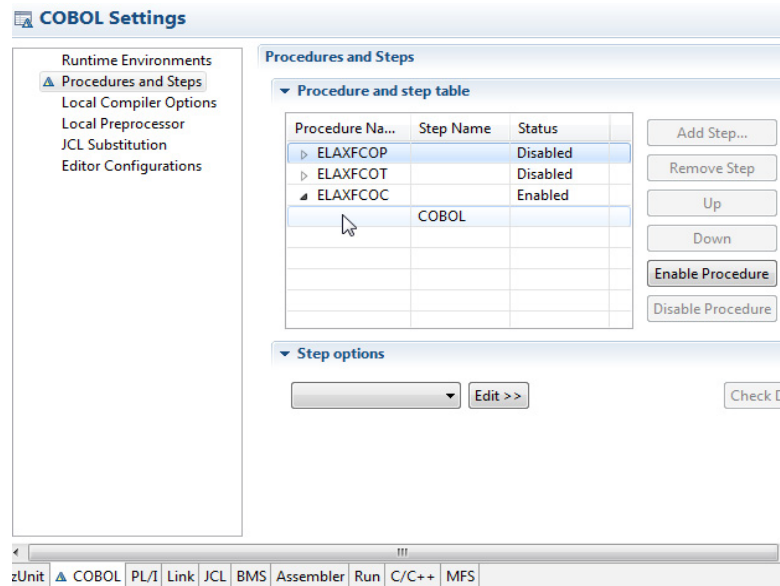


## Overriding or Displaying the Property Group values

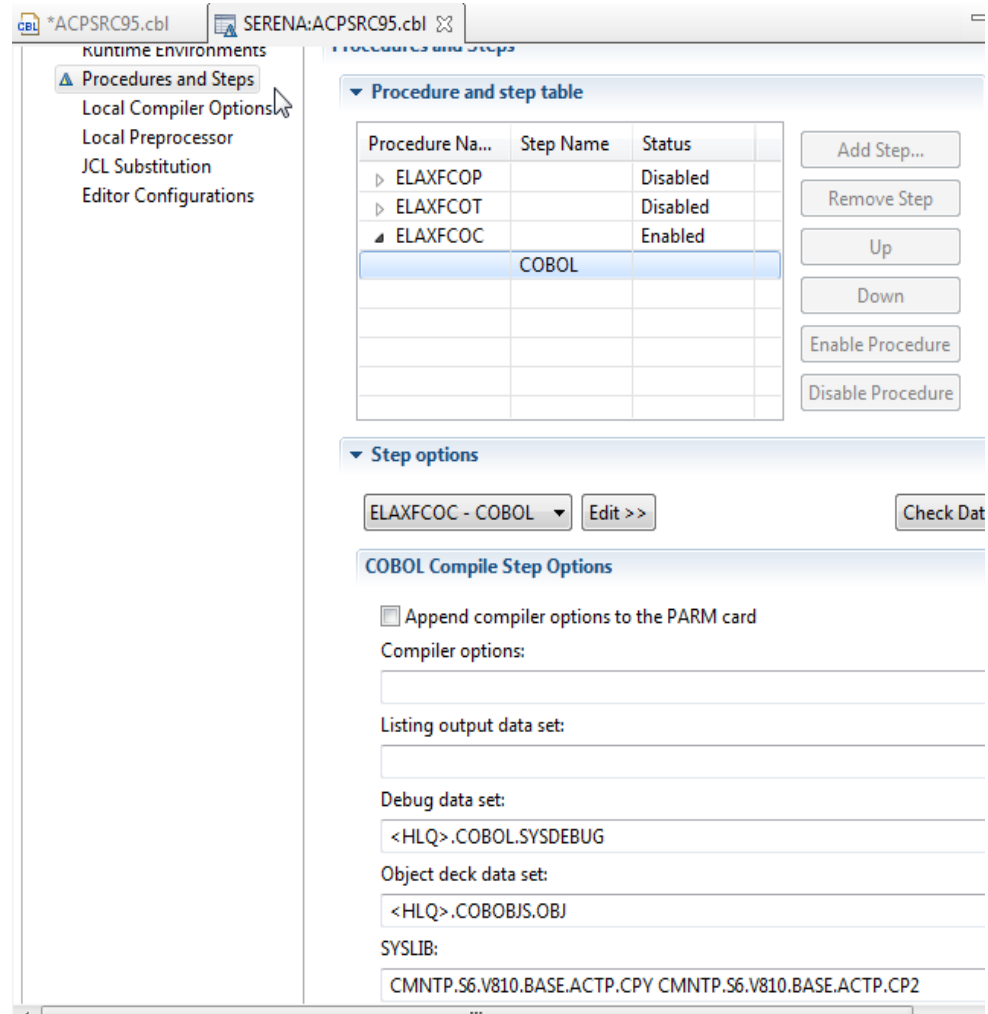
Whilst in edit in Serena Explorer you can look at Property Group values. To do so, click on z/OS Projects (top right) and then expand the subproject, then on the temporary dataset in use, right click and select **Property Group | Override Properties...**



In the next window, named **COBOL Settings**, then select the COBOL tab at the bottom of the frame (in this example it is COBOL source code) and then select (on the left) **Procedures and Steps**, expand the **Procedure Name** and click on the **Step Name**.



Then click on the COBOL Step Name and if you scroll down the step options you will see the box labeled SYSLIB - That will reflect the current data sets being used for your SYSLIB. You may only see baseline data sets (as shown here) if the package has no copybooks at the start of the edit.



If you require further assistance with the creation of z/OS Projects, Subprojects or File Associations then please refer to the appropriate IBM documentation and related support resources.

## Checking a Component into ZMF from an RDz Project

### Functional Description

**Component Checkin** Component checkin from an RDz project works exactly like **Checkin** from the Serena perspective. The **Checkin** option on the contextual menu for an RDz component enables you to check in a component from a personal development library on the z/OS server to a change package managed by ChangeMan ZMF. Both native z/OS PDS library members and z/OS Unix HFS files are supported. The **Checkin** function is similar to the "Stage from Development" function in ChangeMan ZMF.

**Component Builds** If the checked in resource is buildable, ZMF for Eclipse automatically prompts you for build job specifications at checkin. Default job cards and build jobs for the associated application are displayed in the prompt. You can omit the build step if desired.

Checkin from a development library is permitted only if the following criteria are met:

- The library type of the member must match a library type defined for the application.
- The component in the change package must not be locked by another TSO user ID.
- The component in the change package must not be a generated component.
- The package must be in DEV status and its install date must be today's date or later.

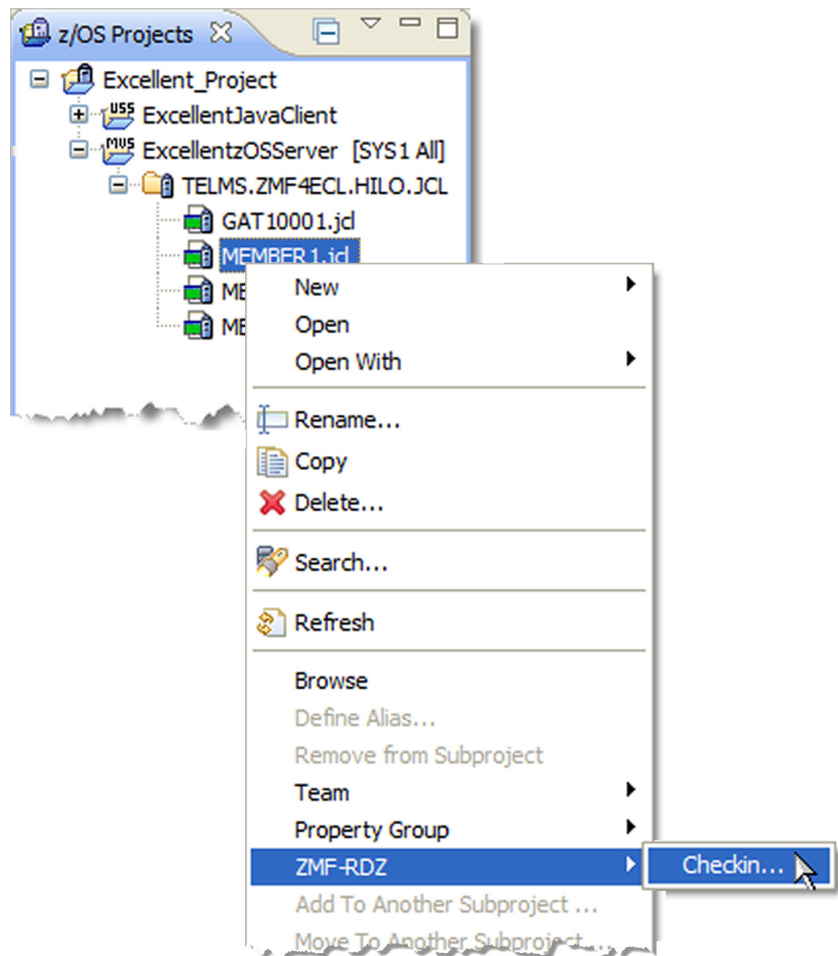


**CAUTION!** If another version of the component already exists in the target change package and is not locked, **it will be overwritten**. No warning message is displayed.

## RDz Component Checkin Procedure

To check a component into a ZMF change package from an RDz project and optionally submit it for build in ZMF, perform the following steps.

- 1 Open or switch to the **z/OS Projects** perspective if it is not already open.
- 2 Be sure you are logged on to z/OS in the **z/OS Projects** perspective and that you are also logged on to z/OS and ZMF in the **Serena** perspective.
- 3 If the component is open for editing and you aren't certain it has been saved, right-click on the editor window to bring up its contextual menu and click **Save**.
- 4 In the **z/OS Projects** navigation view, expand the project and subproject nodes for the component you want to check in and browse to the component.
- 5 Right-click on the component, then select **ZMF-RDz | Checkin** from the menu.



6 When the **Checkin** wizard displays, enter the requested information.



**IMPORTANT!** See [Chapter 5, "Checking In a Component"](#) on page 134 for step-by-step **Checkin** instructions.



## Chapter 5

---

# ChangeMan ZMF Component Functions

Creating a New Component	122
Deleting a Component	124
Scratching a Component under Change Control	125
Renaming a Component under Change Control	127
Checking Out a Component	129
Checking In a Component	134
Viewing Component History	138
Locking and Unlocking Components	141
Browsing a Component	141
Browsing a Component Listing	142
Editing a Component	143
Building a Component	143
Recompiling a Component	150
Relinking a Component	155
Viewing the Component Staging Versions List	159
Browsing Component Staging Versions	162
Comparing Component Staging Versions	162
Comparing Components to Baseline or Promotion	163
Viewing Component History	138
Source-to-Load Relationships	164
Component Bill of Materials	165
Component Impact Analysis	168
Query Component Functionality	171

## Creating a New Component

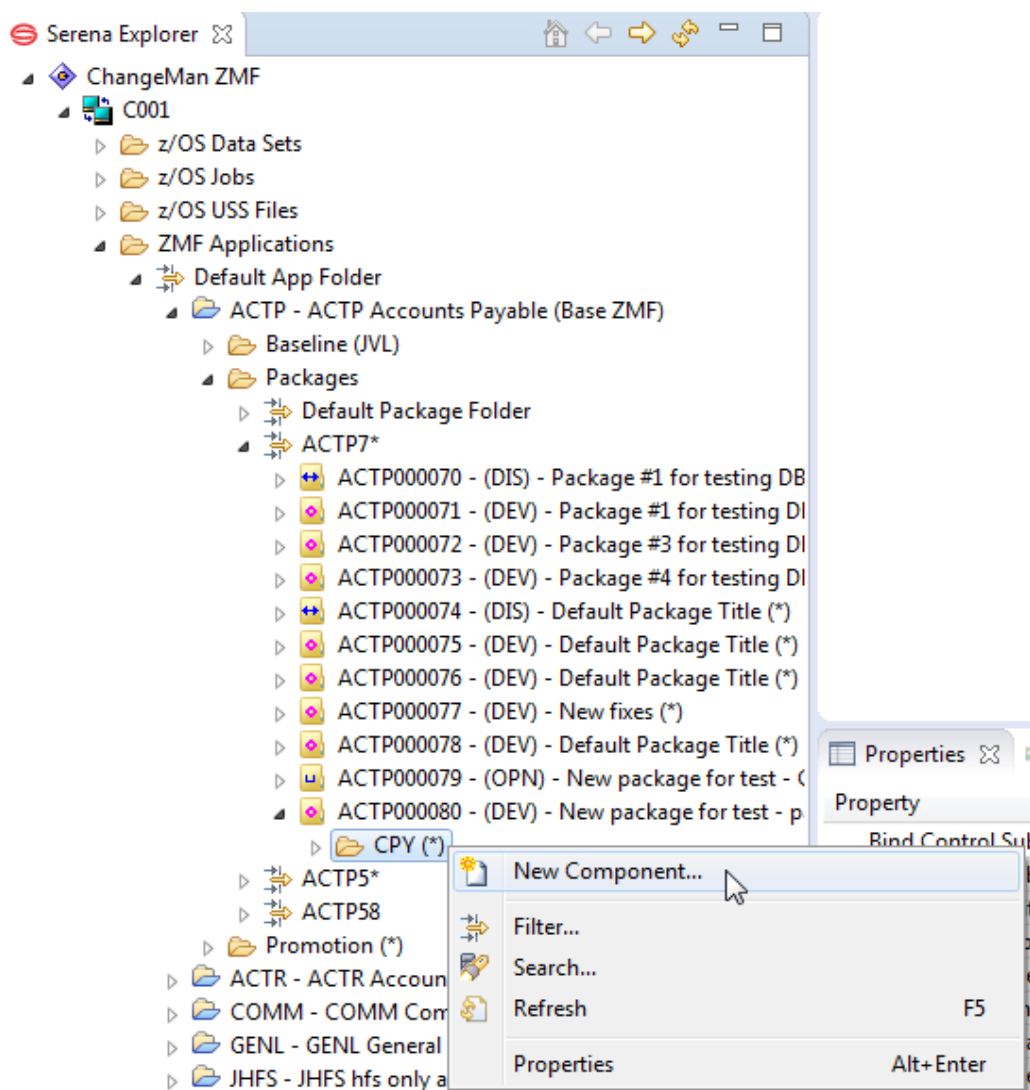
**Functional Description** A new software component can be created under change control in a ChangeMan ZMF package staging library using the **New Component** function of ZMF for Eclipse. The component may be either a PDS library member in the native z/OS file system or a file in the z/OS Unix System Services (USS) Hierarchical File System (HFS). Only one component can be created at a time.

### Invoking the Function and Viewing Results

**Invoking the New Component Function**

The **New Component** function is invoked from the following menu:

- Package staging library contextual menu** — In the **Serena Explorer** navigation view of the **Serena** perspective, find the ZMF server where the desired repository resides. Expand the server's **ZMF Applications** node, then the folder for the application to which the new component will belong. Under the **Packages** folder for that application, expand the particular change package that will contain the new component, then select the staging library where the new component will reside. Right-click on the library to bring up its contextual menu, then select the **New Component** option.



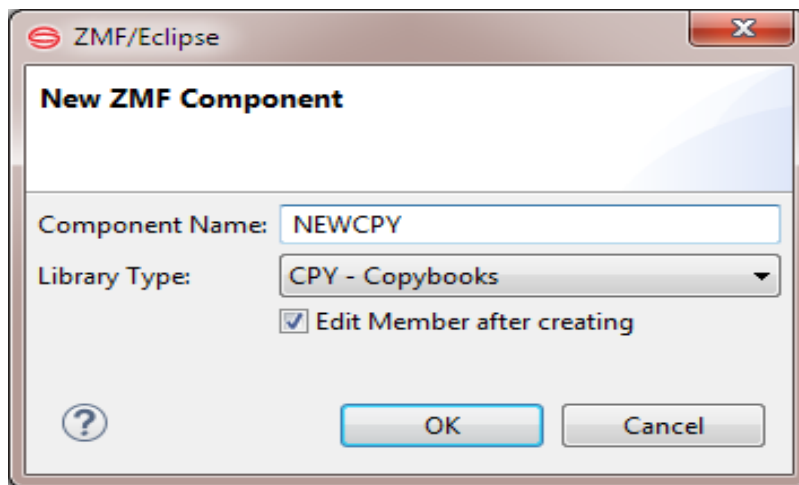
Viewing a New Component

The newly created component can be viewed in the following locations:

- **Serena Explorer navigation view** — In the **Serena Explorer** navigation view of the **Serena** perspective, right-click on the staging library in which the new component is expected to reside and select the **Refresh** option to refresh its contents. Then expand the staging library node. The newly created component should display in the component list for that library.
- **Package Components table view** — In the **Serena Explorer** navigation view, right-click on the change package containing the newly created component, then select the **Package Components** option from the package contextual menu. All components for the package are listed in table form under the tab for the **Package Components View**, which displays in the lower right pane of the perspective window.

## New Component Wizard Step-by-Step

- 1 When you invoke the **New Component** function, the **New ZMF Component** window displays.



- 2 Enter the following information:

Field/Checkbox	Description
<b>Name</b>	<ul style="list-style-type: none"> <li>■ For a new PDS member, type a member name with a maximum length of 8 characters. Names are not case-sensitive (that is, they are normalized to upper case). For example: Newmbr is converted to NEWMBR</li> <li>■ For a new HFS (USS) file, type the complete path and file name with qualifiers. Names are case-sensitive. For example: com/company/app/class/filename.java.JAV</li> </ul>
<b>Edit member after creating</b>	<ul style="list-style-type: none"> <li>■ Check this box to automatically invoke a workbench editor for the new component after it is created.</li> <li>■ Leave unchecked if you don't want to edit the component at this time.</li> </ul>

- 3 Click **OK** to create the component.

**NOTE** If the component already exists in the package, you will be given the option to overlay the existing version or to cancel the request.

## Deleting a Component

### Functional Description

In ZMF for Eclipse, the **Delete** function immediately deletes the target component in a package, the target library member in a PDS personal development library, or the target file in a Unix System Services (USS) Hierarchical File System (HFS) personal development library. The deletion is local in scope and is executed outside ZMF change control. For that reason it cannot be reversed. Multiple components may be deleted concurrently.

The contents of baseline libraries are not affected by the **Delete** operation.



**CAUTION!** To perform a versioned and reversible deletion of a baselined object under ZMF change control, use the **Scratch** function, not **Delete**.

See [Chapter 5, "Scratching a Component under Change Control" on page 125](#) for details.

### Invoking the Delete Function

The immediate **Delete** function is invoked from the contextual menus for each of the following object types.

**TIP** Instead of using the contextual menu, you may select an item and press the **Delete** key.

- *Serena package components* — In the **Serena Explorer** view of the **Serena** perspective, expand the **ZMF Applications** node and the folder to reach the desired application, then expand the **Packages** node. Navigate to the desired change package, then expand the desired package and staging library nodes to list its components. Select one or more desired components to delete, then right-click on the selection to bring up its contextual menu. Select the **Delete** option.

See [Chapter 2, "ZMF Operations on Package Components" on page 69](#) for more information.

- *Serena data set members* — In the **Serena Explorer** view of the **Serena** perspective, expand the **Data Sets** node, the appropriate viewing filter, and the desired PDS library to list its members. Select one or more members to delete, then right-click on the selection to bring up its contextual menu. Select the **Delete** option.

See [Chapter 2, "Working with z/OS Data Set Members" on page 46](#) for more information.

- *Serena USS (HFS) files* — In the **Serena Explorer** view of the **Serena** perspective, expand the **z/OS USS Files** node, the appropriate viewing filter, and the desired folder to list its subfolders and files. Select one or more objects to delete, then right-click on the selection to bring up its contextual menu. Select the **Delete** option.

See [Chapter 2, "Working with z/OS Unix HFS Files" on page 53](#) for more information.

- *Java perspective objects* — In the **Package Explorer** view of the **Java** perspective, expand the desired project, folder, and subfolder nodes to locate the desired objects

for deletion. Select one or more objects to delete, then right-click on the selection to bring up its contextual menu. Select the **Delete** option.

The local, immediate delete function is a native function of the workbench and is located on the main contextual menu for the object, not in the **Team** submenu

See [Chapter 3, "Component-Level ZMF Functions" on page 107](#) for more information.

- *z/OS project objects (RDz only)* — In the **z/OS Projects** navigation view of the **z/OS Projects** perspective, expand the desired project, folder, and subfolder nodes to locate the desired objects for deletion. Select one or more objects to delete, then right-click on the selection to bring up its contextual menu. Select the **Delete** option.

The local, immediate delete function is a native function of the workbench and is located on the main contextual menu for the object, not in the **ZMF-RDz** submenu.

## Scratching a Component under Change Control

### Functional Description

The **Scratch** function requests ZMF to set up a versioned delete (or *scratch*) operation on one or more baselined components. Control records containing instructions for the **Scratch** operation are stored in a change package for scheduled execution, and are applied to the appropriate baseline object(s) when the package is baselined.

The **Scratch** operation can be reversed using the **Backout** function.



**CAUTION!** To immediately delete a component without modifying any objects in baseline libraries, use the **Delete** function, not **Scratch**.

See ["Deleting a Component" on page 124](#) for details.

### Invoking or Canceling the Scratch Function

Invoking the  
Scratch Function

The **Scratch** function for ZMF components is invoked from the following menu:

- *Baseline component contextual menu* — In the **Serena Explorer** view of the **Serena** perspective, expand the **ZMF Applications** node and navigate to the desired application. Expand the application node and its **Baseline** node, then expand the node for the appropriate library to list its components. Select one or more components for versioned deletion, then right-click on the selection to bring up its contextual menu. Select the **Scratch** option.

See [Chapter 2, "ZMF Operations on Baseline Library Components" on page 67](#) for more information.

Cancelling a  
Scratch Operation

**Scratch** control records can be removed from a change package before the package is baselined and the **Scratch** operation is executed. Do this using the **Remove Scratch** function, which is invoked from the following menu:

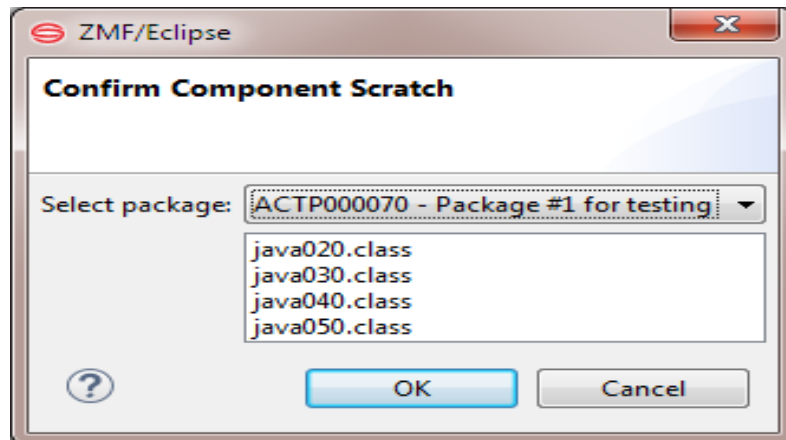
- *Package contextual menu* — In the **Serena Explorer** view of the **Serena** perspective, expand the **ZMF Applications** node and navigate to the desired application. Expand the application node and its **Packages** node, then scroll down to the package containing the **Scratch** control records. Right-click on the package name to bring up its contextual menu, then select the **Remove Scratch** option.

See [Chapter 2, "Package Contextual Menu Functions"](#) on page 64 for more information.

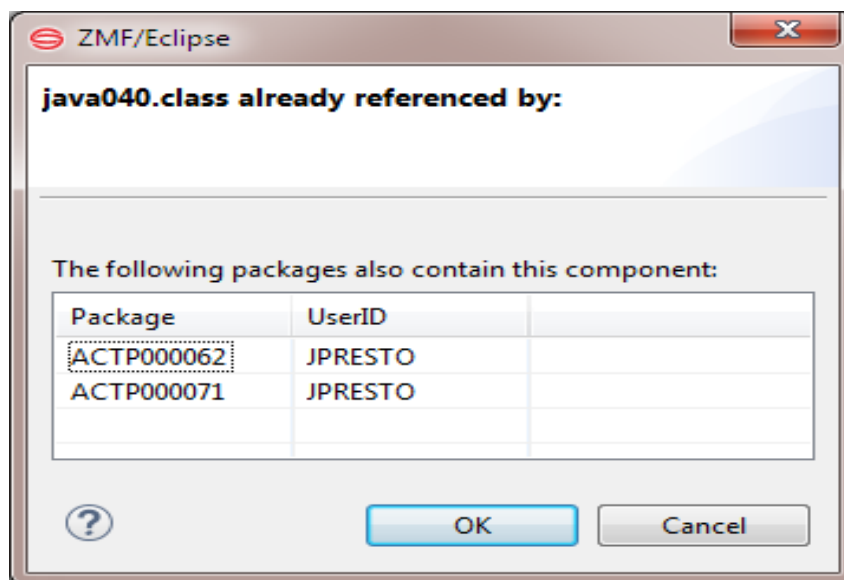
All scratch records in the package are removed by the **Remove Scratch** operation.

## Scratch Wizard Step-by-Step

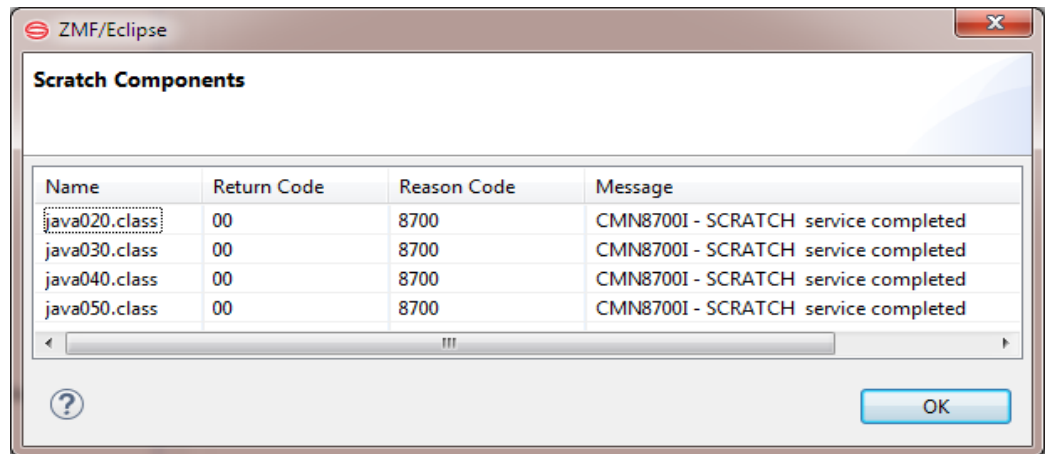
- 1 Select the baseline components to be scratched, then select **Scratch** from the contextual menu. The **Confirm Component Scratch** window displays.



- a Verify that the components listed in the text box are correct.
  - b From the **Select Package** pull-down list, chose the package to contain the scratch control records for a versioned deletion of these components from baseline. The chosen package will control deletion timing.
  - c Click **OK**.
- 2 For each selected component, the **Scratch** wizard will display a window listing all the change packages currently in motion that contain a version of that component. The component name is showed in the window title.



- a If you did not expect to see any packages associated with the named component, or if you are unsure how the install dates of the listed packages interact with the install date for the scratch package, click **Cancel** to exit the **Scratch** wizard.
  - b If the packages shown are those you expect to see associated with the named component, and you know how their install dates interact with the planned install date for the scratch package, click **OK** to proceed with the scratch operation.
  - c Repeat [Step 2](#) for each component selected for scratching.
- 3 When the final component has been **OK'd** for scratching, scratch instructions are placed in the change package and status messages are returned. For example:



## Renaming a Component under Change Control

### Functional Description

For baselined components, the **Rename** function sets up a versioned and reversible rename operation. Control records containing instructions for the **Rename** operation are stored in a change package for scheduled execution. They are applied to the appropriate baseline object(s) when the package is baselined.

The versioned **Rename** operation can be reversed using the **Backout** function.



**NOTE** ZMF for Eclipse does not permit the direct renaming of software components in change packages or in personal development libraries. However, you can delete a component and create a new component with a new name in its place, without affecting baselined objects.

### Invoking or Canceling the Rename Function

Invoking the  
Rename Function

The versioned **Rename** function for ZMF components is invoked from the following menu:

- *Baseline component contextual menu* — In the **Serena Explorer** view, expand the **ZMF Applications** node, the node for the desired application, its **Baseline** node, and the node for the library containing the component to be renamed. Right-click on the component to bring up its contextual menu, then select the **Rename** option.

See [Chapter 2, "ZMF Operations on Baseline Library Components" on page 67](#) for more information.

Canceling a Rename Operation

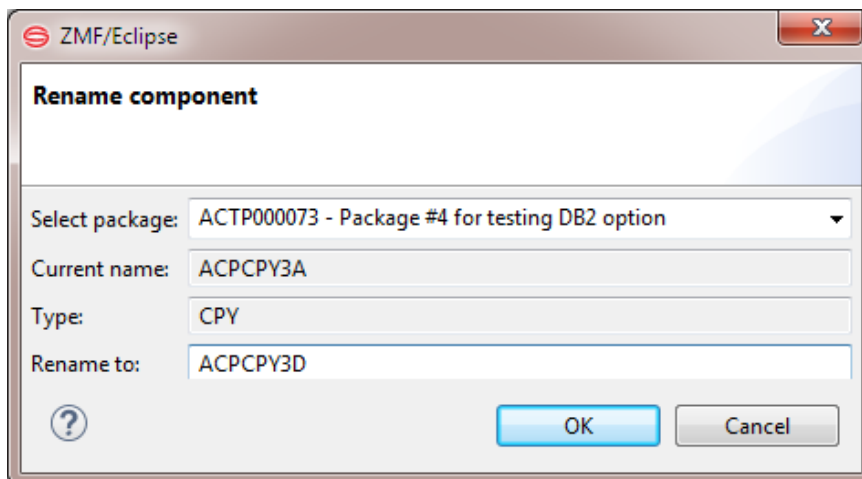
Versioned **Rename** instructions can be removed from a change package before the **Rename** operation executes. Do this using the **Remove Rename** function, which is located on the following menu:

- *Package contextual menu* — In the **Serena Explorer** view, navigate to the desired application and package containing the **Rename** control records. Right-click on the package to bring up its contextual menu and select the **Remove Rename** option.

See [Chapter 2, "Package Contextual Menu Functions" on page 64](#) for more information.

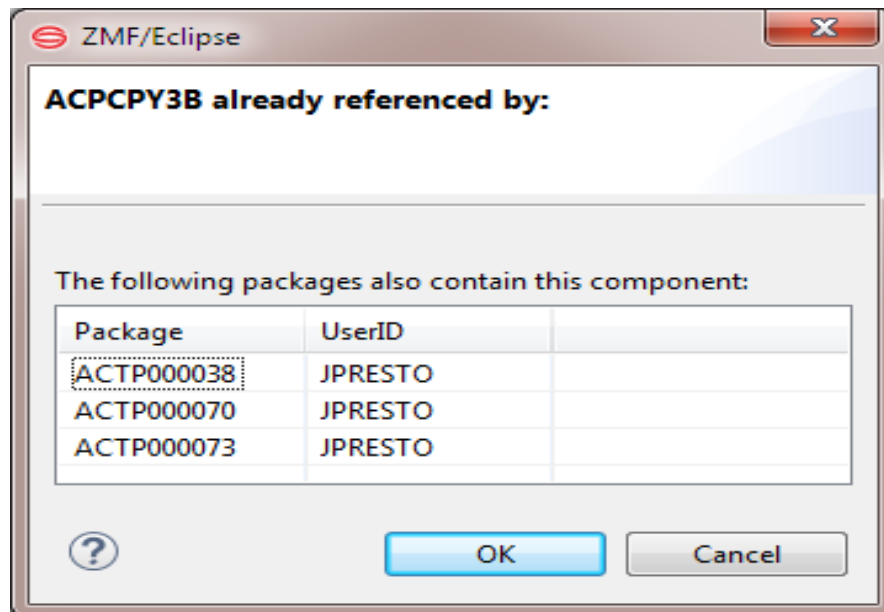
## Versioned Rename Wizard Step-by-Step

- 1 Select the baseline component to be renamed, then select **Rename** from the contextual menu. The **Rename Component** window displays.



- a Verify that the component name in the **Current Name** field and the library type listed in the **Type** field are correct.
  - b From the **Select Package** pull-down list, chose the package to contain the rename control records that will manage a versioned rename of this component in baseline. The chosen package will control the timing of the **Rename** operation.
  - c In the **Rename To** field, type the new name for the component.
    - If the component is a PDS member, type a member name with a maximum of 8 characters. The name is normalized to upper case and is not case sensitive.
    - If the component is a file managed by the z/OS Unix System Services (USS) Hierarchical File System (HFS), type the complete path and file name with all qualifiers or extensions. Long names are supported. Names are case sensitive.
  - d Click **OK**.
- 2 The **Rename** wizard displays a window listing all the change packages currently in motion that contain a version of the component to be renamed. The current component name is showed in the window title.





- a If you did not expect to see any packages associated with the component, or if you are unsure how the install dates of the listed packages interact with the install date for the versioned rename package, click **Cancel** to exit the **Rename** wizard.
- b If the packages shown are those you expect to see associated with the component, and you know how their install dates interact with the planned install date for the rename package, click **OK** to proceed with the **Rename** operation.

## Checking Out a Component

### Functional Description

- Checkout to a ZMF Change Package Software components may be checked out for revision from a baseline library or a promotion library and copied to staging libraries in a change package under the control of ChangeMan ZMF. This is done from the workbench using the **Checkout** function of ZMF for Eclipse. See the *ChangeMan ZMF User's Guide* for rules governing **Checkout**.
- Checkout to a Personal Library If personal development libraries are enabled in ZMF, the **Checkout** function in ZMF for Eclipse may also be used to copy components from a package staging library to a PDS or HFS personal development library on the mainframe. The personal development library feature also enables checkout of components from a package to an RDz subproject.
- Downloads with Retroactive Checkout In addition, an entire baseline library, promotion library, or change package managed by ZMF on the mainframe can be copied to a desktop project managed by the workbench using ZMF for Eclipse. Later, when you open a component in this set for revision, ZMF for Eclipse automatically invokes the **Checkout** function retroactively. Retroactive checkout can be initiated from contextual menus as well. (See "[Dynamic Integration with ChangeMan ZMF](#)" on page 100 for details.)
- Externally created components, components imported from another repository, or revisions made to a component outside a change package are checked in to a package using the **Checkin** function. (See "[Checking In a Component](#)" on page 134 for details.)

Workbench Differences The **Checkout** function works the same way in ZMF for Eclipse as it does in native ChangeMan ZMF, with these exceptions:

- You can check out components from a package library to a PDS or HFS personal development library on the mainframe or to an RDz subproject.
- If you check out a component in a library type for which Component Activity File (CAF) support is enabled, the corresponding component in the Component Activity File Type is not checked out. If you do not check out the corresponding component manually, the **Audit** function will detect a SYNCH6! error in the package.

See the *ChangeMan ZMF User's Guide* for additional information about the rules governing checkout.

## Invoking the Checkout Function

The **Checkout** function can be accessed from the following menus in ZMF for Eclipse:

- **Serena perspective** — In the **Serena Explorer** view, expand the **ZMF Applications** node. Below it, expand the desired node for **Baseline** libraries, **Packages**, or **Promotion** libraries. Navigate through the tree of libraries until you find the desired component(s) and select one or more of them to check out. Then right-click on the selection to bring up its contextual menu and choose the **Checkout** option.

See [Chapter 2, "Working with ZMF Components" on page 67](#) for more information.

- **Java perspective** — In the **Package Explorer** view, expand the desired project node and navigate through the folder tree until you find the desired component. Right-click on the component to bring up the contextual menu, select the **Team** submenu, then choose **ZMF Checkout**. Alternatively, navigate to a component in a project you created by download from ZMF and open that component. The **Checkout** function will be invoked automatically.

See [Chapter 3, "Component-Level ZMF Functions" on page 107](#) for more information.)

## Procedure for Checking Out a Component

To check out a component from a baseline, package, or promotion library, perform the following steps:

- 1 Invoke the **Checkout** function from the **Serena Explorer**, the **Package Explorer**, or by opening a component in a project created by download from ZMF.
- 2 When the **Checkout Parameters** window displays, identify the library to check out from, the package to check out to, and (optionally if you check the box) a personal development library on the mainframe (click browse to get a list, or new to create a new library) where the checked out component should be copied.
- 3 *Optionally*, click the **History** button on the **Checkout Parameters** window to bring up the **Component History** list. This list shows all the packages and promotion sites containing versions of the component(s) named in the **Component(s)** field of the **Checkout Parameters** window. This information can help you decide where the desired components should be checked out from. See ["Viewing Component History" on page 138](#) for details.
- 4 Click **Next** to display the next window, and then click **Finish** to perform the checkout.

## Checkout Parameters Window

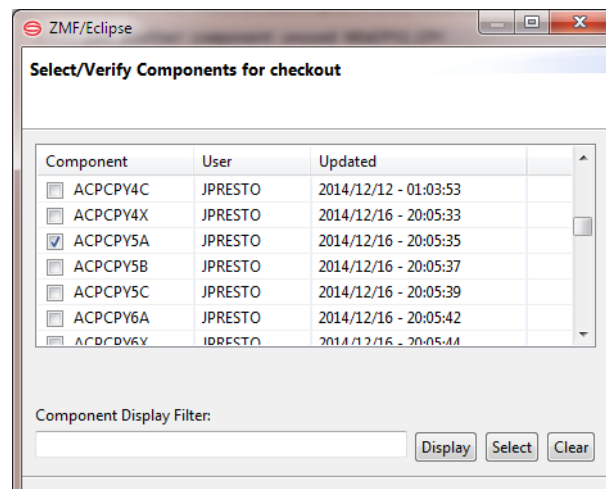
When you invoke the **Checkout** function, the **Checkout Parameters** window displays.

Fill in the fields as described in the following table.

Field or Box	Description
<b>Package</b>	<p>From the drop-down list, select the package to which the named component(s) should be checked out.</p> <p><b>NOTE:</b> Only packages with a current or future install date are shown in the drop-down list.</p> <p><b>TIP</b> Alternatively, click the <b>Filter</b> button to display the <b>Package Checkout Filter</b> dialog box to filter the package list by Creator, Work Request, or Department, thus producing a more manageable list</p>
<b>Component type</b>	This pull down allows you to select the type.

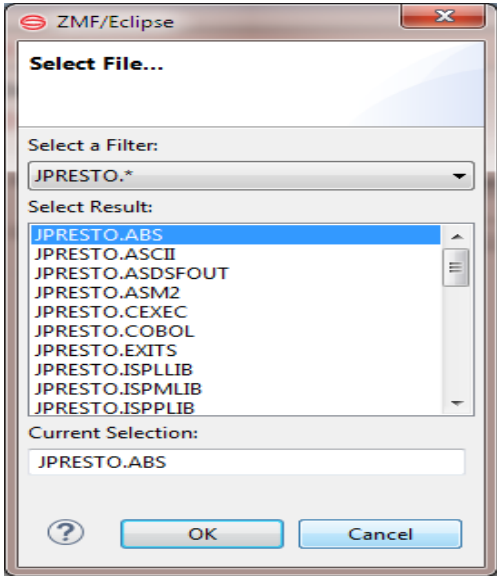
Field or Box	Description
<b>Check out From</b>	This section of the window allows you to simply use radio buttons and drop-down lists as appropriate to decide from where to check out the named component(s).
<b>Edit Component</b>	Check this box to open the component(s) for editing after checkout.
<b>Lock Component After Checkout</b>	Check this box to lock the component(s) after checkout. <b>NOTE:</b> You cannot lock a component that is already locked by another user.
<b>Checkout to Personal Library:</b>	If the checked out components should be copied to a personal development library on the mainframe, check this box.  To update the text field, enter a library name or click the <b>Browse</b> button to bring up the <b>Select File</b> window. You may also click the <b>New</b> button to display the <b>Allocate Data Set</b> dialog box to create a new library.

Then the second window asks you to **Select/Verify components for checkout**



## Select File Window

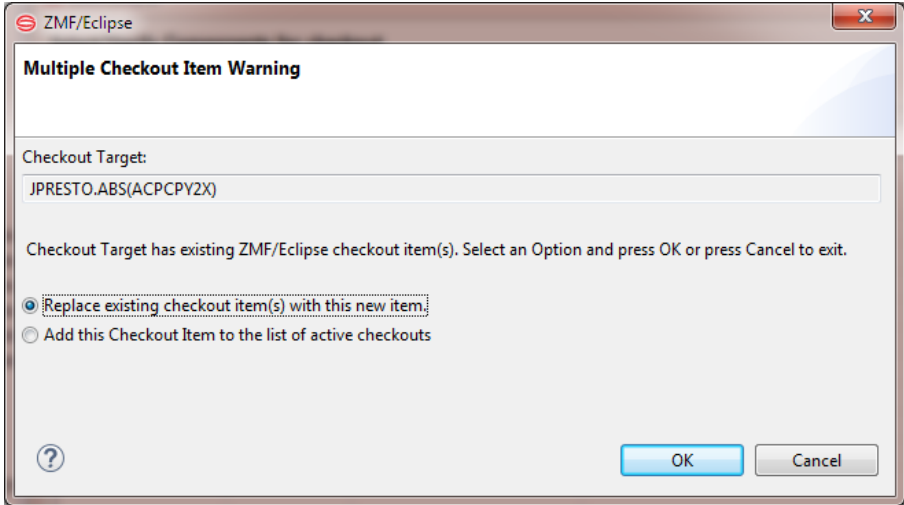
If you click the option to **Checkout to Personal Library** in the **Checkout Parameters** window, then click the **Browse** button, the **Select File** window displays. This window lets you select a personal development library on the mainframe using filename filters.



Fill in the fields as described in the following table, and click OK to action.

Field or Box	Description
<b>Select a Filter</b>	From the drop-down list, select the desired mainframe filename filter. Libraries that meet the filter criteria display in the text box.
<b>Select Result</b>	This text box shows all libraries on the mainframe that meet the naming criteria of the selected filter. Select the desired library from this list.
<b>Current Selection</b>	Shows the personal development library currently selected. The checked out component will be copied to this library.

**Note** that if the component is present already you will receive a **Multiple Checkout Item Warning** window, where you can decide what to do further.



# Checking In a Component

## Functional Description

**Checkin to a Staging Library** After creation or revision outside ChangeMan ZMF, software components may be checked in to a staging library in a ZMF change package using the **Checkin** function. **Checkin** works with components that reside in a PDS or HFS (USS) personal development library on the mainframe, in an RDz subproject on the mainframe, or in a desktop workspace managed by the workbench. Multiple components may be checked in concurrently.

The **Checkin** function in ZMF for Eclipse is also known as the **Staging** function in the ISPF interface to ChangeMan ZMF.

**Building Components** By default, a source code component is built automatically at **Checkin** (in other words, it is transformed into an executable module). However, you have the option to check the component into a package without building it.

**Checkin Rules** In general, if a component was checked out of a ZMF repository for revision outside of change control, the revised version must be checked in to the same repository, application, package, and library type that was associated with it at checkout. On the other hand, if a component is created outside of ZMF, its initial checkin to change control may target any compatible location.

The following rules apply when checking in a component:

- The file extension must match one of the library types defined for the ZMF application.
- The version of the component in the repository must not be locked by another user.
- The component must not be a generated component in the target package.
- The target package for checkin must be in development (DEV) status.
- The install date for the target package must be today's date or later.

See the *ChangeMan ZMF User's Guide* for additional rules governing checkin/staging.

## Invoking the Checkin Function

The **Checkin** function can be invoked from the following menus:

- **Serena perspective** — In the **Serena Explorer** navigation view, expand the **Data Sets** node or the **z/OS USS Files** node and navigate to the desired component, then right-click on it to bring up its contextual menu. Select the **Checkin** option.

See [Chapter 2, "Checking In and Building a Data Set Member" on page 49](#) for more information.

- **Java perspective** — In the **Package Explorer** navigation view, navigate the project tree to the desired component(s), right-click on the selection to bring up its contextual menu, select the **Team** submenu, then select the **Checkin** option.

See [Chapter 3, "Component-Level ZMF Functions" on page 107](#) for more information.

- **z/OS Projects perspective (RDz only)** — In the **z/OS Projects** navigation view, expand the appropriate subproject node and right-click on the desired component to bring up its contextual menu. Select the **Checkin** option.

See [Chapter 4, "Checking a Component into ZMF from an RDz Project" on page 118](#) for more information.

## Procedure for Checking In a Component

To check in a component to a package staging library, perform the following steps:

- 1 Invoke the **Checkin/Stage** function from the **Serena Explorer**, the **Package Explorer**, or the **z/OS Projects** navigation views.
- 2 When the **Checkin ZMF Components** window displays, identify the component(s) to check in; the library to check in from; the server, application, package, and staging library to check in to, and provide a change description. Optionally, you can turn off component build and/or component locking in ChangeMan ZMF at this time.
- 3 *Optionally*, click the **History** button on the **Checkin ZMF Components** window to bring up the **Component History** list. This list shows all the packages and promotion sites containing versions of the component(s) named in the **Component(s)** field of the **Checkin ZMF Components** window. This information can help you decide where the new component(s) should be checked in. See "[Viewing Component History](#)" on [page 138](#) for details.
- 4 Click **OK** to perform the checkin.



**CAUTION!** Existing package components with the same name as your new component **will be overwritten** at checkin. **No warning message** is displayed.

- 5 If you selected the option to **Launch Build Process**, the **Build** dialog box displays. Supply the requested information and click **OK** to initiate the build. See "[Building a Component](#)" on [page 143](#) for information on completing the **Build** dialog box.

**NOTE** The **Launch Build Process** option is enabled only for LIKE-SRC and LIKE-OTH library types.

## Component Checkin Wizard - Main Window

When you invoke the **Checkin** function in ZMF for Eclipse, the **Checkin ZMF Components** window displays.

**ZMF Checkin: 1 Component(s)**

Component(s) - update stage name in list below

Name	Stage Name
ACPCPY60	ACPCPY60

From: CMNTP.S6.ACTP.CPY

Server: C001

Application: ACTP

Package: ACTP000027

Library Type: JVS

Comment: new

Launch Build Process  
 Delete Source from Personal  
 Lock Component(s) after Checkin  
 Retain Checkout Details in Eclipse

OK Cancel

Fill in the fields as described in the following table.

Field or Box	Description
<b>Component(s)</b>	Displays the component(s) selected for checkin. Edit the text field as desired to add or remove components for checkin. All components must be of the same library type. Click the <b>History</b> button at right of this field to bring up the <b>Component History</b> list.
<b>Source Library</b>	Displays the library in which the selected component(s) currently reside.



Field or Box	Description
<b>Server</b>	<p>Displays the server hosting the ZMF repository where the checked in components will reside after checkin.</p> <ul style="list-style-type: none"> <li>■ If any components in the <b>Component(s)</b> field were originally checked out from a ZMF repository, the server hosting that repository is shown and cannot be changed.</li> <li>■ If the components in the <b>Component(s)</b> field originated outside ZMF, your currently connected server displays. This value may be changed. Click the <b>Servers</b> button to select from a list of ZMF servers in a pop-up window.</li> </ul>
<b>Application</b>	<p>Supply the 4-character name of the application to associate with the checked-in components.</p> <ul style="list-style-type: none"> <li>■ If any components in the <b>Component(s)</b> field were originally checked out from ZMF, the application from which they were checked out is shown and cannot be changed.</li> <li>■ If the components in the <b>Component(s)</b> field are newly created or otherwise originated outside ZMF, no application is displayed. Click the <b>Applications</b> button to list available applications in a pop-up window for selection.</li> </ul>
<b>Package</b>	<p>Supply the ID of the change package where the components should be checked in.</p> <ul style="list-style-type: none"> <li>■ If any components in the <b>Component(s)</b> field were originally checked out from ZMF, the package from which they were checked out is shown. This value cannot be changed.</li> <li>■ If the components in the <b>Component(s)</b> field are all newly created or otherwise originated outside ZMF, no package is displayed. Click the <b>Packages</b> button to bring up a list of available packages in a pop-up window, then select the target package for checkin.</li> </ul>
<b>Library Type</b>	<p>Supply the library type to receive the checked-in components. All selected components must share the same library type.</p> <ul style="list-style-type: none"> <li>■ If any components in the <b>Component(s)</b> field were originally checked out from a ZMF repository, their library type at checkout is shown. This library type cannot be changed.</li> <li>■ If the components in the <b>Component(s)</b> field are newly created or originated outside ZMF, no library type is shown. Click the <b>Library Types</b> button to bring up a list of library types in a pop-up window for selection.</li> </ul>
<b>Comment</b>	<p>Type a component change description for ZMF. At least one character is required.</p>
<b>Launch Build Process</b>	<p>Check this box to automatically launch the component <b>Build</b> process after checkin. You will be prompted to supply build parameters when checkin completes.</p> <p><b>NOTE</b> This option is enabled only for LIKE-SRC and LIKE-OTH library types.</p> <p>(See "<a href="#">Building a Component</a>" on page 143 for details.)</p>

Field or Box	Description
<b>Lock Component(s) After Checkin</b>	Check this box to lock the component(s) after checkin. <b>NOTE:</b> You cannot lock a component that is already locked by another user.
<b>Retain Checkout Details in Eclipse</b>	Check this box to retain component checkout/checkin metadata as component defaults for future <b>Checkout</b> dialog boxes.

When ready, click **OK** to initiate the **Checkin** process.

## Viewing Component History

### Functional Description

The **Component History** list retrieves the change history for one or more selected components. This history includes all the packages and promotion sites that have contained a version of the component, the timestamps associated with each change state, the build procedure executed on the component at checkin, and the users who have worked on that component. The selected component(s) may reside in baseline, promotion, or package staging libraries.

Component history information can help you decide which version of the component to check out or which package a software component should be checked out to or checked in to for revision. It can also help you diagnose problems or identify users who might have important information about a change.

### Invoking and Viewing the Component History List

The **Component History** list displays in slightly different ways, depending on how it is invoked.

- The **History** button in the main **Checkout** and **Checkin** dialogs displays component history for all components named in the **Component(s)** field of the **Checkout Parameters** window or the **Checkin ZMF Components** window. The results are shown in a pop-up window.

See ["Checking Out a Component" on page 129](#) and ["Checking In a Component" on page 134](#) for more information on the **History** button.

- The **Component History** option in a component contextual menu displays the component history for a single selected component. This option may be selected for baseline, package, and promotion components in the **Serena Explorer** navigation view of the **Serena** perspective. It may also be selected for Java projects shared with ZMF in the **Team** submenu of the **Package Explorer** navigation view in the **Java** perspective. The results are shown under the **Component History** table view tab in the bottom right pane of the perspective where the function is invoked.

For more information on the **Component History** option, see the following topics:

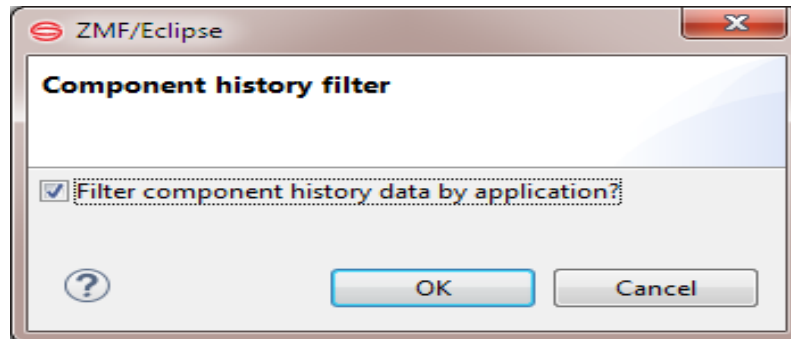
- [Chapter 2, "ZMF Operations on Baseline Library Components" on page 67](#)
- [Chapter 2, "ZMF Operations on Package Components" on page 69](#)
- [Chapter 2, "ZMF Operations on Promoted Components" on page 73](#)

- Chapter 3, "Component-Level ZMF Functions" on page 107

## Component History during Checkout and Checkin

When you invoke the **Component History** function by clicking the **History** button [...] to the right of the Component: area in the main **Checkout** or **Checkin** windows, the **Component History** function displays two pop-up windows in the following order.

- 1 **Component History Filter** — This dialog asks whether the component history list should be filtered by application.



- a By default, filtering by application is turned on. If component filtering by application is not desired, click the checkbox to deselect application filtering.
  - b Click **OK**.
- 2 **Component History** — For each selected component named in the **Component(s)** field of the main **Checkout** or **Checkin** window, the **Component History** window lists all the packages and promotion sites containing a version of the component.

Components are sorted by name within library type and are identified in the **Package** column in the following form:

*libtype:component*

The packages containing the identified component are listed immediately below the library/component identifier. For example:

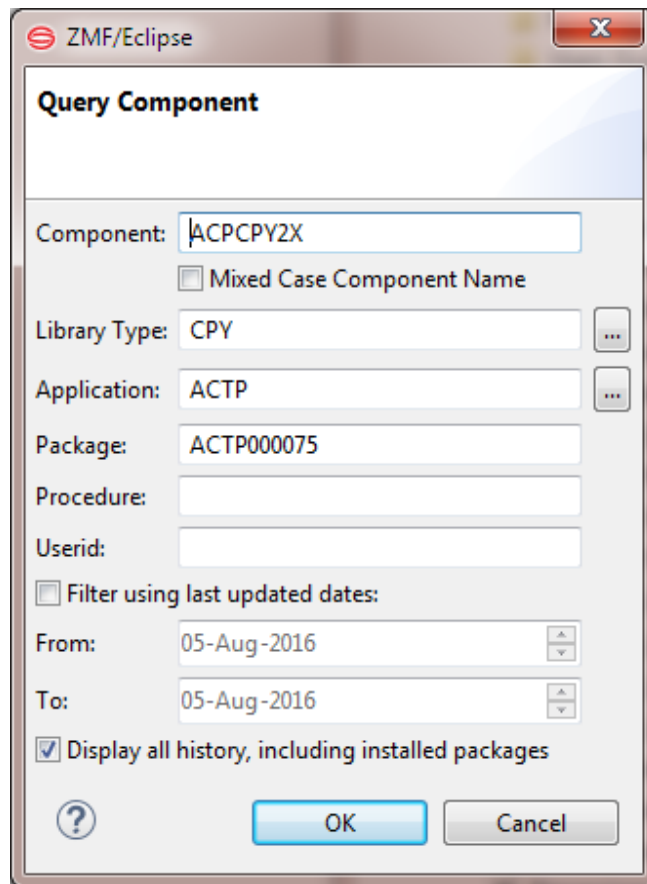
Package	Promotion	VV.MM	Last Action	Size	Procname	User
CPY: ACPCPY2X						
ACTP000073		01.01	2016/07/28 21:33:07	4		JPRESTO
ACTP000075		01.03	2016/08/04 21:31:15	5		JPRESTO

- a Scroll through the list to find each component of interest. Review the packages associated with the component as needed.
- b Click **OK** to return to the **Checkout Parameters** window or the **Checkin ZMF Components** window.

## Component History in Contextual Menus

When you invoke the **Component History** function from a component contextual menu in the **Serena Explorer** navigation view, the following events occur:

- 1 The **Query Component** dialog displays. This dialog asks whether the component history list should be filtered by application.



- a By default, filtering by application is turned on. If component filtering by application is not desired, erase the application to deselect application filtering, or you can select from a list of applications via the [...] button
  - b Select or deselect the check box to **Filter using last updated dates**, as desired.
  - c Select or deselect the check box **Display all history including installed packages** as desired.
  - d Click **OK**.
- 2 The **Query Component** table view displays under a self-named tab in the lower right pane of the current perspective. The component name and library type is shown in the banner area below the tab. The packages and promotion areas containing a version of the component are shown in the table. For example:

Component	Library Type	Package	Package St...	Type	Promotion	VV.MM	Date	Size
ACPCPY2X	CPY	ACTP000075	DEV			01.04	2016/08/04 21:45:16	5

Review the packages associated with the component as needed.

## Locking and Unlocking Components

Functional Description

The **Lock** function restricts revision privileges for a ZMF package component exclusively to the user who sets the lock. The **Unlock** function turns off the **Lock** restriction, enabling revision by multiple developers. Multiple components may be locked or unlocked at once.



**PRIVILEGES** Only a ChangeMan ZMF administrator can unlock a component that was locked by a different user.

Invoking the Lock and Unlock Functions

The component **Lock** and **Unlock** functions can be invoked from the following menus:

- **Serena perspective** — In the **Serena Explorer** navigation view of the **Serena** perspective, expand the **Applications** and **Packages** nodes, then locate the change package and staging library containing the desired component(s). Select one or more components, right-click on the selection to bring up its contextual menu, then choose the **Lock** or **Unlock** option. The operation takes effect immediately.

See [Chapter 2, "ZMF Operations on Package Components" on page 69](#) for more information.

- **Java perspective** — In the **Package Explorer** navigation view of the **Java** perspective, expand the project and folder containing the desired subfolder(s) or component(s). Select one or more subfolders or components, right-click on the selection to bring up its contextual menu, select the **Team** submenu, then select the **ZMF Lock** or **ZMF Unlock** option. The operation takes effect whenever a selected workbench component is checked in to a change package. For example, opening a downloaded component for edit will trigger an automatic **Checkin** and **Lock** in the background.

See [Chapter 3, "Component-Level ZMF Functions" on page 107](#) for more information.

Package components can also be locked by clicking the appropriate checkbox during **Checkout** or **Checkin**.

## Browsing a Component

Functional Description

The **Browse** function provides read-only viewing access in a workbench editor to software components residing on the mainframe in a ChangeMan ZMF repository. Components locked by another user may nevertheless be browsed in read-only mode.

Components may be browsed in baseline libraries, package staging libraries, and promotion libraries.

Invoking the  
Browse Function

Invoke the **Browse** function in the **Serena** perspective for the following object types:

- *Baseline components* — In the **Serena Explorer** navigation view, expand the **ZMF Applications** node, the folder for the desired application, the **Baseline** node, and the node for the library where the desired component resides. Right-click on the desired component to bring up its contextual menu, then select the **Browse** option. The component opens in an editor tab in the upper right pane of the perspective.

See [Chapter 2, "ZMF Operations on Baseline Library Components" on page 67](#) for more information.

- *Package components* — In the **Serena Explorer** navigation view, expand the **ZMF Applications** node, the folder for the desired application, and the **Packages** node. Find the package where the desired component resides, expand it, and expand the appropriate staging library within it to show a list of components. Right-click on the desired component to bring up its contextual menu, then select the **Browse** option. The component opens in an editor tab in the upper right pane of the perspective.

See [Chapter 2, "ZMF Operations on Package Components" on page 69](#) for more information.

- *Promotion components* — In the **Serena Explorer** navigation view, expand the **ZMF Applications** node, the folder for the desired application, and the **Promotion** node. Find the promotion site where the desired component resides, expand it, and expand the appropriate promotion library within it to show a list of components. Right-click on the desired component to bring up its contextual menu, then select the **Browse** option. The component opens in an editor tab in the upper right pane of the perspective.

See [Chapter 2, "ZMF Operations on Promoted Components" on page 73](#) for more information.

You can also invoke the **Browse** function for the most recent baseline version and all staging versions of a package component in any perspective where the **Staging Versions** table view is displayed.

- *Component staging versions table view* — Under the **Staging Versions** tab in the bottom right pane of the perspective, click the checkbox for the component version you wish to browse and uncheck all other versions. Then right-click on the table to bring up the contextual menu and select the **Browse** option.

See ["Viewing the Component Staging Versions List" on page 159](#) for more information.

## Browsing a Component Listing

Functional  
Description

Developers may choose to save a compile listing or other printable reports for a package component in a ZMF repository LST library on the mainframe. Saved listings may be browsed in a workbench editor directly by navigating to the listing member and using the component **Browse** function.

However, if you are working with a component source member, you do not need to navigate away from it to see its listing. Instead, you can open the listing using the **Browse Listing** navigation shortcut. **Browse Listing** maps the member name of the current component to the LST library type, then opens the corresponding listing member (if it exists) in a separate tab using the listing editor.

Invoking the  
Browse Listing  
Function

The **Browse Listing** navigation shortcut is invoked in the **Serena** perspective from the following menu:

- *Package source component contextual menu* — In the **Serena Explorer** navigation view, expand the **ZMF Applications** node, the node for the desired application, and the **Packages** node. In the package where the desired component resides, find the desired component in a *like-source* staging library — that is, a library defined to ZMF as like-source and identified with a library type qualifier such as SRC, CBL, PLI, or JAV. Right-click on the desired component to bring up its contextual menu, then select the **Browse Listing** option. If a listing member for that components exists in the LST library for the package, it opens in an editor tab in the upper right pane of the perspective.

See [Chapter 2, "ZMF Operations on Package Components" on page 69](#) for more information.

## Editing a Component

Functional  
Description

The **Edit** function provides workbench editor access for revisions to software components residing on the mainframe in a ChangeMan ZMF repository. The workbench text editor supports page-level editing and other features that make it much easier to use than the ISPF line editor on the mainframe.

When dynamic integration is enabled, saving a change in the workbench editor automatically triggers checkout into the associated change package if the component has not yet been checked out. Saving of backup staging versions also takes place automatically, if the staging versions facility is installed in the repository.

The following rules apply to the **Edit** function in ZMF for Eclipse:

- Only components in package staging libraries may be edited. Baseline and promotion library components may not.
- If a component is locked by another user, it may not be opened for **Edit**.

Invoking the  
Edit Function

Invoke the **Edit** function from the following menu:

- *Serena perspective*— In the **Serena Explorer** navigation view, expand the **ZMF Applications** node, the folder for the desired application, and the **Packages** folder. In the package where the desired component resides, right-click on the desired component to bring up its contextual menu, then select the **Edit** option. The component opens in an editor tab in the upper right pane of the perspective.

See [Chapter 2, "ZMF Operations on Package Components" on page 69](#).

## Building a Component

### Functional Description

The **Build** function transforms a package source component into an executable module in the same change package. Default build procedures are predefined in ChangeMan ZMF, but build parameters may be modified using the **Build** wizard. Multiple components may be built concurrently.

The following rules apply to the **Build** function:

- The package must be in development (DEV) status.
- The component type must be *like-source*.
- The source component and component type must already exist in the package.
- The component must not be a generated component.
- If multiple components are being built, they must all be the same type.
- The component may be locked or unlocked.
- If the **Auto Submit Build** option in settings is checked, and history exists for a module, selecting build will submit the build process without going through all of the various selection dialogs, however if Auto Submit Build is checked, you can choose the "**Build (with dialog)**" context menu option, which then will force the build dialog to be displayed, even though **Auto Submit Build** is set on.

See the *ChangeMan ZMF User's Guide* for more information about the **Build** function.

## Invoking and Viewing Results of the Build Job

Invoking the  
Build Job

In the **Serena** perspective, the component **Build** function is invoked in two ways:

- *From the contextual menu of a package component* — In the **Serena Explorer** navigation view of the **Serena** perspective, expand the **ZMF Applications** node, the desired application, the **Packages** node for that application, the desired change package, and the appropriate like-source staging library containing the component to be built. Right-click on the component to bring up its contextual menu, then select the **Build** option.

See [Chapter 2, "ZMF Operations on Package Components" on page 69](#) for more information.

- *At checkin* — The **Build** function can be invoked by clicking the appropriate checkbox at component **Checkin**. (See "[Checking In a Component](#)" on page 134.)

In the **Java** perspective, if a project is shared with ZMF, the component **Build** function can be invoked from the contextual menus for the following project objects:

- *Project contextual menu* — In the **Package Explorer** navigation view, right-click on the desired project to bring up its contextual menu, select the **Team** submenu, then select **Build**. ZMF for Eclipse will build all like-source components in the project.
- *Folder contextual menu* — In the **Package Explorer** navigation view, expand the desired project and right-click on the desired folder to bring up its contextual menu. Select the **Team** submenu, then select **Build**. ZMF for Eclipse will build all like-source components in the folder.
- *Component contextual menu* — In the **Package Explorer** navigation view, expand the desired project and folder, then select one or more like-source components to build. Right-click on the selection to bring up its contextual menu, select the **Team** submenu, then select **Build**. ZMF for Eclipse will build the selected components.

See "[ZMF Functions in the Package Explorer](#)" on page 101 for more information on ZMF functions in the **Java** perspective.

Viewing Build Job  
Results

The results of the **Build** job can be viewed in the **Serena** perspective of the workbench. In the **Serena Explorer** navigation view, find the ZMF server where the build job was



executed and expand its **z/OS Jobs** node. Under the appropriate job viewing filter, look for the job identified with the job name supplied on your **Build** JOB statement.

See [Chapter 2, "Working with z/OS Jobs" on page 55](#) for more information on viewing the output of your job.

## Build Wizard Windows

The **Build** wizard displays three windows. These windows display default build parameters and allow you to modify them if desired.

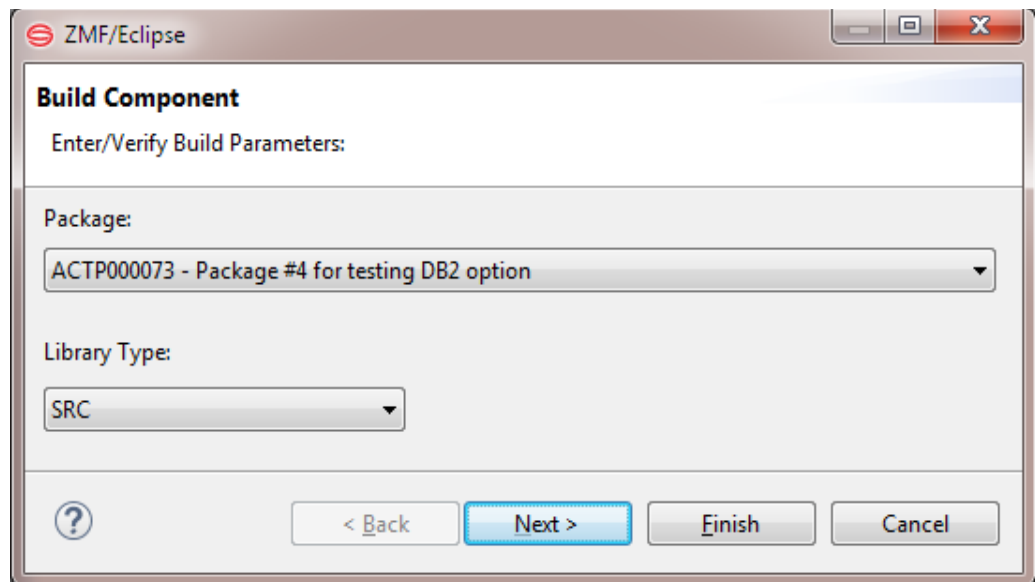
Defaults in the editable wizard fields are populated differently, depending on how many components have been selected for the build job:

- *One component* — Displayed defaults are populated from the component build history, if it exists. Otherwise, the defaults defined for the source language in ZMF are shown.
- *Multiple components* — Displayed defaults are those defined for the source language in ZMF. These defaults are automatically overridden at execution time by the most recently used build parameters in each selected component's build history.



**IMPORTANT!** If you change any displayed defaults, your entries will be used to build all selected components, regardless of predefined build defaults for the source language or actual component build history.

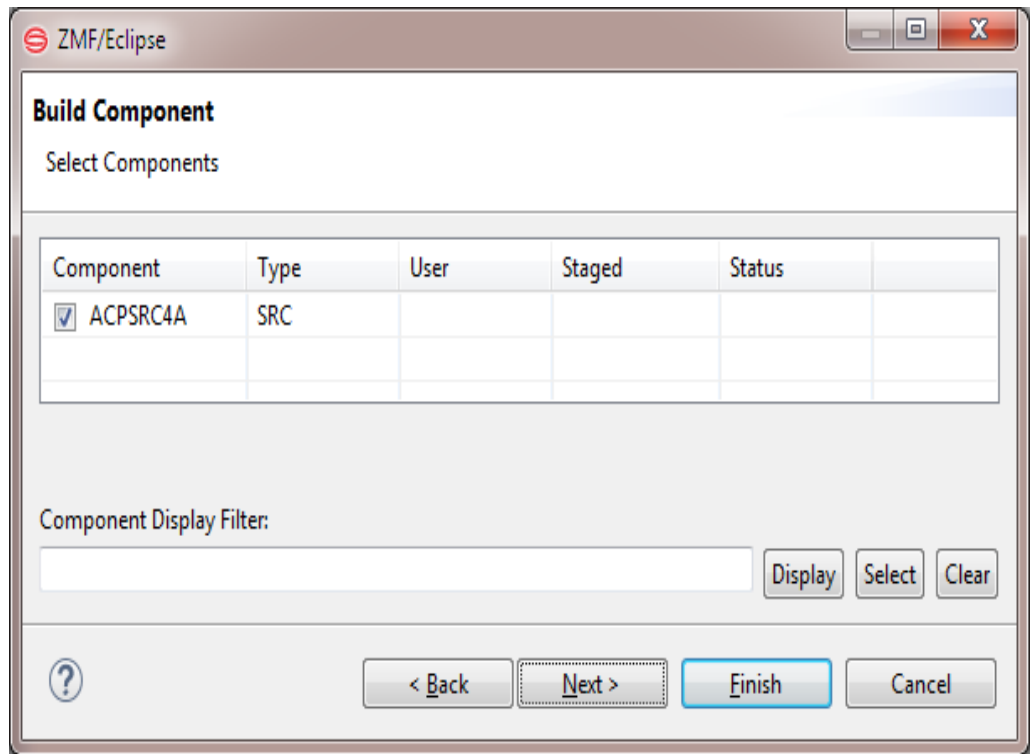
- 1 When you invoke the **Build** function from a component contextual menu, the first of two **Build Component - Enter/Verify Build Parameters** windows displays.



Edit the default entries as needed according to the following table, then click **Next**.

Field	Description
<b>Package</b>	<p>Supply the package ID where the selected like-source component(s) reside after checkin or checkout, and where the executable module(s) resulting from the build will be stored.</p> <p>By default, the package shown is that associated in ZMF for Eclipse with the most recent checkout or checkin of the component(s) listed in the <b>Component(s)</b> field.</p> <p>If the named component(s) should be built in a different package, select the desired package from the drop-down list.</p>
<b>Library Type</b>	Lists the Library Type. Verify that the selection is correct.
<b>Component(s)</b>	Lists the selected component(s) to be built. Verify that the selection is correct.
<b>Job Statement</b>	<p>Displays the JCL JOB statement that will be used to run the <b>Build</b> job on the mainframe. Modify it as necessary.</p> <p>By default, your TSO user ID is appended with an alphabetic character to create the job name. Each time this dialog displays, the last character of the default job name is incremented alphabetically.</p> <p>If you change the displayed values, your changes will be reused as defaults the next time a job card is displayed.</p> <p>The following syntax conventions apply:</p> <ul style="list-style-type: none"> <li>■ A maximum of four lines are allowed.</li> <li>■ Lines start with double slashes and may not exceed 71 bytes in length.</li> <li>■ An asterisk in position three marks a comment.</li> <li>■ A blank in position three continues the JCL command from the previous line. The content of a continuation line must begin in positions 4 to 16. Parameters may not span lines.</li> </ul>

- 2 The **Build** wizard displays the second of two **Build Component - Enter/Verify Build Parameters** windows.



Edit the fields as needed according to the following table. When ready, click **Next**.

<b>Language</b>	Select a source code language from the drop-down list.
<b>Target load library</b>	Select a target from the drop-down list.
<b>Override History</b>	Unless this is checked, you can't adjust the three fields, Language, Target load library and Build procedure.
<b>Build Procedure</b>	Select a build procedure from the drop-down list. Procedures shown are those defined for the selected <i>language</i> in ZMF. <b>NOTE:</b> If a designated build procedure has been defined for a <i>component</i> in ZMF, it will be used for that component even if a different procedure is selected in this field for the component selection as a whole.

<b>DB2 Subsystem</b>	<p>If the component(s) being built access a DB2 relational database, select the appropriate DB2 subsystem from the drop-down list.</p> <p><b>NOTE:</b> A value is required if you select the <b>DB2 Pre-compile</b> option immediately below this field.</p>
<b>DB2 Pre-compile</b>	<p>If the component(s) being built access a DB2 relational database, check this box to include a DB2 pre-compile step in the build.</p> <p><b>NOTE:</b> If this box is checked, a value is also required in the <b>DB2 Subsystem</b> field.</p>
<b>Compile Parm</b> s	<p>This 34-byte field displays a comma-separated list of compile parameters. These parameter values are used only if a value is not already set elsewhere by the following:</p> <ul style="list-style-type: none"> <li>■ System defaults for the compiler</li> <li>■ Hard-coded options in designated compile procedures</li> <li>■ User-defined compile options set by file tailoring</li> </ul> <p>For more information about compile parameters, see the <i>ChangeMan ZMF User's Guide</i>.</p>
<b>Link Parm</b> s	<p>This 34-byte field displays a comma-separated list of link parameters. These parameter values are used only if a value is not already set elsewhere by the following:</p> <ul style="list-style-type: none"> <li>■ System defaults for the compiler</li> <li>■ Hard-coded options in designated compile procedures</li> <li>■ User-defined link or binder options set by file tailoring</li> </ul> <p>For more information about link parameters, see the <i>ChangeMan ZMF User's Guide</i>.</p>
<b>Job Statement</b>	<p>Displays the JCL JOB statement that will be used to run the <b>Recompile</b> job on the mainframe. Modify it as necessary.</p> <p>By default, your TSO user ID is appended with an alphabetic character to create the job name. The last character of the default job name is incremented alphabetically each time a job card displays. If you change the displayed values, your changes are used as defaults.</p> <p>The following syntax conventions apply:</p> <ul style="list-style-type: none"> <li>■ A maximum of four lines are allowed.</li> <li>■ Lines must begin with double slashes and may not exceed 71 characters in length.</li> <li>■ An asterisk in position three marks a comment.</li> <li>■ A blank in position three continues the JCL command from the previous line. The content of a continuation line must begin in positions 4 to 16. Parameters may not be broken across continuation lines.</li> </ul>

- 3** The final window displayed by the **Build** wizard requests values for user-defined build parameters defined by your ZMF administrator. These options are used in JCL file tailoring for the build process.

The display defaults are those in effect for the language used by the selected source components. The labels displayed for each user-defined option are configured in the BUILD member of the ZDDOPTS configuration library.

Only options with a status of Editable="Y" can be changed in this window.

The example screen below shows some common user-defined options, but those used at your site will vary.

Variable Name	Variable Value	Variable Help
Compile only	N	
IMS DLITxxx entry	N	
CICS precompile	N	
Drop INCLUDE stmts	N	
Easytrieve object	N	
User Option 06	N	
User Option 07	N	
User Option 08	N	
User Option 09	N	
SN User Opt 10	N	
SN User Opt 11	N	
Fetch(F),Subload(S),Load( ), Zunit(U)		
User Option 13	N	
User Option 14	N	
User Option 15	N	
User Option 16	N	
User Option 17	N	
User Option 18	N	
User Option 19	N	
User Option 20	N	
Long user option 1-01	Y	
Long user option 44-02	UserOp 44-02	
Long user option 72-05	UserOp 72-05	

Select variable in table to update value; Shift+Up / Shift+Down to navigate

Name: CICS precompile

Value: N

? < Back Next > Finish Cancel

Make any desired changes, then click **Finish** to submit the **Build** job for execution.

## Recompiling a Component

### Functional Description

The **Recompile** function lets you perform build processing without checking out like-source components into your change package. This process performs the minimum

processing necessary to build like-load components from source code, but reduces the risk that you will inadvertently change source code. The Recompile is invoked from a like-source member in a baseline or promotion library. The resulting like-load member is checked in (or *staged*) to a package staging library.

Use the **Recompile** function when you want to include new versions of like-copy components in a build process and the like-source component is not changing.

The following rules apply to the **Recompile** function:

- The recompiled component must be of a *like-source* library type.
- Only baseline and promotion components may be recompiled into a package.
- You cannot recompile generated components.
- Only one component may be recompiled at a time.

See the *ChangeMan ZMF User's Guide* for more information about the **Recompile** function.

## Invoking and Viewing Results of the Recompile Job

### Invoking the Recompile Job

Invoke the **Recompile** function from the contextual menus for the following object types:

- *Baseline components* — In the **Serena Explorer** navigation view of the **Serena** perspective, expand the **ZMF Applications** node, the folder for the desired application, the **Baseline** node, and the node for the like-source library where the desired source component resides. Right-click on the component to bring up its contextual menu, then select the **Recompile** option.

See [Chapter 2, "ZMF Operations on Baseline Library Components" on page 67](#) for more information.

- *Promotion components* — In the **Serena Explorer** navigation view of the **Serena** perspective, expand the **ZMF Applications** node, the folder for the desired application, and the **Promotion** node. Find the promotion site where the desired source component resides, expand it, and expand the appropriate like-source promotion library within it to show a list of components. Right-click on the desired component to bring up its contextual menu, then select the **Recompile** option.

See [Chapter 2, "ZMF Operations on Promoted Components" on page 73](#) for more information.

### Viewing Recompile Job Results

The results of the **Recompile** job can be viewed in the **Serena** perspective of the workbench. In the **Serena Explorer** navigation view, find the ZMF server where the job was executed and expand its **z/OS Jobs** node. Under the appropriate job viewing filter, look for the job identified with the job name supplied on your **Recompile** JOB statement.

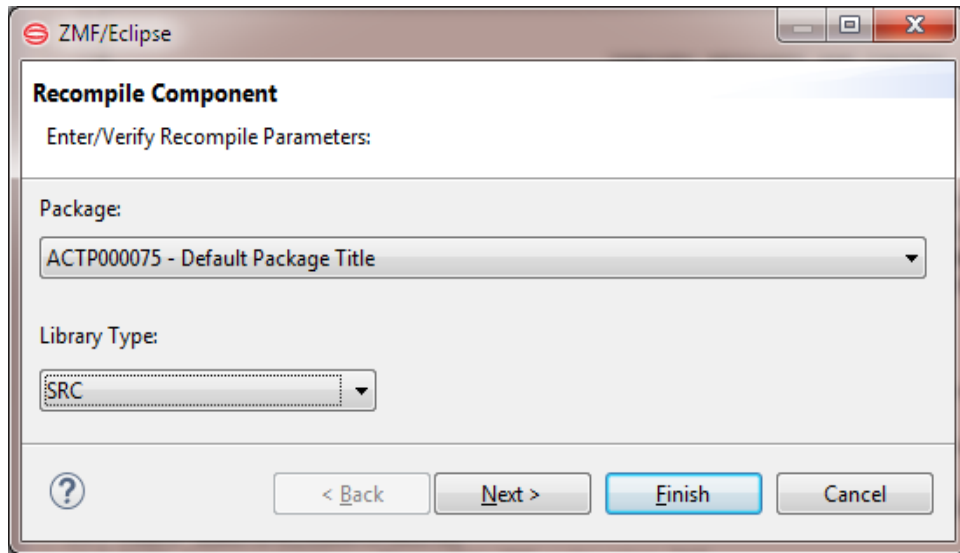
See [Chapter 2, "Working with z/OS Jobs" on page 55](#) for more information on viewing the output of your job.

## Recompile Wizard Screens

The **Recompile** wizard displays three windows. These windows display default build parameters and allow you to modify them if desired.

Displayed defaults are populated from the component build history, if it exists. Otherwise, the defaults defined for the source language in ZMF are shown.

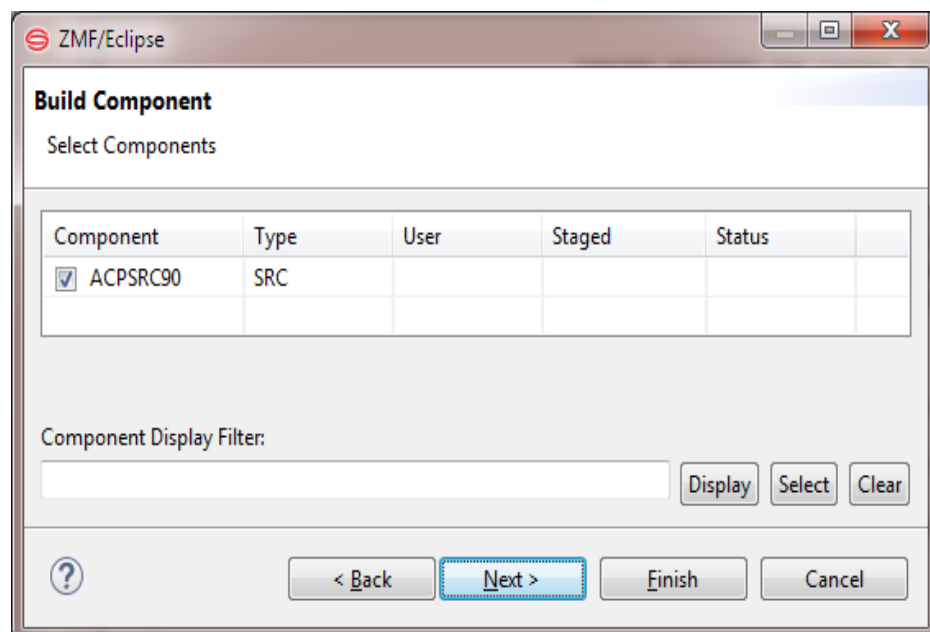
- 1 When you invoke the **Recompile** function from a component contextual menu, the first window, **Recompile Component - Enter/Verify Recompile Parameters** window displays.



Edit the default entries as needed after the following table. When ready, click **Next**.

Field	Description
<b>Package</b>	Select the target package to receive the like-load component created by the recompile job. The pull-down menu displays packages with an install date equal to or later than today's date.
<b>Library Type</b>	Displays the library type. Verify that the entry is correct.

- 2 The **Recompile** wizard displays the second window - **Build Component**.





### 3 The **Recompile** wizard displays the third window - **Recompile Component**.

**Recompile Component**  
Enter/Verify Recompil Parameters:

Language: COBOL2 Target load library:  Override History

Build procedure: CMNCOB2 DB2 subsystem:  DB2 pre-compile

Compile Params:

Link Params:

Job Statement:

```
//JPRESTOR JOB (0),ZMFECLIPSE,MSGCLASS=X,CLASS=A,NOTIFY=JPRESTO
//*
//*
//*
```

< Back Next > Finish Cancel

Edit the default entries as needed using the following table. When ready, click **Next**.

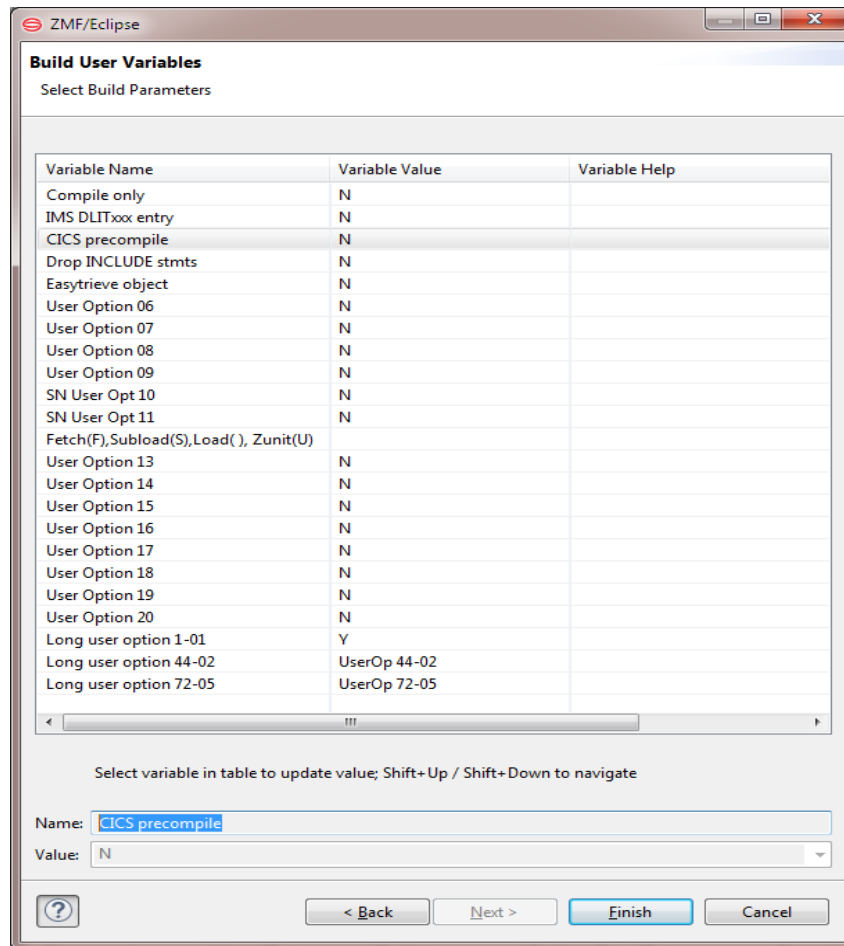
Field	Description
<b>Language</b>	Select a source language from the drop-down list.
<b>Procedure</b>	Select a designated build procedure from the drop-down list. <b>NOTE:</b> If a designated build procedure has been defined for a component in ChangeMan ZMF, it will be used even if you type a different procedure name in this field.
<b>DB2 Subsystem</b>	If the component being recompiled accesses a DB2 relational database, select the appropriate DB2 subsystem from the drop-down list. <b>NOTE:</b> A value is required if you select the <b>DB2 Pre-compile</b> option immediately after this field.
<b>DB2 Pre-compile</b>	If the component being recompiled accesses a DB2 relational database, check this box to include a DB2 pre-compile step. <b>NOTE:</b> If this box is checked, a value is also required in the <b>DB2 Subsystem</b> field.

Field	Description
<b>Compile Parm</b> s	<p>This 34-byte field displays a comma-separated list of link parameters. These parameter values are used only if a value is not already set elsewhere by the following:</p> <ul style="list-style-type: none"> <li>■ System defaults for the compiler</li> <li>■ Hard-coded options in designated compile procedures</li> <li>■ User-defined compile options set by file tailoring</li> </ul> <p>For more information about link parameters, see the <i>ChangeMan ZMF User's Guide</i>.</p>
<b>Link Parm</b> s	<p>This 34-byte field displays a comma-separated list of link parameters. These parameter values are used only if a value is not already set elsewhere by the following:</p> <ul style="list-style-type: none"> <li>■ System defaults for the compiler</li> <li>■ Hard-coded options in designated compile procedures</li> <li>■ User-defined link or binder options set by file tailoring</li> </ul> <p>For more information about link parameters, see the <i>ChangeMan ZMF User's Guide</i>.</p>
<b>Job Statement</b>	<p>Displays the JCL JOB statement that will be used to run the <b>Recompile</b> job on the mainframe. Modify it as necessary.</p> <p>By default, your TSO user ID is appended with an alphabetic character to create the job name. The last character of the default job name is incremented alphabetically each time a job card displays. If you change the displayed values, your changes are used as defaults.</p> <p>The following syntax conventions apply:</p> <ul style="list-style-type: none"> <li>■ A maximum of four lines are allowed.</li> <li>■ Lines must begin with double slashes and may not exceed 71 characters in length.</li> <li>■ An asterisk in position three marks a comment.</li> <li>■ A blank in position three continues the JCL command from the previous line. The content of a continuation line must begin in positions 4 to 16. Parameters may not be broken across continuation lines.</li> </ul>

- 4 The final window displayed by the **Recompile** wizard is the **Build User Variables** window, which requests values for user-defined build parameters defined by your ZMF administrator. These options are used in JCL file tailoring for the recompile process.

The display defaults are those in effect for the language used by the selected source component. The labels displayed for each user-defined option are configured in the BUILD member of the ZDDOPTS configuration library.

Only options with a status of editable="Y" can be changed in this window.



Edit as needed, then click **Finish** to submit the **Recompile** job for execution.

## Relinking a Component

### Functional Description

The **Relink** function binds a *like-load* software component with the reusable objects it references to create an executable module.

The following rules apply to the **Relink** function:

- The relinked component must be of a *like-load* library type.
- Baseline and promotion components may be relinked without any need for a compile or full build.
- Package components may be relinked if they were relinked in baseline or promotion previously and do not have a corresponding source component in the package.
- You cannot relink generated components.
- Only one component can be relinked at a time.

See the *ChangeMan ZMF User's Guide* for more information about the **Relink** function.

## Invoking and Viewing Results of the Relink Function

Invoking the Relink Job      The **Relink** function is invoked (not shown if not applicable) from the contextual menus for the following object types:

- *Baseline components* — In the **Serena Explorer** navigation view of the **Serena** perspective, expand the **ZMF Applications** node, the node for the desired application, the **Baseline** node, and the node for the like-load library where the desired component resides. Right-click on the desired component to bring up its contextual menu, then select the **Relink** option.

See [Chapter 2, "ZMF Operations on Baseline Library Components" on page 67](#) for more information.

- *Package components* — In the **Serena Explorer** navigation view of the **Serena** perspective, expand the **ZMF Applications** node, the node for the desired application, and the **Packages** node. Find the package where the desired component resides, expand it, and expand the appropriate like-load staging library. Right-click on the desired component to bring up its contextual menu, then select the **Relink** option.

See [Chapter 2, "ZMF Operations on Package Components" on page 69](#) for more information.

- *Promotion components* — In the **Serena Explorer** navigation view of the **Serena** perspective, expand the **ZMF Applications** node, the node for the desired application, and the **Promotion** node. Find the promotion site where the desired component resides, expand it, and expand the appropriate like-load promotion library. Right-click on the desired component to bring up its contextual menu, then select the **Relink** option.

See [Chapter 2, "ZMF Operations on Promoted Components" on page 73](#) for more information.

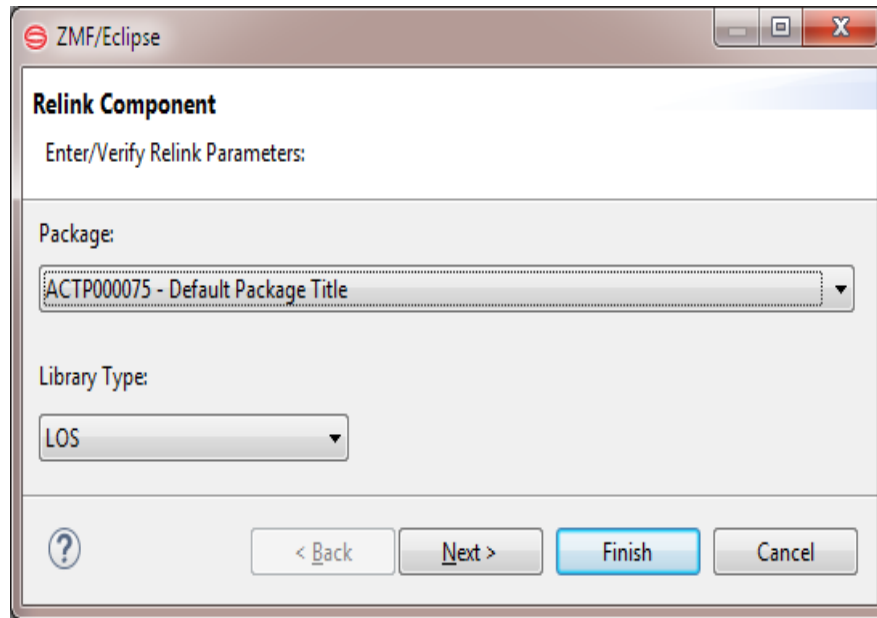
Viewing Relink Job Results      The results of the **Relink** job can be viewed in the **Serena** perspective of the workbench. In the **Serena Explorer** navigation view, find the ZMF server where the job was executed and expand its **z/OS Jobs** node. Under the appropriate job viewing filter, look for the job identified with the job name supplied on your **Relink** JOB statement.

See [Chapter 2, "Working with z/OS Jobs" on page 55](#) for more information on viewing the output of your job.

## Relink Wizard Screens

The **Relink** wizard displays three windows. These windows display default build parameters and allow you to modify them if desired. Displayed defaults are populated from the component build history, if it exists. Otherwise, the defaults defined for the source language in ChangeMan ZMF are shown.

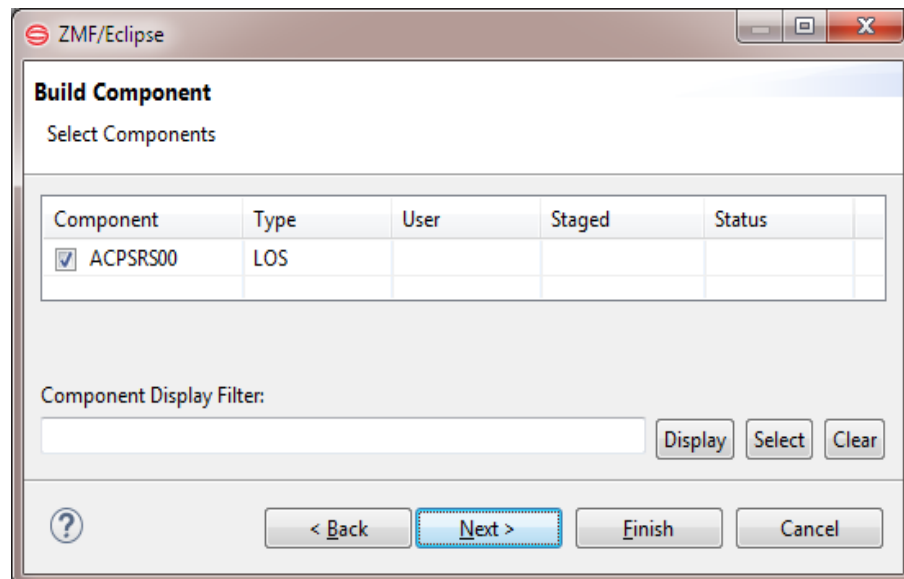
- 1 When you invoke the **Relink** function from a component contextual menu, the first of a series of windows - the **Relink Component - Enter/Verify Relink Parameters** window displays.



Edit the default entries as needed according to the following table, then click **Next**.

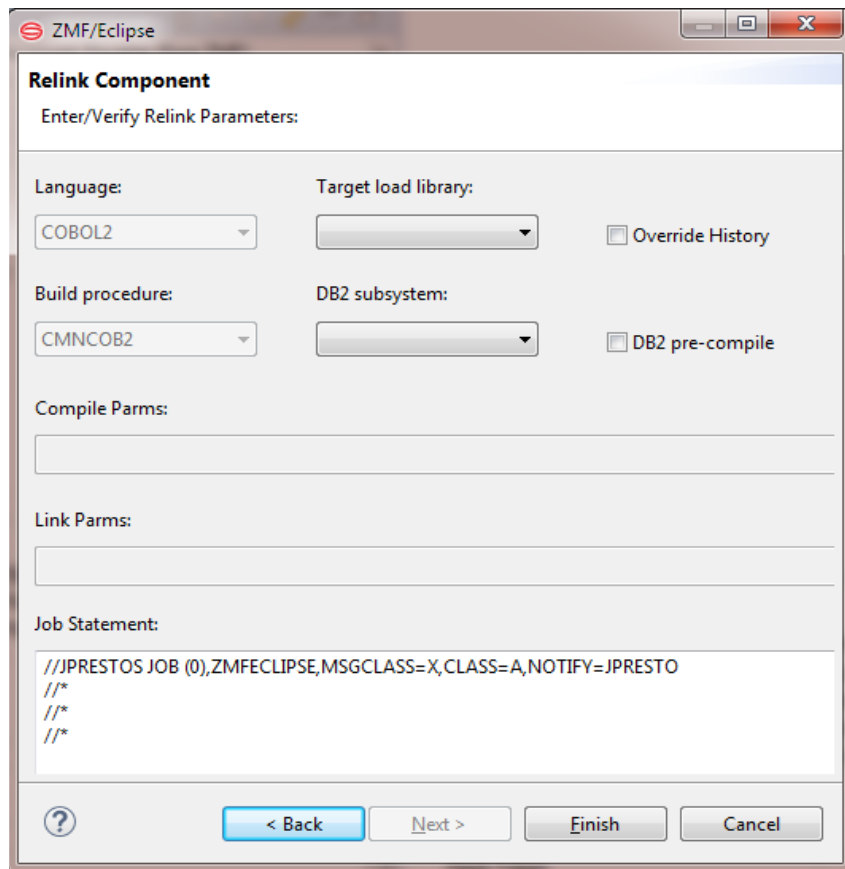
Field	Description
<b>Package</b>	Displays the package containing the selected like-load component to be relinked. If a different package is desired, select it from the drop-down list.
<b>Library Type</b>	Displays the Library type. Verify that is as desired before continuing

2 The **Relink** wizard displays the second window - **Build Component**.



<b>Component(s)</b>	Displays the component(s) to be relinked, verify the component(s) you want to relink are checked.
<b>Component Display Filter" (s)</b>	Displays the component(s) to be relinked, verify the component(s) you want to relink are checked.

**3** The **Relink** wizard displays the third window - **Relink Component**.



Edit the fields as needed according to the following table.

Field	Description
<b>Language</b>	Select a source language from the drop-down list.
<b>Target Load Library</b>	Select a target load library from the drop-down list.
<b>Override history</b>	Check to override the component history.
<b>Build Procedure</b>	Select a designated build procedure from the drop-down list. <b>NOTE:</b> If a designated build procedure has been defined for a component in ChangeMan ZMF, it will override this field.

Field	Description
<b>DB2 Subsystem</b>	If the component being relinked accesses a DB2 relational database, select the appropriate DB2 subsystem from the drop-down list.  <b>NOTE:</b> A value is required if you select the <b>DB2 Pre-compile</b> option immediately below this field.
<b>DB2 Pre-compile</b>	If the component being relinked accesses a DB2 relational database, check this box to include a DB2 pre-compile step.  <b>NOTE:</b> If this box is checked, a value is also required in the <b>DB2 Subsystem</b> field.
<b>Compile Parm</b> s	This field is not editable, as source code is not being compiled.
<b>Link Parm</b> s	This 34-byte field displays a comma-separated list of link parameters. These parameter values are used only if a value is not already set elsewhere by the following: <ul style="list-style-type: none"> <li>■ System defaults for the compiler</li> <li>■ Hard-coded options in designated compile procedures</li> <li>■ User-defined link or binder options set by file tailoring</li> </ul> For more information, see the <i>ChangeMan ZMF User's Guide</i> .
<b>Job Statement</b>	Displays the JCL JOB statement that will be used to run the <b>Relink</b> job on the mainframe. Modify it as necessary.  By default, your TSO user ID is appended with an alphabetic character to create the job name. The last character of the default job name is incremented alphabetically each time a job card displays. If you change the displayed values, your changes are used as defaults.  The following conventions apply: <ul style="list-style-type: none"> <li>■ A maximum of four lines are allowed.</li> <li>■ Lines must begin with double slashes and may not exceed 71 characters in length.</li> <li>■ An asterisk in position three marks a comment.</li> <li>■ A blank in position three continues the JCL command from the previous line. The content of a continuation line must begin in positions 4 to 16. Parameters may not be broken across continuation lines.</li> </ul>

Make any desired changes, then click **Finish** to submit the **Relink** job for execution.

## Viewing the Component Staging Versions List

### Staging Versions in ChangeMan ZMF

A component in a package staging library may optionally be versioned — that is, a snapshot copy of it may be saved in a special backup library — at the occurrence of certain events in its life cycle. This feature is known as the *staging versions facility* of

ChangeMan ZMF. The staging versions facility is optional, but if it is installed on the ZMF server, it is accessible from the workbench using ZMF for Eclipse.

A package component is versioned whenever you check it out from baseline into a package, or check it in (stage it) from a personal development library to a package, or save edits to it, or delete it from a package. Depending on how your ZMF administrator configures it, versioning may be mandatory and automatic or optional and prompted. Moreover, versioning behavior may vary by application and library type. For example, you might be prompted to provide a change description every time you save changes to COBOL source code in the Eclipse editor, but not when you save copybooks.

For more information about staging versions, refer to the *ChangeMan ZMF User's Guide*.

## Staging Versions in ZMF for Eclipse

The versioning of staged components takes place in the background while you work in ZMF for Eclipse. No explicit user command is required to save a staging version.

The **Staging Versions** function in ZMF for Eclipse displays the **Staging Versions** table view in the current workbench perspective. From this table view, you can browse previous versions of a package component, or compare different versions of the component over its change history.

## Displaying the Staging Versions View

Invoking the  
Staging Versions  
Table View

The **Staging Versions** table view can be requested from component contextual menus in the following perspectives:

- **Serena perspective** — In the **Serena Explorer** navigation view, find the desired server and expand its **ZMF Applications** node, the node for the application where the versioned component resides, and the **Packages** node for that application. Navigate to the package and expand the library node containing the component. Right-click on the component to bring up its contextual menu and select **Staging Versions**.

See [Chapter 2, "ZMF Operations on Package Components" on page 69](#) for more information.

- **Java perspective** — In the **Package Explorer** navigation view, expand the node of a desired project that is shared with ChangeMan ZMF. Expand any subordinate folders to navigate to the versioned component, then right-click on that component to bring up its contextual menu. Select the **Team** submenu, then select **Staging Versions**.

See [Chapter 3, "Working with the Java Perspective" on page 97](#) for more information.

The **Staging Versions** table view displays in a tab in the lower right pane of the perspective where it was invoked. For example:



Properties Staging Versions Query Component Progress

Staging Versions - ACPCPY2X.CPY - Package(ACTP000075)

Location	Updater	Date	Change Description
<input type="checkbox"/> Staging	JPRESTO	2016/08/04 21:45:17	Checkin as required
<input type="checkbox"/> Baseline	JPRESTO	2014/12/12 01:03:44	Baseline version
<input type="checkbox"/> Backup	JPRESTO	2016/08/04 21:45:17	
<input type="checkbox"/> Backup	JPRESTO	2016/08/04 21:31:16	
<input type="checkbox"/> Backup	JPRESTO	2016/08/04 21:30:48	

The **Location** column identifies each component version as one of the following:

- *Staging* — The most recent version of the component in the package staging library.
- *Backup* — A versioned snapshot of the component during its history in the package staging library. The oldest Backup version shows the component as it was checked out from baseline into the package or checked in to the package from outside ZMF.
- *Baseline* — The most recent version of the component in baseline.



**NOTE** The most *recent* baseline version of a component may not be the same as the baseline version you checked out. Another user may have baselined a package containing a different version of the component after your checkout took place.

The **Updated** column in the **Staging Versions** table view shows the timestamp associated with each component version. The **Change Description** column records any notes on the change history of the component. Some change descriptions are logged automatically. Others may be entered manually when you are prompted from them during a save operation in the Eclipse editor.

For example, the oldest Backup version in the **Staging Versions** table view above is identified in the **Change Description** field as a new component checked into the package from an Eclipse workbench library. In contrast, a component checked out from baseline into a package will have a change description reporting the checkout and baseline level.

## Contextual Menu for Staging Versions

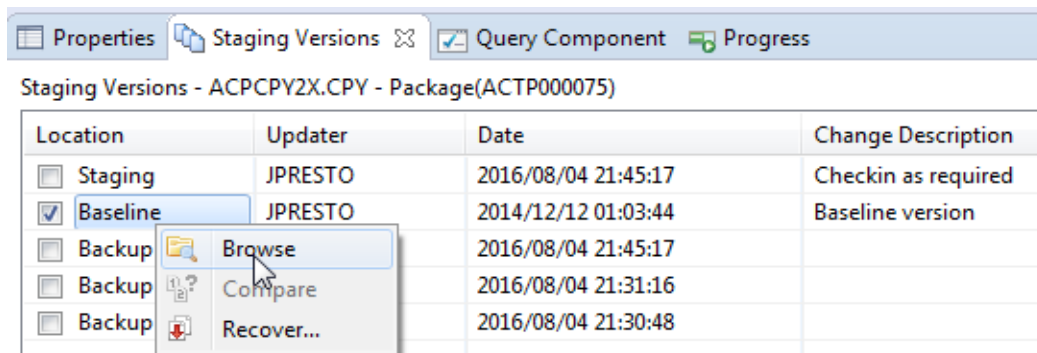
A contextual menu can be invoked by right-clicking anywhere in the **Staging Versions** table view. The actions enabled in the contextual view depend on what component checkboxes, and how many of them, you select in the view.

- [Browsing Component Staging Versions](#) — Enabled if **one** version is selected.
- [Comparing Component Staging Versions](#) — Enabled if **multiple** versions are selected.
- Recover Component Staging Versions - Enabled if a checked component is listed as being in **Backup**.

## Browsing Component Staging Versions

From the **Staging Versions** table view, you can browse the current staging version of a package component, a backup staging version of that component, or the most recent baseline version of that component. To do this, perform the following steps.

- 1 Click the checkbox beside the component version to browse and uncheck all others.
- 2 Right-click anywhere in the table to bring up the **Staging Versions** contextual menu and select the **Browse** option. For example:



- 3 The content of the checked component displays in a read-only editor.



**NOTE** Additional options for browsing components are described in the topic "[Browsing a Component](#)" on page 141.

## Comparing Component Staging Versions

**Functional Description** From the **Staging Versions** table view, you can compare any version of a package component with the current staging version, a backup staging version, or the most recent baseline version of the same component.



**NOTE** The **Compare** function on the **Staging Versions** contextual menu does not perform comparisons against older baseline versions or against promotion libraries. See "[Comparing Components to Baseline or Promotion](#)" on page 163 if you wish to compare package components against baseline or promotion libraries.

**Comparison Step-by-Step** To compare staging versions, perform the following steps:

- 1 Click the checkboxes beside components you want to compare and uncheck all others.
- 2 Right-click anywhere in the table to bring up the **Staging Versions** contextual menu, then select the **Compare** option. For example:

Location	Updater	Date	Change Description
<input checked="" type="checkbox"/> Staging	JPRESTO	2016/08/04 21:45:17	Checkin as required
<input checked="" type="checkbox"/> Baseline		2014/12/12 01:03:44	Baseline version
<input type="checkbox"/> Backup		2016/08/04 21:45:17	
<input type="checkbox"/> Backup		2016/08/04 21:31:16	
<input type="checkbox"/> Backup		2016/08/04 21:30:48	

- 3 The comparison editor displays the selected components side-by-side under the **Compare** tab in the upper right pane of the perspective. Insertions, deletions, and inline changes are highlighted for ease of comparison.

For example, two components might be compared as follows:

ACPCPY2X-(STAGING)		ACPCPY2X-(BASELINE)	
1	* ACPCPY2X.CPY	1	* ACPCPY2X.CPY
2	* PACKAGE ACTP000075 second edit	2	* PACKAGE ACTP000016 S4.V710T19
3	* PACKAGE ACTP000016 S4.V710T19	3	* PACKAGE ACTP000007 S4.V711
4	* PACKAGE ACTP000007 S4.V711	4	01 ACPCPY2X PIC X(01).
5	01 ACPCPY2X PIC X(01).		

The status bar at the bottom of the perspective provides descriptive information about the most recent comparison. For example:

Left: 3 : 1, Right: 2 : 1, incoming deletion #1 (Left: 2 : 2, Right: before line 2)

For additional information on working with the comparison editor, see ["Comparing Components to Baseline or Promotion"](#) on page 163.

## Comparing Components to Baseline or Promotion

### Functional Description

The **Compare to Baseline/Promotion** function lets you compare the current staging version of a package component against other versions in baseline or promotion libraries. Differences are displayed side-by-side under the **Compare** tab in the upper right pane of the perspective. Insertions, deletions, and inline changes are highlighted for ease of comparison.



**NOTE** The **Compare to Baseline/Promotion** function does not perform comparisons involving component staging versions. See ["Comparing Component Staging Versions"](#) on page 162 if you wish to compare component staging versions.

## Invoking the Compare Function

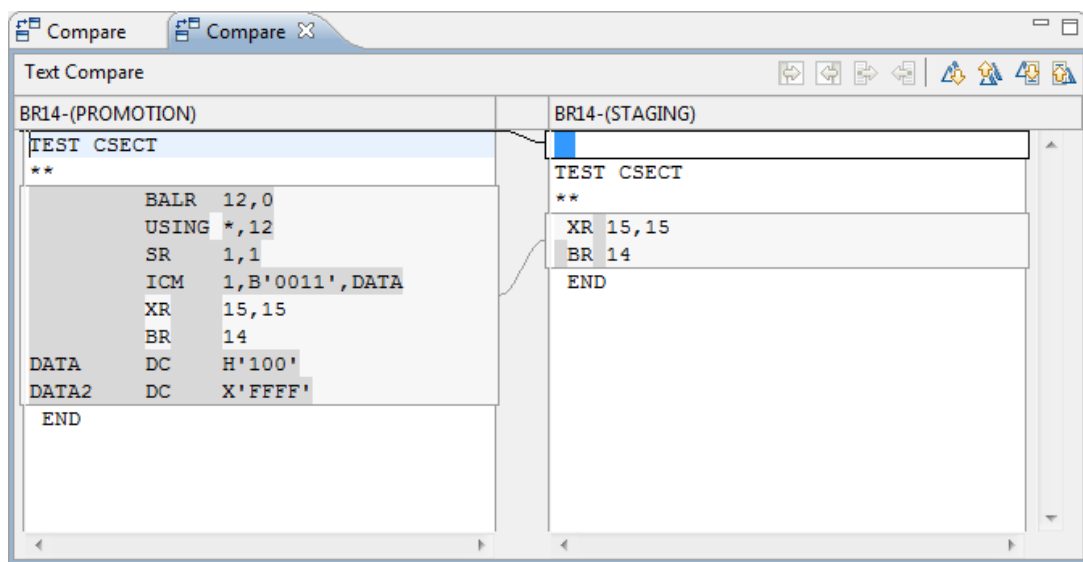
The **Compare to Baseline/Promotion** function is invoked from the package component contextual menu in the **Serena Explorer** navigation view of the **Serena** perspective.

See [Chapter 2, "ZMF Operations on Package Components"](#) on page 69 for more information on invoking this function.

## Viewing Comparison Results

The comparison editor displays the selected components side-by-side under the **Compare** tab in the upper right pane of the perspective. Insertions, deletions, and inline changes are highlighted for ease of comparison.

For example:



## Source-to-Load Relationships

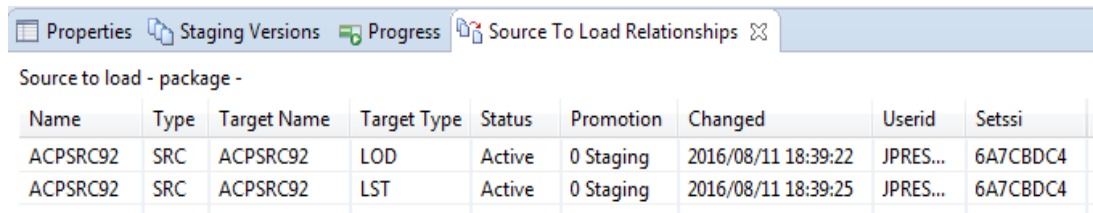
The **Source-to-Load Relationships** function lists the dependencies between source modules and load modules in the current change package.

Invoke the **Source-to-Load Relationships** function from the following menu:

- **Serena perspective** — In the **Serena Explorer** navigation view, expand the node for the ZMF server that hosts the desired repository, then the **z/OS Applications** node, then the node for the desired application, then the **Packages** node. Navigate to the particular package where the component of interest resides, then expand the package and staging library nodes and select the desired like-source component. From the component's contextual menu, select the **Source-to-Load Relationships** option.

See [Chapter 2, "ZMF Operations on Package Components" on page 69](#) for more information.

The resulting report displays in a table view under the **Source to Load Relationships** tab in the lower right pane of the perspective.



The screenshot shows a software interface with a tab labeled "Source To Load Relationships". Below the tab is a table with the following data:

Name	Type	Target Name	Target Type	Status	Promotion	Changed	Userid	Setssi
ACPSRC92	SRC	ACPSRC92	LOD	Active	0 Staging	2016/08/11 18:39:22	JPRES...	6A7CBDC4
ACPSRC92	SRC	ACPSRC92	LST	Active	0 Staging	2016/08/11 18:39:25	JPRES...	6A7CBDC4

## Component Bill of Materials

### Functional Description

The **Component Bill of Materials** function performs a top-down query to find all subordinate components on which the selected, higher-level component has dependencies. For example, a higher-level program might call multiple subroutines, and you might want a list of all the subroutines involved before you change the program.

A bill-of-materials query can be performed for superordinate components that are like-source, like-load, like-copybook, or a member of a JCL procedure library. Subordinate components can be copybooks, subroutines, JCL procedures, or name-to-symbol mappings for data sets and programs referenced by the superordinate component.

The dependency relationship of interest is selectable at the time the bill of materials list is requested. Only one dependency type may be queried at a time.

### Invoking and Viewing the Bill of Materials

Invoking the  
Bill of Materials

The **Component Bill of Materials** list is invoked from the following menus in the **Serena Explorer** navigation view of the **Serena** perspective:

- *Baseline component contextual menu* — Expand the **z/OS Applications** node for the relevant ZMF server. Navigate to the desired application node and expand it, then expand the **Baseline** node and the node for the relevant baseline library, then right-click on the component whose build of materials you wish to view. When the contextual menu displays, select **Component Bill of Materials**.

See [Chapter 2, "ZMF Operations on Baseline Library Components" on page 67](#) for more information.

- *Package component contextual menu* — Expand the **z/OS Applications** node for the relevant ZMF server. Navigate to the desired application node and expand it, then expand the **Packages** node, then the node for the relevant change package and staging library, then right-click on the component whose build of materials you wish to view. When the contextual menu displays, select **Component Bill of Materials**.

See [Chapter 2, "ZMF Operations on Package Components" on page 69](#) for more information.

- *Promotion component contextual menu* — Expand the **z/OS Applications** node for the relevant ZMF server. Navigate to the desired application node and expand it, then expand the **Promotion** node, then the node for the relevant promotion site, promotion level, and library type, then right-click on the component whose build of materials you wish to view. When the contextual menu displays, select **Component Bill of Materials**.

See [Chapter 2, "ZMF Operations on Promoted Components"](#) on page 73 for more information.

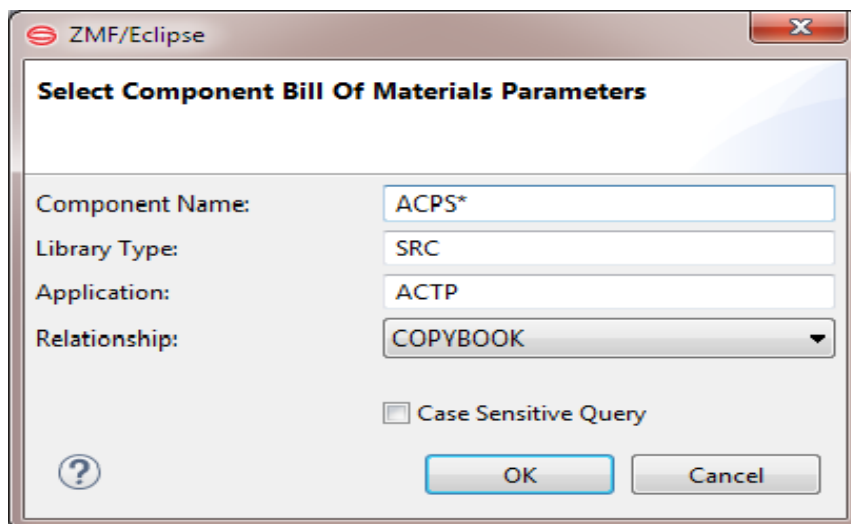
Viewing the Bill of Materials

The requested build of materials list displays in a ZMF table view under the **Component Bill of Materials** tab in the lower right pane of the perspective.

## Bill of Materials Wizard Step-by-Step

To request a component bill of materials list, perform the following steps.

- 1 Select the **Component Bill of Materials** option from the superordinate component's contextual menu.
- 2 The **Select Component Bill of Materials Parameters** window displays the selected component for verification and prompts you for a dependency relationship.



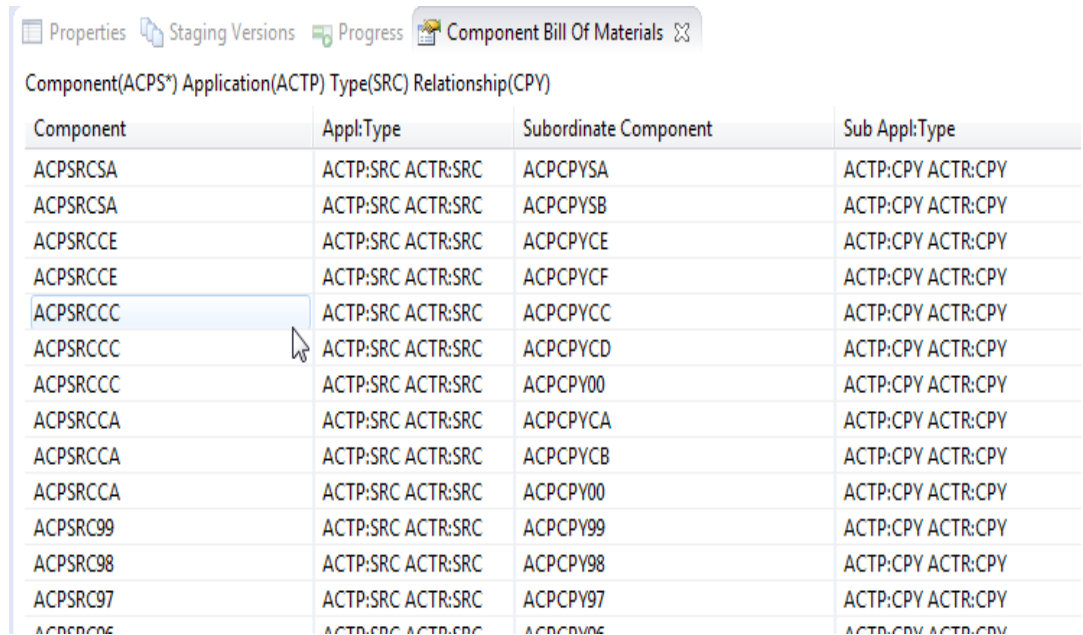
- a Verify that the **Component Name**, **Library Type**, and **Application** values shown are correct. If desired, edit the values in these text fields.



**NOTE** Wild cards may be used to request multiple superordinate components, multiple library types, or cross-application queries.

- b In the **Relationship** field, select from the pull-down list the dependency relationship for which a component bill of materials list is desired. Options:
  - Copybook — Requests all copybooks referenced by the higher-level component. Returns SRC, like-source, CPY, and like-copybook objects only; other library types are ignored.
  - DSN Name/Symbol — Requests all data set name-to-symbol mappings used by the higher-level component (that is, any subordinate JCL components that include a DSN= or DSNAME= parameter in

- a DD statement). Returns JCL, like-JCL, PRC, and like-procedure objects only.
- **JCL Procedure** — Requests all JCL procedures invoked by the higher-level component (that is, any subordinate JCL components that include an EXEC statement). Returns JCL, like-JCL, PRC, and like-procedure objects only.
  - **PGM Name/Symbol** — Requests all program name-to-symbol mappings used by the higher-level component (that is, any subordinate JCL components that contain a PGM= parameter in an EXEC statement). Returns JCL, like-JCL, PRC, and like-procedure objects only.
  - **Subroutine** — Requests all subroutines called by the higher-level component (that is, all subordinate load modules that reference a statically link-edited subprogram). Returns LOD and like-load objects only.
- c Click the **Case Sensitive Query** checkbox if the search for component dependencies should respect case in component, library, and application names. This option is frequently desirable with z/OS USS HFS components.
- d Click **OK** to submit the query.
- 3 A list of subordinate components matching your search criteria are returned under the **Component Bill of Materials** tab in the bottom right pane of the perspective. The search parameters appear in the information bar below the tab name. For example:



Component	Appl:Type	Subordinate Component	Sub Appl:Type
ACPSRCSA	ACTP:SRC ACTR:SRC	ACPCPYSA	ACTP:CPY ACTR:CPY
ACPSRCSA	ACTP:SRC ACTR:SRC	ACPCPYSB	ACTP:CPY ACTR:CPY
ACPSRCCE	ACTP:SRC ACTR:SRC	ACPCPYCE	ACTP:CPY ACTR:CPY
ACPSRCCE	ACTP:SRC ACTR:SRC	ACPCPYCF	ACTP:CPY ACTR:CPY
ACPSRCCC	ACTP:SRC ACTR:SRC	ACPCPYCC	ACTP:CPY ACTR:CPY
ACPSRCCC	ACTP:SRC ACTR:SRC	ACPCPYCD	ACTP:CPY ACTR:CPY
ACPSRCCC	ACTP:SRC ACTR:SRC	ACPCPY00	ACTP:CPY ACTR:CPY
ACPSRCCA	ACTP:SRC ACTR:SRC	ACPCPYCA	ACTP:CPY ACTR:CPY
ACPSRCCA	ACTP:SRC ACTR:SRC	ACPCPYCB	ACTP:CPY ACTR:CPY
ACPSRCCA	ACTP:SRC ACTR:SRC	ACPCPY00	ACTP:CPY ACTR:CPY
ACPSRC99	ACTP:SRC ACTR:SRC	ACPCPY99	ACTP:CPY ACTR:CPY
ACPSRC98	ACTP:SRC ACTR:SRC	ACPCPY98	ACTP:CPY ACTR:CPY
ACPSRC97	ACTP:SRC ACTR:SRC	ACPCPY97	ACTP:CPY ACTR:CPY
ACPSRC96	ACTP:SRC ACTR:SRC	ACPCPY96	ACTP:CPY ACTR:CPY

# Component Impact Analysis

## Functional Description

The **Impact Analysis** function performs a bottom-up query to find all higher-level components that reference or invoke a selected, subordinate component. For example, a reusable subroutine might be called by multiple higher-level programs, and you might want to discover which programs those are before making any changes to the subroutine.

The subordinate component selected for impact analysis may be might be a copybook, subroutine, JCL procedure, or physical object name for a data set, program, or JCL procedure referenced in a name-to-symbol mapping. The higher-level components returned by an impact analysis query may be like-source, like-load, like-copybook, or members of a JCL procedure library.

The dependency relationship of interest is selectable at the time the **Impact Analysis** function is requested. Only one dependency type may be queried at a time.

## Invoking and Viewing the Impact Analysis Report

Invoking the  
Impact Analysis  
Report

The **Impact Analysis** report is invoked from the following menus in the **Serena Explorer** navigation view of the **Serena** perspective:

- *Baseline component contextual menu* — Expand the **z/OS Applications** node for the relevant ZMF server. Navigate to the desired application node and expand it, then expand the **Baseline** node and the node for the relevant baseline library, then right-click on the component whose build of materials you wish to view. When the contextual menu displays, select **Impact Analysis**.

See [Chapter 2, "ZMF Operations on Baseline Library Components" on page 67](#) for more information.

- *Package component contextual menu* — Expand the **z/OS Applications** node for the relevant ZMF server. Navigate to the desired application node and expand it, then expand the **Packages** node, then the node for the relevant change package and staging library, then right-click on the component whose build of materials you wish to view. When the contextual menu displays, select **Impact Analysis**.

See [Chapter 2, "ZMF Operations on Package Components" on page 69](#) for more information.

- *Promotion component contextual menu* — Expand the **z/OS Applications** node for the relevant ZMF server. Navigate to the desired application node and expand it, then expand the **Promotion** node, then the node for the relevant promotion site, promotion level, and library type, then right-click on the component whose build of materials you wish to view. When the contextual menu displays, select **Impact Analysis**.

See [Chapter 2, "ZMF Operations on Promoted Components" on page 73](#) for more information.

Viewing the  
Impact Analysis  
Report

The requested impact analysis report displays in a ZMF table view under the **Impact Analysis** tab in the lower right pane of the perspective.



## Impact Analysis Wizard Step-by-Step

To request an impact analysis report, perform the following steps.

- 1 Select the **Impact Analysis** option from the subordinate component's contextual menu.
- 2 The **Component Impact Analysis** parameters window prompts you to specify a description of the subordinate component(s). It also requests a dependency relationship and any results filters you might want to apply.

- a Under the heading **Subordinate Component Information**, specify the **Name**, **Library Type**, and **Application** of the subordinate component whose impact is to be analyzed. Observe the rules in the table for each subordinate component type.

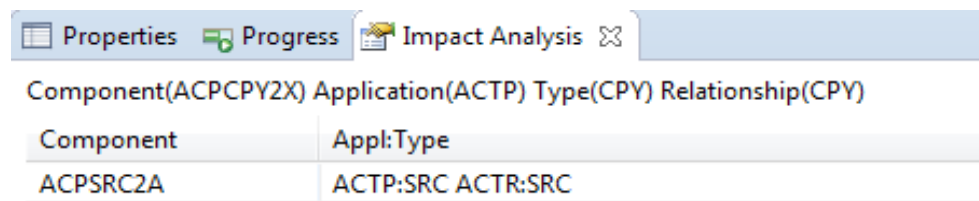
Type of Component	Field Entries
<b>Copybook</b>	<ul style="list-style-type: none"> <li>■ <b>Name</b> of referenced copybook, or a pattern with at least one literal character and an asterisk (*) wildcard.</li> <li>■ <b>Library Type</b> must be CPY or a like-copybook library.</li> <li>■ <b>Application</b> where the subordinate component is used. May be a pattern or a standalone asterisk (*) wildcard for cross-application components.</li> </ul>

Type of Component	Field Entries
<b>Data Set</b>	<ul style="list-style-type: none"> <li>■ <b>Name</b> of referenced data set, or a pattern with at least one literal character and an asterisk (*) wildcard. Data set names or patterns may match fully qualified physical data set names, symbolic data set names such as &amp;SYMBOLIC, or Generation Data Group (GDG) names without a relative generation number. NOTE: Temporary files such as &amp;&amp;TEMP are not supported.</li> <li>■ <b>Library Type</b> must be an asterisk (*) wildcard.</li> <li>■ <b>Application</b> where the subordinate component is used. May be a pattern or a standalone asterisk (*) wildcard for cross-application components.</li> </ul>
<b>JCL Procedure</b>	<ul style="list-style-type: none"> <li>■ <b>Name</b> of a JCL procedure referenced by other JCL components in an EXEC statement, or a pattern with at least one literal character and an asterisk (*) wildcard.</li> <li>■ <b>Library Type</b> must be an asterisk (*) wildcard.</li> <li>■ <b>Application</b> where the subordinate component is used. May be a pattern or a standalone asterisk (*) wildcard for cross-application components.</li> </ul>
<b>Subroutine</b>	<ul style="list-style-type: none"> <li>■ <b>Name</b> of statically linked subprogram module, or a pattern with at least one literal character and an asterisk (*) wildcard.</li> <li>■ <b>Library Type</b> must be LOD or a like-load library.</li> <li>■ <b>Application</b> where the subordinate component is used. May be a pattern or a standalone asterisk (*) wildcard for cross-application components.</li> </ul>
<b>Program</b>	<ul style="list-style-type: none"> <li>■ <b>Name</b> of a program referenced in the PGM= parameter of a JCL EXEC statement, or a pattern with at least one literal character and an asterisk (*) wildcard.</li> <li>■ <b>Library Type</b> must be an asterisk (*) wildcard.</li> <li>■ <b>Application</b> where the subordinate component is used. May be a pattern or a standalone asterisk (*) wildcard for cross-application components.</li> </ul>

**b** In the **Relationship** field, select from the pull-down list the dependency relationship for which an impact analysis report is desired. Options:

- Copybook — Requests all higher-level source members that reference the named component as a copybook. Returns SRC, like-source, CPY, and like-copybook objects only.
- DSN Name/Symbol — Requests all data set name-to-symbol mappings that reference the named data set. Returns JCL components that include the data set as the value of a DSN= or DSNAME= parameter in a DD statement. Returns JCL, like-JCL, PRC, and like-procedure objects only.
- JCL Procedure — Requests all JCL procedures that invoke the named component in an EXEC statement. Returns JCL, like-JCL, PRC, and like-procedure objects only.

- PGM Name/Symbol — Requests all program name-to-symbol mappings that reference a target program. Returns any JCL components that include the named program as the value of a PGM= parameter in an EXEC statement. Returns JCL, like-JCL, PRC, and like-procedure objects only.
  - Subroutine — Requests all programs that call the named subroutine (that is, all higher-level load modules that reference the named, statically link-edited subprogram). Returns LOD and like-load objects only.
- c Under the heading **Superior Component Information (Filtering)**, retain the default asterisk (\*) wildcards to see all returned results, or enter a pattern with wildcards to filter results by component **Name**, **Library Type**, or **Application**.
- d Click the **Case Sensitive Query** checkbox if the search for component dependencies should respect case in component, library, and application names. This option is frequently desirable with z/OS USS HFS components.
- e Click **OK** to submit the query.
- 3 A list of higher-level components matching your search criteria are returned under the **Impact Analysis** tab in the bottom right pane of the perspective. The search parameters are shown in the information bar below the tab name. For example:

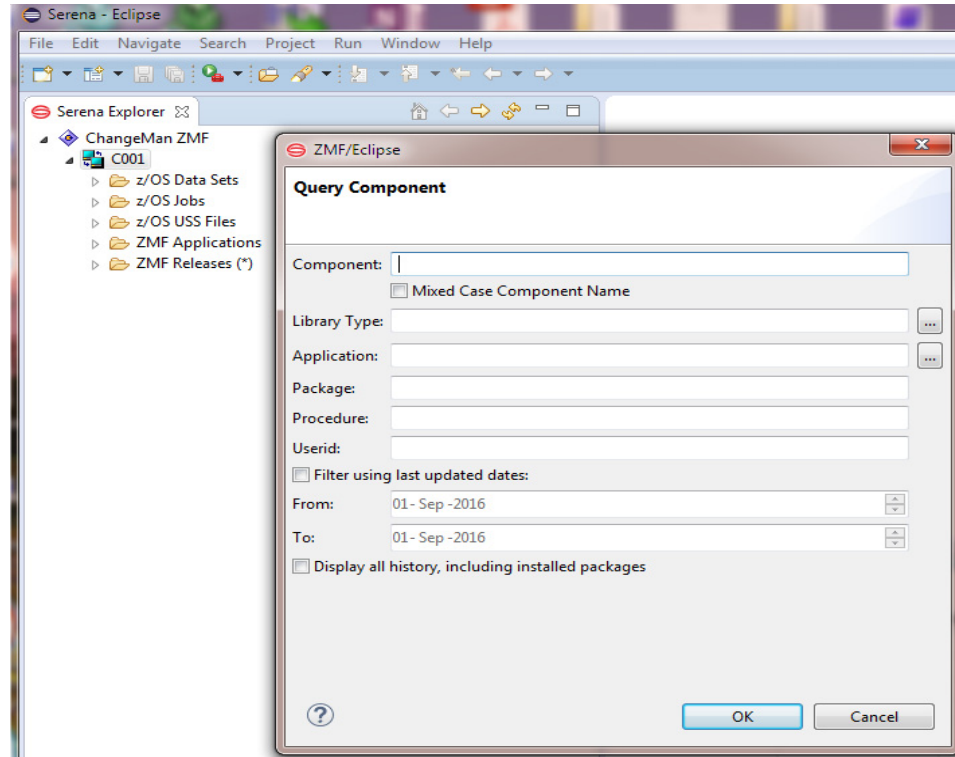


Component	Appl:Type
ACPSRC2A	ACTP:SRC ACTR:SRC

## Query Component Functionality

Take the following actions to access the **Query Component** function:

- Right-click on the desired ZMF server listed under the Server node of the Serena Explorer. The ZMF Server contextual menu is displayed.
- Select **Query Component**. The Query Component dialog is displayed:



- Fill in the fields in the **Query Component** dialog to specify the desired component query criteria. The dialog has the following fields:

Field	Description
Component	Specify the full name of the desired component or a pattern that contains a wildcard to specify all components that satisfy the pattern selection criteria.
Mixed Case Component Name	Check this checkbox to indicate that the target component has a mixed case name.
Library Type	Specify the component library type or a pattern that contains a wildcard character. <b>Required.</b>
Application	Specify the name of the target application to restrict the search to a specific application.
Package	Specify the Package ID to restrict the search to a specific package.
Procedure	JCL or procedure name.
Userid	TSO userid.
Filter using last updated dates:	Check this checkbox to restrict the displayed results to components that were last updated between the <b>From</b> and <b>To</b> dates specified.
From	From date. Only active if the <b>Filter using last updated dates</b> checkbox is checked.

Field	Description
To	To date. Only active if the <b>Filter using last updated dates</b> checkbox is checked.
Display all history including installed packages	Check this checkbox to display all history including installed packages for the components that satisfy the query specifications.

- Click **OK**. Query results are displayed in the **Query Component** view:

The screenshot shows a software interface with three tabs: 'Properties', 'Query Component' (which is active), and 'Progress'. Below the tabs is a table with the following columns: Component, Library Type, Package, Package Status, Type, Promotion, VV.MM, Date, Size, Procname, User, and SETSSI. The table contains 25 rows of data, including component names like ACPCPYSB, ACPCPYSA, ACPCPYCF, etc., and their associated metadata.

Component	Library Type	Package	Package Status	Type	Promotion	VV.MM	Date	Size	Procname	User	SETSSI
ACPCPYSB	CPY	ACTP000073	DEV			01.01	2016/07/28 21:35:15	3		JPRESTO	6A6A7213
ACPCPYSA	CPY	ACTP000073	DEV			01.01	2016/07/28 21:35:11	3		JPRESTO	6A6A720F
ACPCPYCF	CPY	ACTP000028	DIS			01.01	2015/01/20 22:51:08	1		JPRESTO	678ED35C
ACPCPYCF	CPY	ACTP000032	DEV	Rename		00.00	2015/01/12 21:00:56			JPREST2	67842D88
ACPCPYCF	CPY	ACTP000033	DEV	Rename		00.00	2015/01/12 20:51:49			JPRESTO	67842B65
ACPCPYCF	CPY	ACTP000073	DEV			01.01	2016/07/28 21:35:06	3		JPRESTO	6A6A720A
ACPCPYCE	CPY	ACTP000073	DEV			01.01	2016/07/28 21:35:03	3		JPRESTO	6A6A7207
ACPCPYCD	CPY	ACTP000073	DEV			01.01	2016/07/28 21:34:59	3		JPRESTO	6A6A7203
ACPCPYCC	CPY	ACTP000073	DEV			01.01	2016/07/28 21:34:55	3		JPRESTO	6A6A71FF
ACPCPYCB	CPY	ACTP000032	DEV	Rename		00.00	2015/01/12 22:00:21			JPREST2	67843B75
ACPCPYCB	CPY	ACTP000073	DEV			01.01	2016/07/28 21:34:50	3		JPRESTO	6A6A71FA
ACPCPYCA	CPY	ACTP000032	DEV			03.00	2016/04/06 20:52:57	3		JPRESTO	69D56EA9
ACPCPYCA	CPY	ACTP000073	DEV			01.01	2016/07/28 21:34:47	3		JPRESTO	6A6A71F7
ACPCPY99	CPY	ACTP000073	DEV			01.01	2016/07/28 21:34:43	5		JPRESTO	6A6A71F3
ACPCPY98	CPY	ACTP000073	DEV			01.01	2016/07/28 21:34:39	4		JPRESTO	6A6A71EF
ACPCPY97	CPY	ACTP000073	DEV			01.01	2016/07/28 21:34:35	4		JPRESTO	6A6A71EB
ACPCPY96	CPY	ACTP000073	DEV			01.01	2016/07/28 21:34:31	4		JPRESTO	6A6A71E7
ACPCPY95	CPY	ACTP000073	DEV			01.03	2016/07/28 21:34:25	4		JPRESTO	6A6A71E1
ACPCPY94	CPY	ACTP000073	DEV			01.01	2016/07/28 21:34:21	4		JPRESTO	6A6A71DD
ACPCPY93	CPY	ACTP000073	DEV			01.01	2016/07/28 21:34:17	5		JPRESTO	6A6A71D9



## Chapter 6

---

# ChangeMan ZMF Package Functions

Creating a Change Package	176
Deleting a Change Package	190
Undeleting a Change Package	191
Editing Package Properties	192
Removing Scratch Records from a Package	193
Removing Rename Records from a Package	193
Freezing a Package	194
Unfreezing and Refreezing Package Components	197
Promoting a Package	199
Demoting a Package	209
Auditing a Package	213
Resetting Audit Lock	221
Approving or Rejecting a Package for Installation	222
Rebuilding Installation JCL	224
Backing a Package Out of Production	225
Reverting a Package to Development Status	227
Querying Packages	228
Querying Package Components	235
Viewing the Component Work List	237
Viewing Source-to-Load Relationships	237
Viewing Scratch/Rename Components	238
Viewing Promotion History	239
Viewing Promotion Libraries	239
Viewing Baseline Libraries	240
Displaying Site Activities	241

# Creating a Change Package

## Functional Description

Changes and additions to the code base of an application are organized into *change packages* by ChangeMan ZMF. You can create new change packages from the workbench using the **Create Package** function.

The **Create Package** option invokes the **Create Package** wizard, which walks you through the many steps required to create a new change package in the ChangeMan ZMF repository on the mainframe. The new package is automatically assigned a package name consisting of a four-byte application ID concatenated with a unique package number. The newly created package is automatically added to the **Serena Explorer** navigation view under the **Packages** node for the selected application.

Invoking the  
Create Package  
Function

The **Create Package** wizard is invoked from the following workbench menu:

- **Serena perspective** — In the **Serena Explorer** view, expand the node of the ZMF server hosting the repository where the package will reside. Expand the **ZMF Applications** node for that server, open the relevant folder with the application, then right-click on the name of the application to which the new package will belong. Select the **New Package** menu option.

See [Chapter 2, "Working with ZMF Applications" on page 57](#) for more information.

Results

After the **Package Create** request is submitted, a message dialog box displays the success or failure of the request.

## Package Create Procedure

To create a new change package for an application, perform the following steps.

- 1 Invoke the **Create Package** wizard from the contextual menu for the desired application in the **Serena Explorer** navigation view.
- 2 When the **Package Create** wizard displays the **Package General Information** window, enter the requested information, then click **Next** to advance to the next window.
- 3 Multiple windows will request additional package parameters. The actual windows displayed by the wizard will vary depending on the method of package creation you choose (such as the Quick Method) and on the ZDDOPTS configuration parameters defined by your ZMF administrator.
- 4 When all information is entered, click **Finish** to submit the **Create Package** job for execution.
- 5 A message dialog from ChangeMan ZMF reports the results of the package creation request. Click **OK**.

## Package Create Wizard Screens

The **Package Create** wizard displays up to nine dialog screens that correspond to the nine panels in the ISPF interface to ChangeMan ZMF. The actual screens displayed depend on several factors:



- Package "level" (simple, participating, or complex/super)
- Site type (D, DP, or All) targeted by the ZMF instance managing the package
- Settings specified in the PKGCREAT member of ZDDOPTS for the ZMF instance
- Data entered in the first few screens of the wizard itself

Quick Method The "Quick Method" of package creation (also known as the "Short Method" in the ISPF interface to ZMF) skips all screens that are optional for the selected package level.

The following table lists the **Package Create** wizard screens and indicates which screens are displayed for different data entry methods and package types. Screens with further display conditions are identified by a numbered note that describes those conditions. Wizard screens, if displayed, appear in the order listed from top to bottom in the table. Click on the screen name to skip to the instructions for that wizard screen.

Screen Display  
Conditions

Wizard Screen	Quick Method			Long Method		
	Simple	Participating	Complex & Super	Simple	Participating	Complex & Super
<a href="#">General Information</a>	X	X	X	X	X	X
<a href="#">Description</a>				X	X	X
<a href="#">Installation Instructions</a>				X	X	
<a href="#">Scheduling Dependencies</a>				1, 2	1, 2	
<a href="#">Affected Applications</a>					X	
<a href="#">Participating Packages</a>			X			X
<a href="#">User Variables</a>	3	3		3	3	
<a href="#">All Site Information</a>	4	4		4	4	
<a href="#">DP Site Information</a>	5	5		5	5	

NOTES:

- 1 Displayed only if **Other** is specified for **Scheduler** type on the **Installation Instructions** screen.
- 2 Scheduling dependencies are not allowed for unplanned changes. Change package type to "Planned" if scheduling dependencies must be enforced.
- 3 Displayed only if user-defined variables are configured in the PKGCREAT member of ZDDOPTS.
- 4 Displayed only if the ZMF instance is managing an "All" site.
- 5 Displayed only if the ZMF instance is managing a "DP" site.



**NOTE** For more information about creating a change package, refer to the *ChangeMan ZMF User's Guide*.

## General Information About the Package

The first dialog screen displayed by the **Package Create** wizard is the **Package General Information** screen. Package type, level, duration, and basic reporting information are entered on this screen. Editable fields are pre-populated with their most recent values.

The screenshot shows a dialog box titled "ZMF/Eclipse - Package Create" with a sub-header "Package General Information". Below the sub-header is the instruction "Enter Package General Parameters below". The dialog contains several input fields and checkboxes:

- Package title: Default Package Title
- Application: ACTP
- Requestor name: John Doe
- Requestor phone: Phone 123456
- Package level: Simple (dropdown menu)
- Work request: TN3270
- Department: IDD
- Create using quick method
- Unplanned change
- Temporary change
- Unplanned reason: (empty dropdown menu)
- Temporary days: 5
- Release: (empty dropdown menu)
- Release Area: (empty dropdown menu)

At the bottom of the dialog are four buttons: a help icon (?), "< Back", "Next >", and "Finish" (highlighted in blue), and "Cancel".

The following table describes the fields on the **Package General Information** screen.

Field/Box	Description
<b>Package Title</b>	Type a title for the package. This field is case-sensitive and has a maximum length of 72 characters.
<b>Application</b>	Displays the name of the application where the changes in this change package will be applied. You must have UPDATE access to this application defined in ZMF and in your security system.
<b>Requestor Name</b>	Type the name of the person requesting this change package. This field is not case-sensitive and has a maximum length of 25 characters.
<b>Requestor Phone</b>	Type a telephone number for the package requestor (the person whose name you entered in the <b>Requestor Name</b> field). This field has a maximum length of 15 characters.
<b>Work Request ID</b>	Type a work request identifier. Values may be alphanumeric, may include embedded spaces, and are not case-sensitive. Maximum length is 12 characters. This field may be used by the ChangeMan ZMF INFO option.
<b>Department</b>	Type the functional department associated with this change package. This field is used for reporting only. This field is not case-sensitive and has a maximum length of 4 characters.

Field/Box	Description
<b>Package Level</b>	<p>Select a package level (simple, participating, or complex/super).</p> <p><b>Simple</b> — The changes in this package are not related to any other package. In addition, the package does not affect any other application, nor does it require changes to software or operational procedures in other applications. (Default)</p> <p><b>Participating</b> — This package is related to one or more other participating packages which are associated with the same complex or super package. SYSLIB statements in build jobs include staging libraries from other participating packages under the same complex or super package.</p> <p><b>Complex or Super</b> — This package is the administrative parent of two or more participating packages that have interdependent changes to software or operational procedures. If you select this option, a subsequent dialog box will prompt you for a list of participating packages. Remote sites and installation dates are set individually in each participating package. No staging libraries are allocated to complex or super packages.</p> <p><b>NOTE:</b> You must define a <b>Complex</b> or <b>Super</b> package as such when the package is created. You cannot redefine a simple or participating package as a complex or super package later using the <b>Package Properties</b> editing wizard.</p>
<b>Create Using Quick Method</b>	<p>Check this option to create a package using the ZMF "Short" method of package creation. This method automatically takes default values wherever possible, allowing the wizard to skip several screens.</p> <p>If you do not select this option, the "Long" method of package creation will be used. Most or all wizard screens will display.</p>
<b>Unplanned Change</b>	<p>Select a ZMF package type (planned or unplanned).</p> <p><b>Planned</b> — Leave the <b>Unplanned Change</b> box unchecked for planned change packages. Planned packages follow the standard package life cycle for permanent or temporary change packages. Complex, super, and participating packages must be planned.</p> <p><b>Unplanned</b> — Check the <b>Unplanned Change</b> box for unplanned change packages. Unplanned packages contain unscheduled changes and possibly emergency fixes. Depending on global and application administrative settings, some package life cycle steps and requirements may be skipped. Complex, super, and participating packages may not be unplanned.</p> <p><b>NOTE:</b> Package type (planned or unplanned) is independent of package life cycle (permanent or temporary). Either type may follow either life cycle.</p>
<b>Unplanned Reason</b>	<p>If you selected <b>Unplanned Change</b> as the package type, choose a reason code from the drop-down list.</p>

Field/Box	Description
<b>Temporary Change</b>	<p>Select a package life cycle (permanent or temporary).</p> <p><b>Permanent</b> — Leave the <b>Temporary Change</b> box unchecked if the change package is permanent. At install time, the package components will be installed in the defined production libraries and baselined in the repository. Complex, super, and participating packages must be permanent.</p> <p><b>Temporary</b> — Check the <b>Temporary Change</b> option for temporary change packages. At install time, temporary changes will be installed in special override libraries concatenated on top of your production library concatenations. The changes are automatically removed after a specified number of days has passed. (Install duration is entered in the <b>Temporary Days</b> field.) Baseline and production libraries are not updated for temporary packages. Complex, super, and participating packages may not be temporary.</p> <p><b>NOTE:</b> Package life cycle (permanent or temporary) is independent of package type (planned or unplanned). Either type may follow either life cycle.</p>
<b>Temporary Days</b>	<p>If you selected <b>Temporary Change</b>, enter the number of days that the changes should be retained in override libraries in production. (This field is disabled for permanent packages.)</p> <p><b>NOTE:</b> Change duration is the number of calendar days, not the number of full 24-hour periods since the time-of-day at which actual installation occurred. ZMF increments the count of calendar days at 24:00 system time nightly.</p>

Enter the general package information and then click **Next** to continue.

## Package Description

The **Package Description** screen captures a text description of changes contained in the package. It is displayed only if you are creating a package using the long method (that is, the **Create using quick method** box was *not* checked on the general information screen).

Enter up to 46 lines of text to in the **Package Description** field. Lines may not exceed 72 characters in length.

Click **Next** to continue. You may also click **Back** to make changes.

## Package Installation Instructions

The **Installation Instructions** screen requests information about installation scheduling, options in the event of an installation problem, and any special instructions for this particular package. It is displayed only if you are creating a simple or participating package using the long method.

The following table describes the fields on the **Installation Instructions** screen.

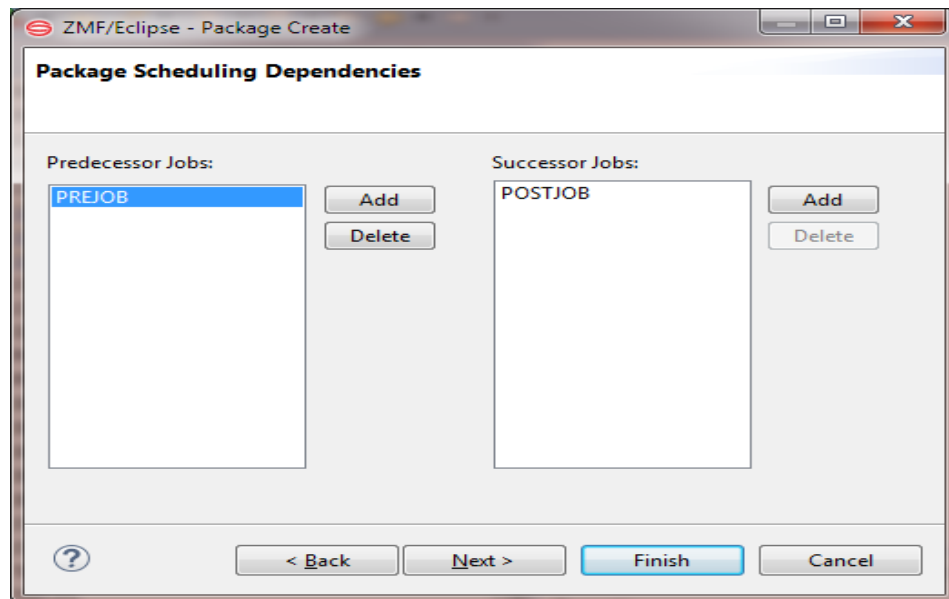
Field/Box	Description
<b>Contingency</b>	<p>Select the option that describes what should be done if your change package fails to install properly:</p> <ul style="list-style-type: none"> <li>■ <b>Hold production and contact analyst</b> The analyst name and phone number must be supplied on the site information screen displayed later. This is not the change requestor named on the general information screen.</li> <li>■ <b>Backout change and continue production</b></li> <li>■ <b>Other</b> If you select <b>Other</b>, you must provide an explanation in the corresponding text box of this dialog.</li> </ul>
<b>Scheduler</b>	<p>Select an option to determine how the package installation process will be initiated.</p> <p><b>ChangeMan</b> — Package installation will be initiated by the internal ChangeMan ZMF scheduler. When the install date and time arrive, ZMF will install the package at the defined site.</p> <p><b>Manual</b> — Package installation will be initiated whenever approval is received from the final package approver.</p> <p><b>Other</b> — Package installation will be initiated by an external scheduler such as CA-7, CA-Scheduler, or CA-ADC2. Scheduling records are inserted into the scheduler database by ZMF when the package is distributed. If this option is selected, the <b>Scheduling Dependencies</b> screen displays after this dialog.</p> <p><b>NOTE:</b> Scheduler defaults and available options are determined by global administration settings in ChangeMan ZMF.</p> <ul style="list-style-type: none"> <li>■ When <b>ChangeMan</b> is the global default, you can change the <b>Scheduler</b> value to <b>Manual</b>, but not to <b>Other</b>.</li> <li>■ When <b>Manual</b> is the global default, you cannot change the <b>Scheduler</b> value.</li> <li>■ When <b>Other</b> is the global default, you can change the <b>Scheduler</b> value to either <b>ChangeMan</b> or <b>Manual</b>.</li> </ul>
<b>Installation Instructions</b>	Enter up to 46 lines of instructions for installing and backing out the change package. Lines may not exceed 72 characters each.

Click **Next** to continue, or click **Back** to return to the previous screens and make changes.

## Installation Scheduling Dependencies

The **Scheduling Dependencies** screen asks for any jobs that must be run by an *external* scheduler before and/or after the execution of the *first* install job defined for the package in ChangeMan ZMF. It is displayed only if you selected **Other** for the **Scheduler** type on the **Installation Instructions** screen.

The information you enter is inserted into your external job scheduler database by ChangeMan ZMF job CMN17, which runs when the package is distributed.



The following table describes the fields on this screen.

Field/Box	Description
<b>Predecessor Jobs</b>	Jobs in your automated job scheduler that must run before the first install job for the package.
<b>Successor Jobs</b>	Jobs in your automated job scheduler that must run after the first install job for the package.

Scheduler  
Controlled  
Install Jobs

Enter the predecessor and successor jobs that must run before and after the *first* install job defined to ZMF for your package. The predecessor and successor jobs will be initiated by the external scheduler, not by ZMF.

- To add a job name, click the **Add** button and enter a job name when prompted.
- To delete a job name, select it and then click the **Delete** button.
- To change a job name, delete it and then add a new one.



**TIP** If you do not know which scheduler jobs to run at this time, leave the predecessor and successor job list fields blank. You can provide this information later using the **Package Properties** wizard. (See "[Editing Package Properties](#)" on page 192.)

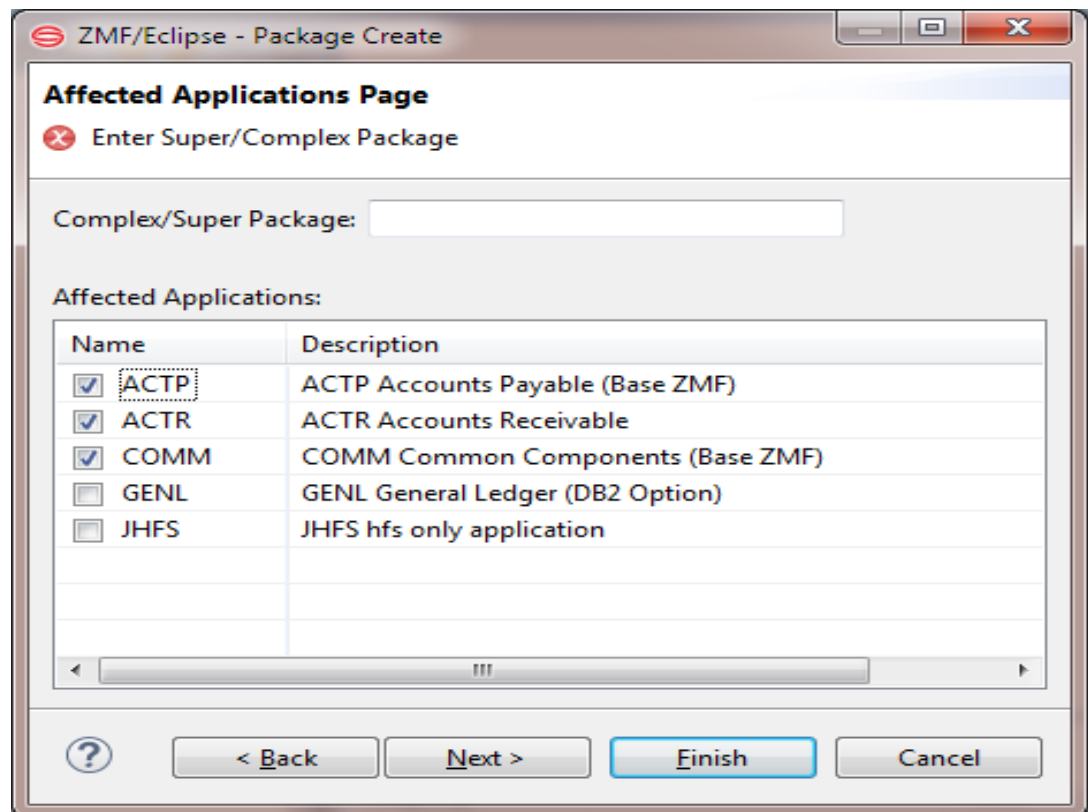
ZMF Controlled  
Install Jobs

After completion of the externally-scheduled first install job (with its predecessors and successors), the remaining install jobs defined for this package at the target location are executed by ZMF.

Click **Next** to continue, or click **Back** to return to previous screens and make changes.

## Applications Affected by Change Package

The **Affected Applications Page** screen requests you to identify any additional applications affected by the newly created change package. It is displayed only if you are creating a participating package using the long method.



The following table describes the fields on the **Affected Applications Page** screen.

Field/Box	Description
<b>Complex/Super Package</b>	Enter the package ID of the complex or super package that this participating package will be managed under. The complex or super package must be in open (OPN) status.
<b>Affected Applications</b>	<p>From the checklist, select <i>additional</i> applications <i>other</i> than the application in which the package is being created that are affected by your change package.</p> <p>When you freeze the participating package, the "Interfacing Approvers" defined for each affected application are assigned to your package. These approvers must sign off on the package before it can be installed.</p> <p>The applications listed are selected using the workbench application filters specified for the ChangeMan ZMF instance. These are the same applications that display for the instance in the <b>Remote Systems</b> explorer.</p>



**TIP** If you do not know which applications are affected by the package at this time, you can provide the needed information later using the **Package Properties** wizard.

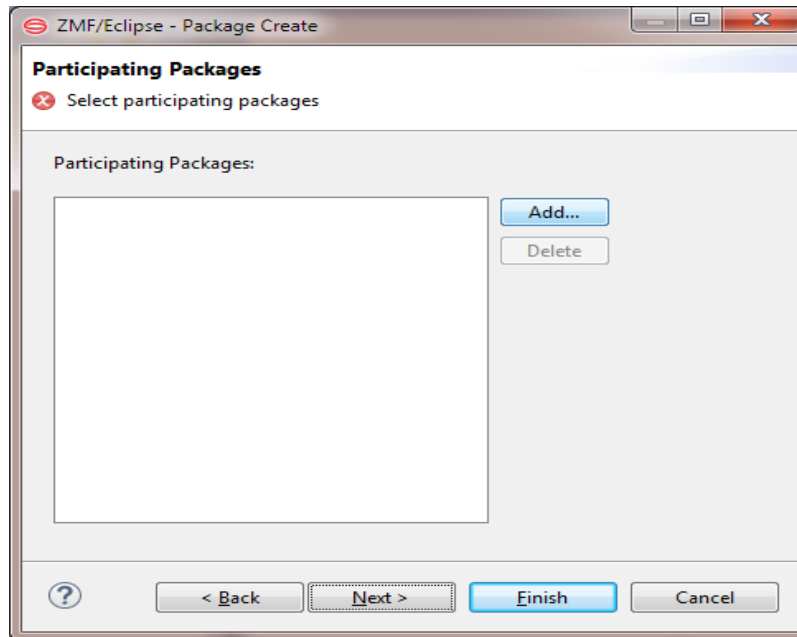
Click **Next** to continue, or click **Back** to return to the previous screens and make changes.



## Participating Packages for a Complex/Super Package

The **Participating Packages** screen allows you to associate existing participating packages with the complex or super package you are creating. This screen is displayed only if you are creating a complex or super package using the long method.

Valid participating packages must be in DEV status, must have an install date that has not passed, and must not be associated with another complex or super package.



- To add a package to the list, click **Add** and enter a package ID when prompted.
- To delete a package from the list, select it and then click **Delete**.
- To change a package ID, delete the package and then add a new one.



**TIP** If you do not know the package IDs of the participating packages associated with this complex or super package, you can provide this information later using the **Package Properties** wizard.

Click **Finish** to create the package. Click **Back** to return to the previous screens and make changes.

## User-Defined Package Variables

The **Package User Variables** screen requests data entry for any customer-defined package variables. It is displayed only when you are creating a simple or participating package, and only when customer-defined variables have been enabled for client data entry in the PKGCREAT member of ZDDOPTS.

Data entry fields are pre-populated with the most recently entered values. To update the values for this package, select a variable in the **Variable Name** column and then enter a value in the **Variable Value** column. Follow the data validation rules for these entries as specified in the PKGCREAT member.

**Package User Variables**

✘ Log dsn: Enter value of type: DSNAME

Variable Name	Variable Value
Priority	5
Job name	
Log dsn	

Name:

Value:

? < Back Next > Finish Cancel



**TIP** You can update these variable values now, or provide this information later using the **Package Properties** wizard.

Click **Next** to continue, or click **Back** to return to the previous screens and make changes.

## Site Installation Information - "All" Site

The **All Site Information Page** display requests site-specific installation information for the target ZMF site. The **All Site Information Page** display is displayed only if you are creating a simple or participating package for a ChangeMan ZMF instance that is configured as an All site. (ZMF for Eclipse detects the site type automatically when you connect to it.)

**All Site Information Page**

Install sites:

Site	Install	Primary Contact	Secondary Contact
<input checked="" type="checkbox"/> U710ALL	2014/01/26 01:00 - 23:00	Primary Contact 201-Primary	Secondary Contact 201-Secondary

Selective Update

Selected Site:

Install Date:

From Time:

To Time:

Primary Contact:

Primary Phone:

Secondary Contact:

Secondary Phone:

Update the site installation information in the text boxes, as described in the following table.

Field/Box	Description
<b>Install Date</b>	The calendar date when the package should be installed at the site.
<b>From Time</b>	The earliest time of day at which installation is acceptable on the install date.
<b>To Time</b>	The latest time of day at which installation is acceptable on the install date.
<b>Primary Contact</b>	Enter the name of the primary person to contact if there is a problem with the installation at this site. Up to 25 characters may be entered.
<b>Primary Phone</b>	Enter the telephone number of the primary contact. Up to 15 alphanumeric characters may be entered.

Field/Box	Description
<b>Secondary Contact</b>	Enter the name of the secondary person to contact if there is a problem with the installation at this site. Up to 25 characters may be entered.
<b>Secondary Phone</b>	Enter the telephone number of the secondary contact. Up to 15 alphanumeric characters may be entered.

As there is only one site for an All site, click any of the update buttons (**Update Selected Site, Update Checked Sites, Update All Sites**) to update the site information.

Click **Back** to makes changes to previous screens. Click **Finish** to create the package.

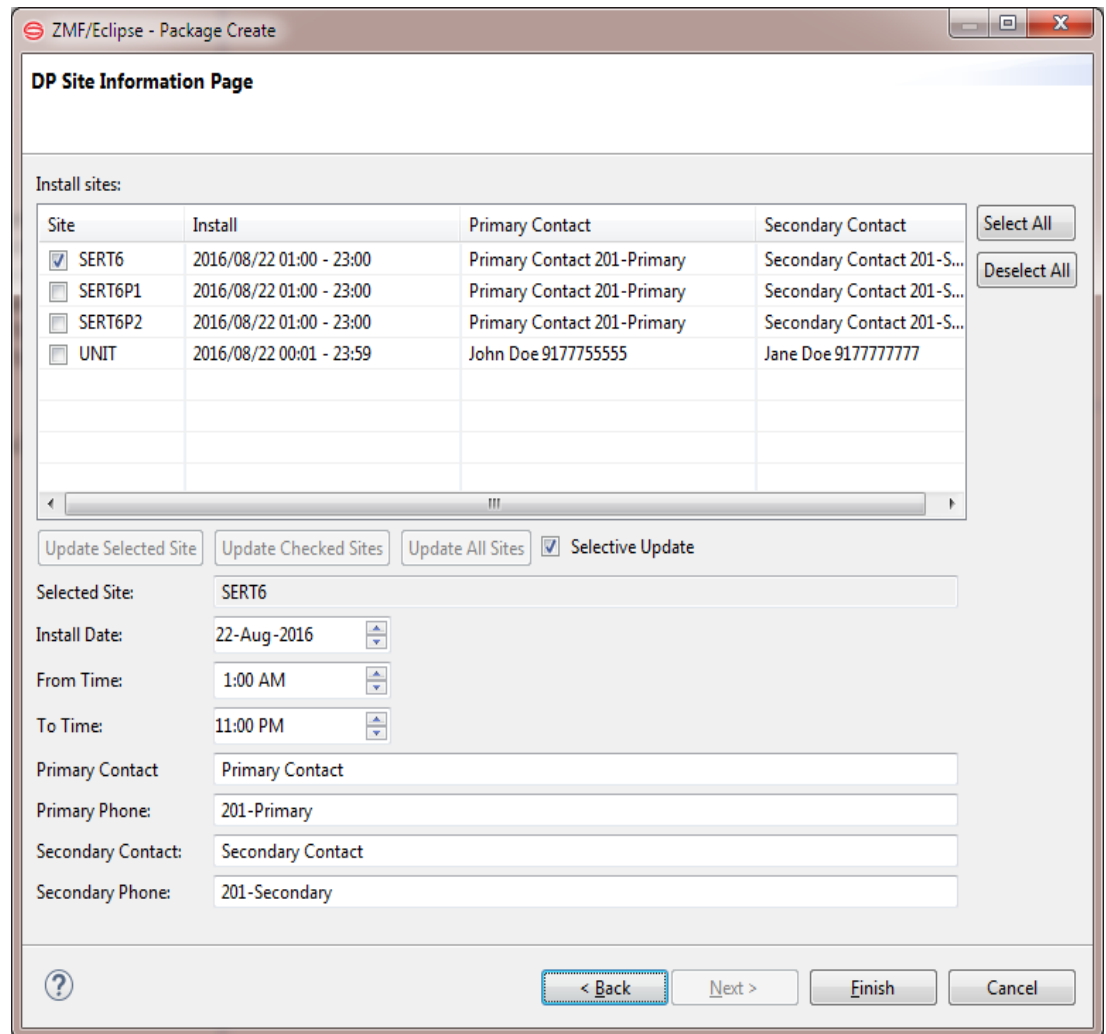
## Site Installation Information - DP Site

The **DP Site Information Page** display requests site-specific installation information for one or more target installation sites managed by a ZMF DP site. The page is displayed only for simple or participating packages managed by a ChangeMan ZMF instance that is configured as a DP site.

The page uses default values if no other values are available. After you update the site information and create a package, the next package create will be initialized with the saved values for each site.

ZMF DP sites can manage package installations across multiple production sites concurrently. At least one site is required.

**NOTE** If only one site has been defined, the Site ID field is pre-populated with that ID when a package is created. If multiple Site IDs have been defined, the Site ID field is pre-populated with the ID of the first site in the list.



The install sites available for the package are displayed in the **Install Sites** window, with associated install dates and contacts for each.

Update the site installation information for the sites to be associated with the package. Update the information in the text boxes as described in the following table.

Field/Box	Description
<b>Install Date</b>	The calendar date that the package will be installed at the site.
<b>From Time</b>	The earliest time of day when the installation should begin.
<b>To Time</b>	The latest time of day when the installation should begin.
<b>Primary Contact</b>	Enter the name of the primary person to contact if there is a problem with the installation at this site. Up to 25 characters may be entered.
<b>Primary Phone</b>	Enter the telephone number of the primary contact. Up to 15 alphanumeric characters may be entered.
<b>Secondary Contact</b>	Enter the name of the secondary person to contact if there is a problem with the installation at this site. Up to 25 characters may be entered.

Field/Box	Description
<b>Secondary Phone</b>	Enter the telephone number of the secondary contact. Up to 15 alphanumeric characters may be entered.

The information may be updated in a variety of ways using the buttons and check boxes. The functions of the buttons and check boxes on the **DP Site Information** page are:

Button/Checkbox	Function
Select All	Selects all sites in the table.
Deselect All	Deselects all sites in the table.
Update Selected Site	Updates the selected site with information from the text fields.
Update Checked Sites	Updates the checked sites with information from the text fields.
Update All Sites	Updates all sites with information from the text fields.
Selective Update	When selected, only information that has been updated in a text field is applied.

### ***Executing the Wizard***

Click **Back** to change previous wizard screens. Click **Finish** to create the package.

## **Deleting a Change Package**

### **Functional Description**

In a package context, the **Delete** function memo-deletes a change package — that is, it flags the package for later, physical deletion by the ChangeMan ZMF housekeeping job.

The following rules apply to the package **Delete** function:

- The package must be in development (DEV) status.
- Only simple and participating packages can be memo-deleted.
- Packages with promoted components cannot be memo-deleted.
- It may be necessary to delete all components in a package before the package itself can be deleted, depending on administrator settings in the ZMF repository.



**TIP** If a package is memo-deleted in error, it can be undeleted using the **Undelete** function.

After the ZMF housekeeping job runs, it is not possible to undelete a change package.

## Invoking Package Deletion

Invoking  
Package Delete

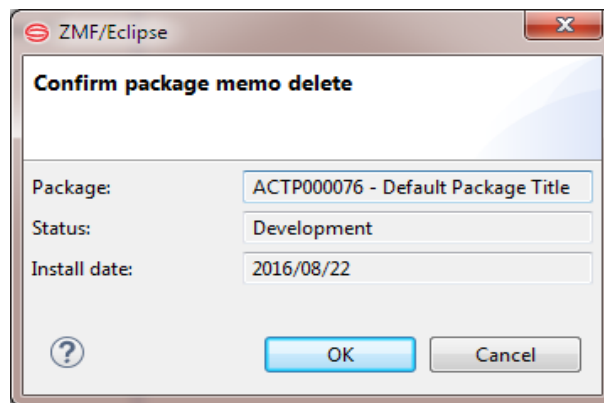
The package **Delete** function is invoked from the following menu:

- **Serena perspective** — In the **Serena Explorer** navigation view, expand the node for the ZMF server hosting the repository where the package resides. Under the node for **ZMF Applications**, expand the node for the target application and its **Packages** subnode. Right-click on the desired package to bring up its contextual menu, then select the **Delete** option.

Results After the package **Delete** request is submitted, a message dialog box displays the success or failure of the request.

## Package Delete Window

The **Package Delete** wizard displays a confirmation screen. For example:



If the named package is correct, click **OK** to memo-delete the package.

## Undeleting a Change Package

### Functional Description

In a package context, the **Undelete** function restores a memo-deleted change package that has not yet been physically deleted by the ChangeMan ZMF housekeeping job. After the ZMF housekeeping job runs, it is not possible to undelete a change package.

### Invoking Package Undelete

Invoking  
Package Undelete

The package **Undelete** function is invoked from the following menu:

- **Serena perspective** — In the **Serena Explorer** navigation view, expand the node for the ZMF server hosting the repository where the package resides. Under the node for **ZMF Applications**, expand the node for the target application and its **Packages** subnode. Right-click on the desired package to bring up its contextual menu, then select the **Undelete** option.

Results After the package **Undelete** request is submitted, a message dialog box displays the success or failure of the request.

# Editing Package Properties

## Functional Description

Many of the properties of an existing change package can be edited from the workbench using the **Package Properties** wizard. This feature is typically used to supply optional information that was skipped when using the "Quick Method" of package creation (also known as the "Short Method" in ZMF).

Invoking the  
Package Properties  
Edit Function

The **Package Properties** wizard is invoked from the following workbench menu:

- **Serena perspective** — In the **Serena Explorer** view, expand the **ZMF Applications** node and right-click on the name of the application to which the new package will belong. Select the **Package Properties** menu option. (See ["Working with ZMF Applications" on page 57.](#))

Results

After the **Package Properties** edit request is submitted, a message dialog box displays the success or failure of the request.

## Package Properties Wizard Screens

The **Package Properties** wizard displays the same set of dialog screens shown by the **Package Create** wizard. The actual screens displayed depend on several factors:

- Package "level" (simple, participating, or complex/super)
- Site type (D, DP, or ALL) targeted by the ZMF instance managing the package
- Settings specified in the PKGCREAT member of ZDDOPTS for the ZMF instance
- Data entered when the package was created

Certain structural features of a change package cannot be changed using the **Package Properties** wizard. You must delete the old package and create a new one with the appropriate organization in the following cases:

- Changing a simple or participating package to a complex or super package
- Changing a complex or super package to a simple or participating package
- Changing the application associated with a simple or participating package
- Changing the type of install site (for example, from DP to ALL)
- Adding site information to a complex or super package (install sites are associated with the individual participating packages, not the package collection)

See ["Package Create Wizard Screens" on page 176](#) for details about what information is required on specific screens.



# Removing Scratch Records from a Package

## Functional Description

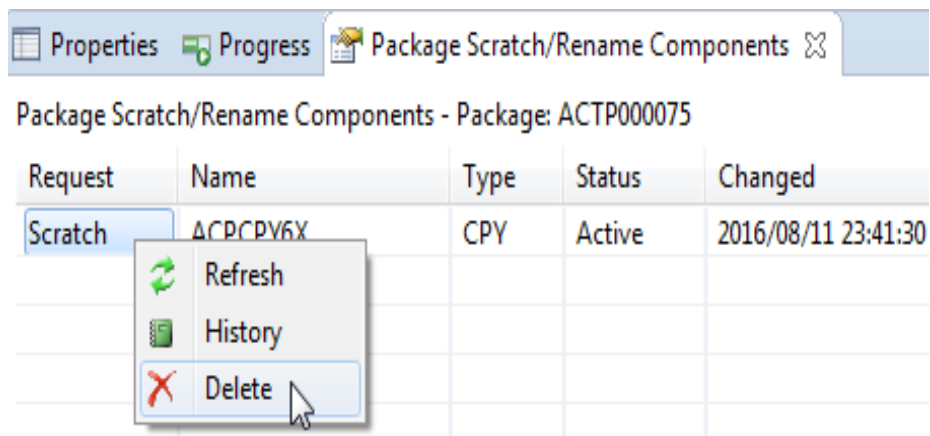
ChangeMan ZMF supports a versioned delete (or *scratch*) operation on baselined components. Control records containing instructions for this **Scratch** operation are stored in a change package for scheduled execution, and are applied to the appropriate baseline object(s) when the package is baselined. (For more information about the **Scratch** function, see [Chapter 5, "Scratching a Component under Change Control" on page 125.](#))

The **Remove Scratch** function removes these control records before the versioned delete is executed. In effect, it cancels one or more previously requested **Scratch** operations.

## Remove Scratch Records Step-by-Step

To remove scratch control records from a change package, perform the following steps.

- 1 In the **Serena Explorer** navigation view, right-click on the package to bring up its contextual menu and select the **Scratch/Rename Component** option. A new tab displays a list of components with scratch or rename records in the package. Right click on the Scratch Request and select **Delete** to remove the scratch request from the package. It should also disappear from the list on the tab.



- 2 Repeat the process to verify that the scratch request has been removed from the package, i.e. right-click on the package again to bring up its contextual menu and select the **Scratch/Rename Component** option. The tab will display a list of components with scratch or rename records in the package, and the one you deleted will be removed from that list.

# Removing Rename Records from a Package

## Functional Description

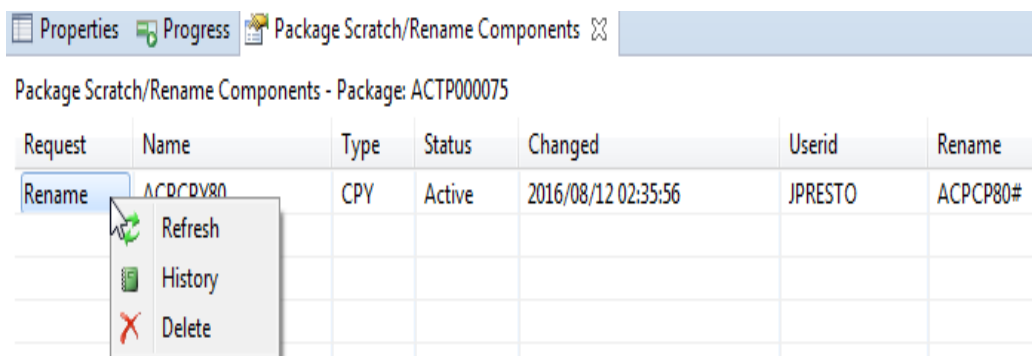
ChangeMan ZMF supports a versioned rename operation on baselined components. Control records containing instructions for this **Rename** operation are stored in a change package for scheduled execution, and are applied to the appropriate baseline object(s)

when the package is baselined. (For more information about the versioned **Rename** function, see [Chapter 5, "Renaming a Component under Change Control" on page 127.](#))

## Remove Rename Records Step-by-Step

To remove rename control records from a change package, perform the following steps.

- 1 In the **Serena Explorer** navigation view, right-click on the package to bring up its contextual menu and select the **Scratch/Rename Component** option. A new tab displays a list of components with scratch or rename records in the package. Right click on the Rename Request and select **Delete** to remove the scratch request from the package. It should also disappear from the list on the tab.



- 2 Repeat the process to verify that the rename request has been removed from the package, i.e. right-click on the package again to bring up its contextual menu and select the **Scratch/Rename Component** option. The tab will display a list of components with scratch or rename records in the package, and the one you deleted will be removed from that list.

## Freezing a Package

The **Freeze** function locks all package components and metadata to prevent further changes. At the same time, ChangeMan ZMF validates the condition of the package and its components. The status of the package and all its components is set to frozen (FRZ).

A package must be in development (DEV) status before it can be frozen. It must be frozen before the package approval process can begin.

After a package is frozen, you may selectively unfreeze a subset of package components to make changes, and then selectively refreeze them again. (See ["Unfreezing and Refreezing Package Components" on page 197.](#))

If you need to add a new component to a package, you must revert the entire package to development (DEV) status first. (See ["Reverting a Package to Development Status" on page 227.](#))

Refer to the *ChangeMan ZMF User's Guide* for additional information.

## Invoking the Freeze Package Function

The **Freeze** function is invoked from the following workbench menus:

- **Serena perspective** — In the **Serena Explorer** navigation view, expand the **ZMF Applications** node, then expand the node for the application to which the desired package belongs. Right-click on the name of the package to be frozen. When the contextual menu displays, select the **Freeze** option.

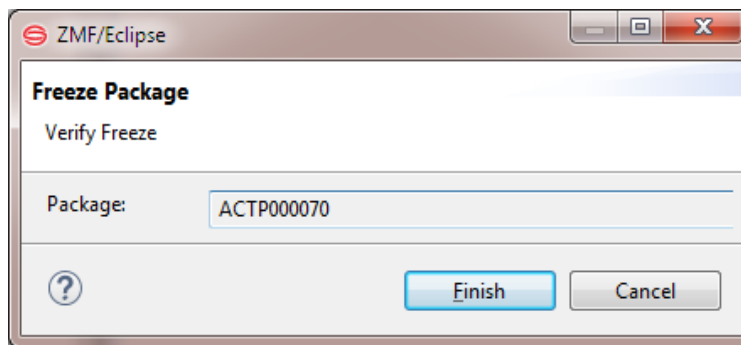
See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

- **Java perspective** — In the **Package Explorer** navigation view, navigate to a project that is shared with a ZMF package. Right-click on the project to bring up its contextual menu, then open the **Team** submenu and select the **ZMF Package Freeze** option.

See [Chapter 3, "Project-Level ZMF Functions" on page 102](#) for more information.

## Working with the Freeze Package Dialog

The **Freeze Package** dialog box asks you to confirm that the displayed package name is in fact the package you want to freeze.



If the package name is correct, click **Finish** to process the **Freeze** request.

## Unfreezing and Refreezing Package Categories

After a package is frozen, you can unfreeze selected package categories to make changes, and then refreeze selected package categories again and proceed with the approval process.

### Unfreezing Package Categories

To unfreeze selected package categories:

- In the Serena perspective right-click on the selected package name to bring up the contextual menu.
- Select **Unfreeze** from the menu. The **Unfreeze** dialog is displayed. The **Unfreeze** dialog has the following fields:
  - Package ID and name
  - The following package categories and their current freeze status:

Package Category	Description
General	Package description, control information, and installation instructions.
Non-Source	Non-source package components.
Source and Load	Source components and the target component types in source-load relationships.
Utilities	Scratch and rename utility requests.
Site	Package installation site and schedule information.
Custom Forms	Online forms in the change package.

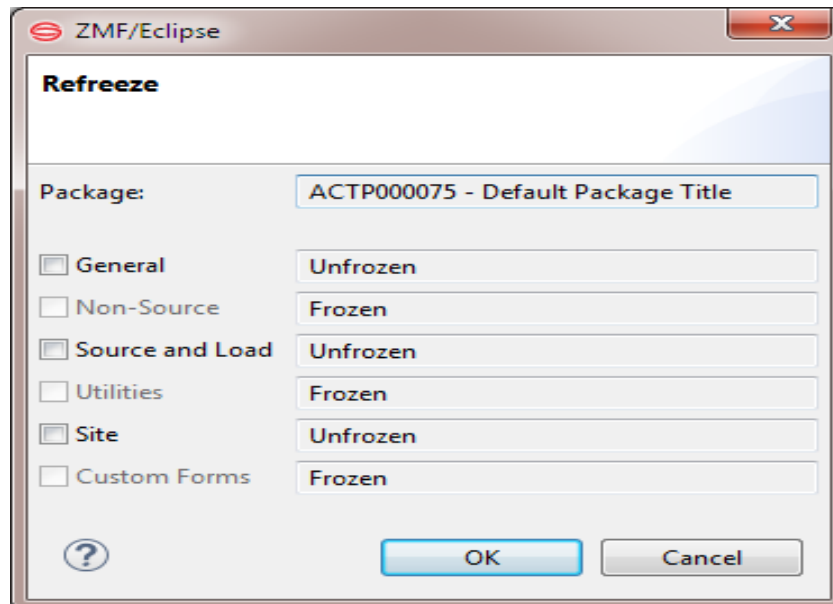
- Place a check in the checkbox that precedes the name of the package category that you want to unfreeze. Package categories that you cannot unfreeze (because they are already unfrozen, for example) are grayed out.
- Click **OK**.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

## Refreezing Package Categories

To refreeze selected package categories:

- In the Serena perspective right-click on the selected package name to bring up the contextual menu. Select **Refreeze Package** from the menu. The **Refreeze** dialog is displayed:



- The **Refreeze** dialog has the following fields:
  - Package ID and name
  - The following package categories, preceded by a checkbox, and their current freeze status:

Package Category	Description
General	Package description, control information, and installation instructions.
Non-Source	Non-source package components.
Source and Load	Source components and the target component types in source-load relationships.
Utilities	Scratch and rename utility requests.
Site	Package installation site and schedule information.
Custom Forms	Online forms in the change package.

- Place a check in the checkbox that precedes the name of the package category that you want to refreeze. Package categories that you cannot refreeze (because they are already frozen, for example) are grayed out, and click OK.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

## Unfreezing and Refreezing Package Components

After a package is frozen, you can selectively unfreeze package components to make changes, then selectively refreeze them again and proceed with the approval process.

If you need to add a new component to a frozen package, you must revert the package to development (DEV) status first. (See ["Reverting a Package to Development Status" on page 227](#).) After the component is added, use the **Package Freeze** function, not **Refreeze Components**, to freeze the package again.

### Unfreezing Package Components

Invoking the Unfreeze Components Function

The **Unfreeze Components** repository function is invoked from the following menus:

- **Serena perspective** — In the **Serena Explorer** view, expand the **ZMF Applications** node, then expand the node for the application to which the frozen package belongs. Right-click on the name of the frozen package. When the contextual menu displays, select the **Unfreeze Components** option.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

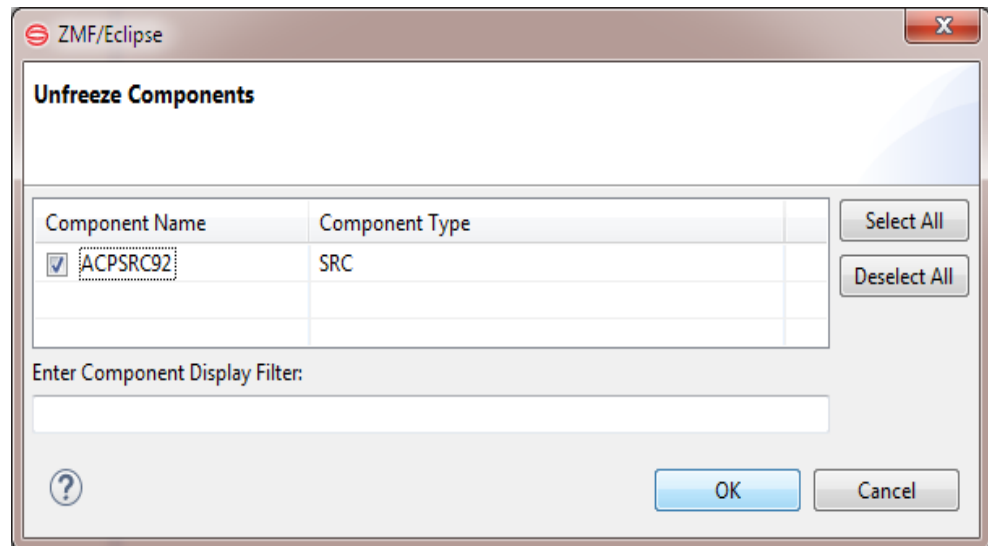
- **Java perspective** — In the **Package Explorer** navigation view, navigate to a project that is shared with a ZMF package. Right-click on the project to bring up its contextual menu, then open the **Team** submenu and select the **ZMF Package Freeze** option.

See [Chapter 3, "Project-Level ZMF Functions" on page 102](#) for more information.

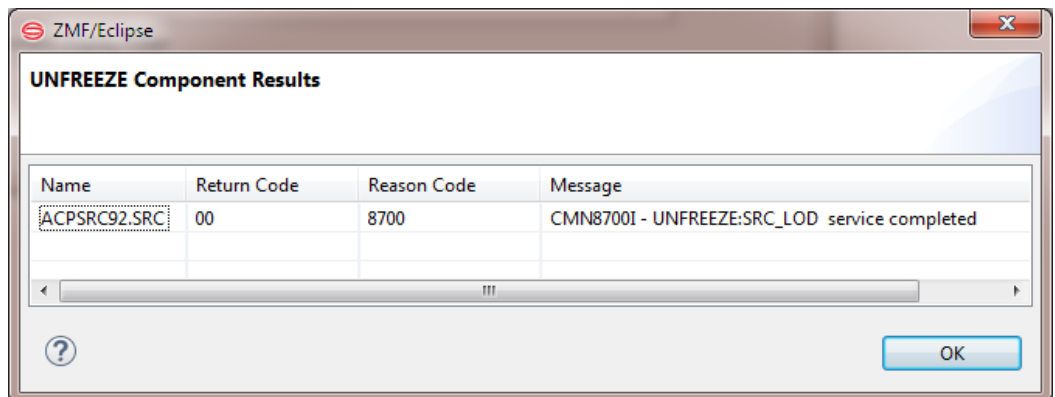
Unfreeze Components Dialog

The **ZMF Unfreeze Components** function works at the package level, but allows you to selectively unfreeze one or more components within that package. To unfreeze package components, perform the following steps:

- 1 Invoke the **Unfreeze Components** function from either the Serena or Java perspective. The **Unfreeze Components** dialog displays:



- 2 Click on a checkbox to select a component for unfreezing. Use the **Select All** and **Deselect All** buttons as desired.
- 3 Click **OK**. A dialog box lists the results for each component.



## Refreezing Package Components

Invoking the  
Refreeze  
Components  
Function  
Dialog

The **Refreeze Components** repository function is invoked from the following menus:

- **Serena perspective** — In the **Serena Explorer** view, expand the **ZMF Applications** node, then expand the node for the application to which the frozen package belongs. Right-click on the name of the frozen package. When the contextual menu displays, select the **Refreeze Components** option.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

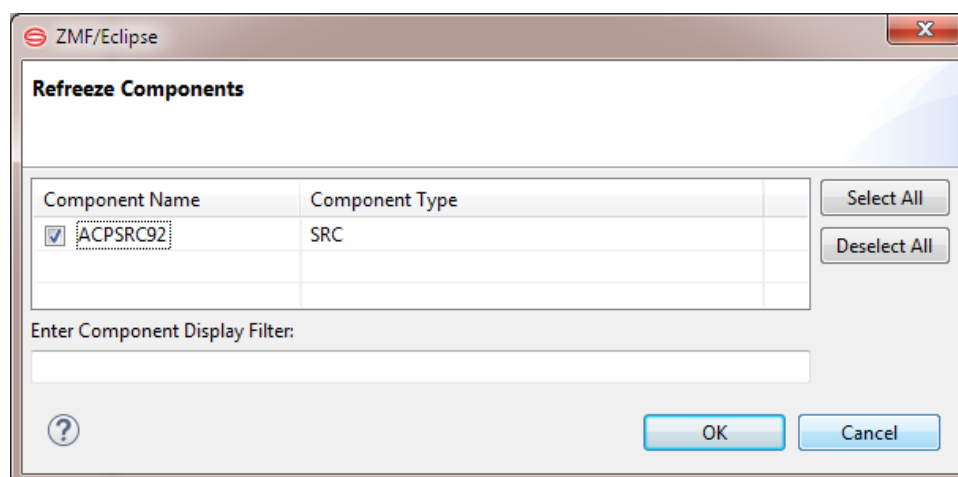
- **Java perspective** — In the **Package Explorer** view, right-click on a project that you have shared with the frozen ZMF repository package. When the contextual menu displays, open the **Team** submenu, then select the **ZMF Refreeze Components** option.

See [Chapter 3, "Project-Level ZMF Functions" on page 102](#) for more information.

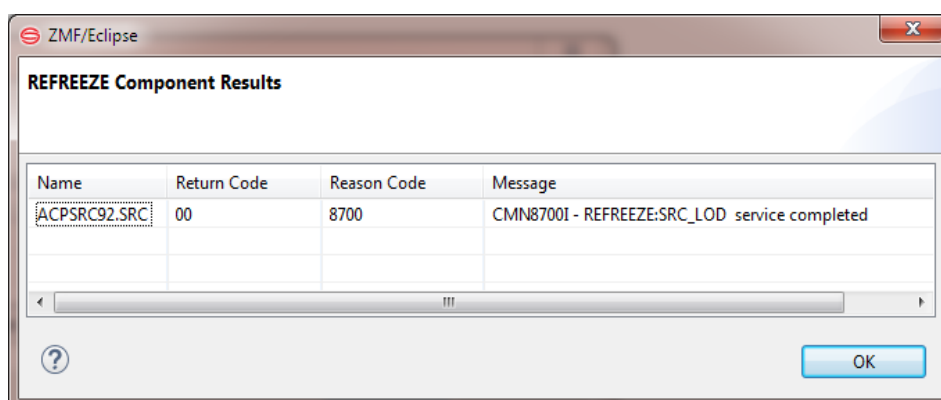
Refreeze  
Components  
Dialog

The **Refreeze Components** function works at the package level, but allows you to selectively unfreeze one or more components within that package. To refreeze package components, perform the following steps:

- 1 Invoke the **Refreeze Components** function from either the Serena or Java perspective. The **Refreeze Components** dialog displays:



- 2 Click on a checkbox to select a component for refreezing.  
Use the **Select All** and **Deselect All** buttons to select and deselect all of the components with one click.
- 3 Click **OK**. A dialog box lists the results.



## Promoting a Package

### Functional Description

The **Promote** function copies components from package staging libraries into previously defined promotion libraries. Promotion can be configured to execute additional processes to prepare promoted components for execution.

The workbench supports *full* promotion of all package components as a group, or *selective* promotion of one or more individual components within the package. The usual promotion rules defined for the application by your administrator remain in force — for example, you may be required to freeze a package before you can promote it.

## Invoking Promotion and Viewing Results

### Invoking the Promote Function

The **Promote** function is invoked from the following workbench menus:

- **Serena perspective** — In the **Serena Explorer** view, expand the **ZMF Applications** node, then expand the **Packages** node below it. Right-click on the name of the package to be promoted. When the contextual menu displays, select the **Promote** option near the top of the menu.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

- **Java perspective** — In the **Package Explorer** view, right-click on a project that you have shared with a ZMF change package. When the contextual menu displays, open the **Team** submenu, then select the **ZMF Promote** option.

See [Chapter 3, "Project-Level ZMF Functions" on page 102](#) for more information.

### Viewing Promote Job Status

Successful submission of the **Promote** job to ZMF does not guarantee the job will complete successfully. Job status can be viewed from the workbench as follows:

- **Serena perspective** — In the **Serena Explorer** view, find the node for the system or site where the relevant ZMF repository resides. Expand the **z/OS Jobs** node under that repository. Then expand the node for the job filter associated with your user ID and/or the application to which the promoted package belongs. The promotion job will be listed by job name and the output can be viewed in a workbench editor.

See [Chapter 2, "Working with z/OS Jobs" on page 55](#) for more information.

- **Remote System Explorer perspective (RDz only)** — In the **Remote Systems** navigation view (at right in the default layout), find the node for the z/OS system where the relevant ZMF repository resides. Expand the **JES** node under the system name, then expand the **My Jobs** node to see a list of job filters. Expand the node for the job filter associated with your user ID and/or the application to which the promoted package belongs. The promotion job will be listed by job name and the output can be viewed in a workbench editor.

See [Chapter 4, "z/OS Projects Perspective Overview" on page 110](#) for more information.

- **z/OS Projects perspective (RDz only)** — In the **Remote Systems** navigation view (at left in the default layout), find the node for the z/OS system where the relevant ZMF repository resides. Expand the **JES** node under the system name, then expand the **My Jobs** node to see a list of job filters. Expand the node for the job filter associated with your user ID and/or the application to which the promoted package belongs. The promotion job will be listed by job name and the output can be viewed in a workbench editor.

See [Chapter 4, "z/OS Projects Perspective Overview" on page 110](#) for more information.

## Procedure for Promoting a Package

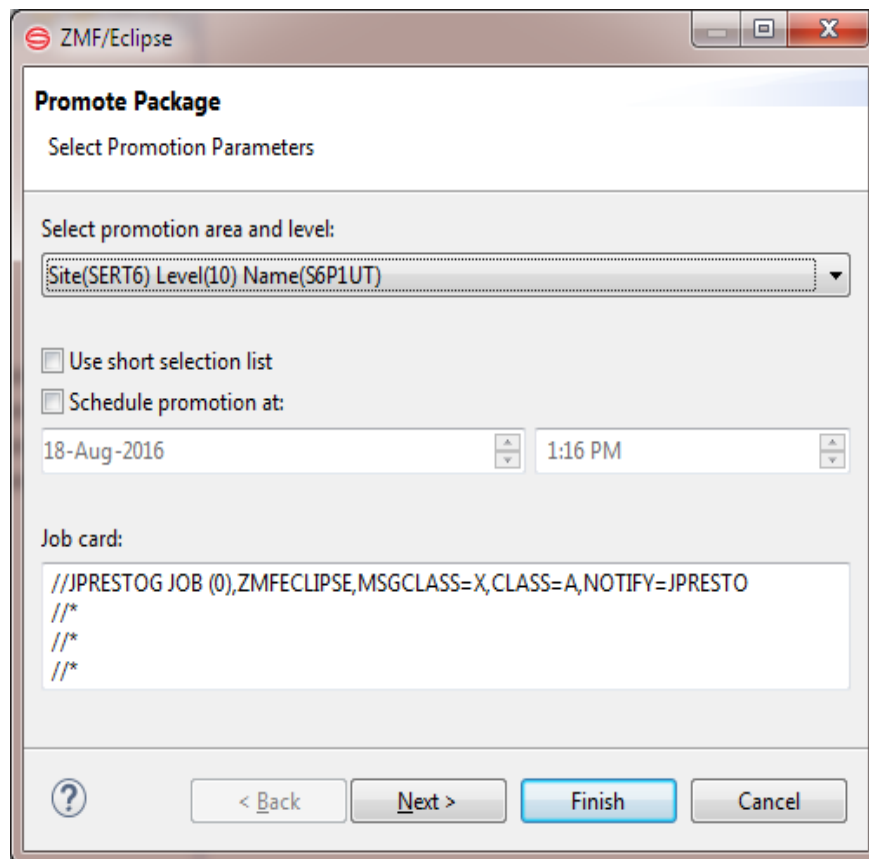
The **Promote** wizard displays up to four windows, one of which is optional. The steps for promoting a package using the wizard are summarized step-by-step here. Click on a blue hyperlink to see a detailed description of each window.



- 1 Invoke the **Promote** function from the **Serena Explorer** view of the Serena perspective.
- 2 The **Promote Package - Select Promotion Parameters** window is the first to display. Use this screen to:
  - a Identify the target promotion site and level.
  - b Schedule a date and time to promote the package.
  - c Edit the JCL for the promotion job card as needed.
  - d If you plan to perform a selective promotion, choose here whether or not to filter the component list that will be displayed on the next screen.
  - e Click **Next**.
- 3 When the **Promote Package - Select Promotion Type** window displays, choose between a full or a selective promotion.
- 4 A list of package components is shown on the **Select Promotion Type** window immediately below the promotion type selection buttons. All package components are shown by default. However, if you checked the option to filter the component list on the first wizard window (see [Step 2](#)), only components that have not previously been copied into the target promotion area are shown.
  - a If you chose selective promotion in [Step 3](#), select or deselect the components to promote using the checkboxes in the component list.
  - b Click **Next**.
- 5 Optionally, the **Promote Package - Specify Promotion User Variables** window displays customer-defined package variables for editing. This window displays only if user-defined package variables have been set up in ZMF and enabled for remote editing in the PROMOTE member of the ZDDOPTS parameter library.
- 6 The **Promote Package - Review Promotion Overlays** window is the final window displayed by the promotion wizard. It displays all components in the target promotion library that will be overlaid by components in the change package if you proceed with the promotion.
  - If the components to be overlaid are those you expect, click **Finish** to generate the promotion job and submit it for execution.
  - Otherwise, click **Back** to make corrections or **Cancel** to exit the promotion wizard.

## Promotion Parameter Window

The **Promote Package - Select Promotion Parameters** window is the first to be displayed by the **Promote Package** wizard.



### Selecting a Target Promotion Site and Level

Select the desired target promotion site and level from the pull-down menu. The promotion areas listed in the menu are those defined for the application in ZMF.

Valid promotion targets for a package depend on the particular promotion rule defined for the application to which the package belongs. They also vary with the current promotion status of the package. In general:

- The promotion level for a package component need not match the promotion level for the package as a whole. For example, a full promote of the entire package to level 10 may be performed successfully, but then a component may be selectively demoted to level 0 (zero), unfrozen, edited, staged back into the package, audited, refrozen, and selectively promoted back to promotion level 10.
- Before the first package promotion (that is, when the package promotion level is zero), individual components can be selectively promoted to a level higher than that of the package. This can be useful for unit testing of components in specific execution environments.
- After the first package promotion, components cannot be selectively promoted to a level higher than that of the package. At this stage, selective promotion is generally used after a selective demote to return a component to the package promotion level.

More information on promotion sites, levels, rules, and paths can be found in the *ChangeMan ZMF User's Guide*.

### **Scheduling a Promotion Date and Time**

Select the desired date and time when you want to schedule the promotion from the drop-down lists.

### **Short Selection List**

This checkbox controls the contents of the component list to be displayed on the next wizard screen.

- *Check this box* to filter the component list. Only components that have not already been promoted to the target promotion level in their current form will be displayed. The displayed components will include newly activated components and components that have been restaged after an earlier version was promoted.
- *Leave unchecked* to display all package components.

### **Job Card**

The **Job Card** field at the bottom of the window displays the JCL JOB statement that will be used to run the **Promote** job on the mainframe. Modify it as necessary.

By default, your TSO user ID is appended with an alphabetic character to create the job name. Each time this dialog displays, the last character of the default job name is incremented alphabetically.

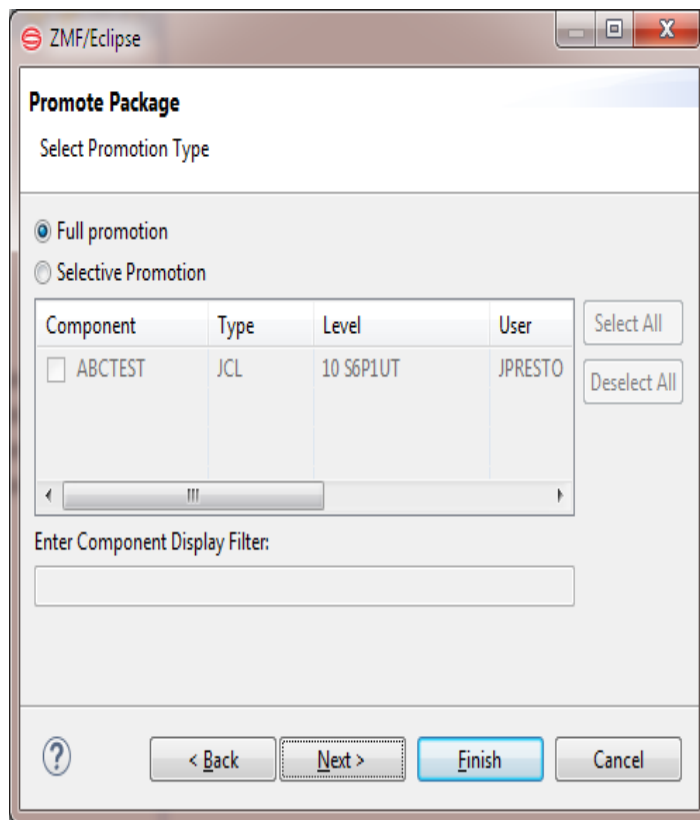
If you change the displayed values, your changes will be saved and used as defaults the next time a job card is displayed.

The following syntax conventions apply:

- A maximum of four lines are allowed.
- Lines must begin with double slashes and may not exceed 71 characters in length.
- An asterisk in position three marks a comment.
- A blank in position three continues the JCL command from the previous line. The content of a continuation line must begin in positions 4 to 16. Parameters may not be broken across continuation lines.

## **Promotion Type and Component Selection Window**

The **Promote Package - Select Promotion Type** window is the second window to be displayed by the promotion wizard.

**Promotion Type**

Select the type of promotion to perform:

- **Full promotion** (to promote the entire package as a whole)
- **Select promotion elements below** (to promote selected components only)

Valid promotion options depend on the current promotion status of the package. The most common situations are summarized in the table below.

Situation	Full Promote	Selective Promote
First promotion of this package at this site.	Yes	Yes
Package has previously been promoted to some non-zero level on this site. Promotion is desired to a new, higher level.	Yes	No
Package has previously been promoted to some non-zero level on this site. Target promotion level is the same as, or lower than, the promotion level of the package as a whole.	No	Yes

**Component Selection List**

The component selection list displays package components and their associated library types. The list may be complete or it may be filtered to show only those components eligible for promotion, depending on whether or not you checked the option to **Use short selection list** on the previous screen.

The checkboxes beside each component name can be selected only for a selective promotion — that is, only if you choose the option to **Select promotion elements below**. Use **Select All** and **Deselect All** to select or deselect components currently displayed in the list. Click the column name to sort the list. Click again to sort in the opposite order.

Check the box beside each components that you wish to promote, then click **Next**.



**NOTE** If the message "No components to promote" shows in title pane of this window, all eligible package components have already been promoted to the target promotion level. Choose **Full Promotion** to promote the package or click **Cancel** to exit the wizard.

### Enter Component Display Filter

For selective promotions, you can use a pattern-matching filter so that only a subset of the components appears in the selection list. For example, to see only the load components that have a type of LOD, you could enter \*.LOD in the **Enter Component Display Filter** field.

## User Variables Window

The **Promote Package - Specify Promotion User Variables** window displays customer-defined package variables for editing. This window displays only if user-defined package variables have been set up in ZMF and enabled for remote editing in the PROMOTE member of the ZDDOPTS parameter library.

Variable Name	Variable Value	Variable Help
Priority	5	
User ID	JPRESTO	
Log dsn		

Select variable in table to update value; Shift+Up / Shift+Down to navigate

Name:

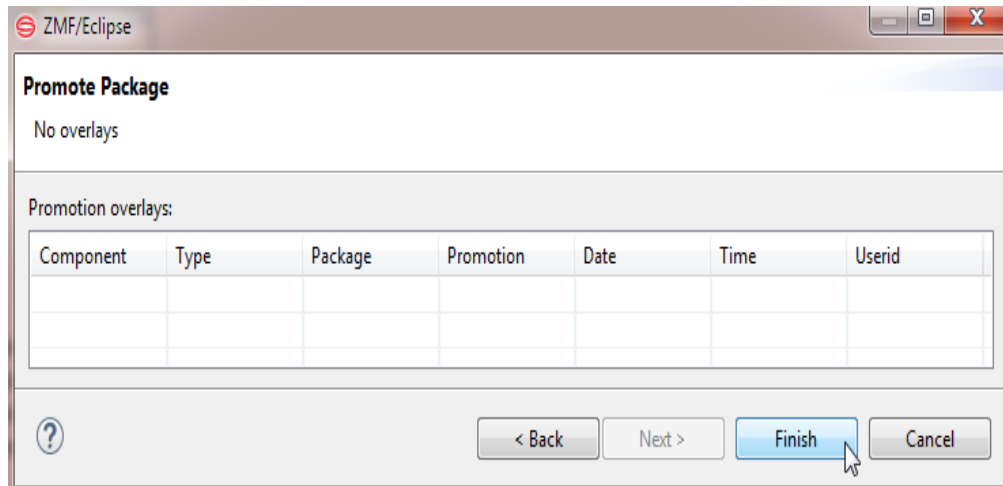
Value:

? < Back Next > Finish Cancel

- Update the variables as needed, following the validation rules specified in the PROMOTE member of ZDDOPTS.
- Click **Next**.

## Promotion Overlays Window

The final window displayed by the **Promote Package** wizard is the **Promote Package** window. It displays all components in the target promotion library that will be overlaid by components in the change package if you proceed with the promotion. If there are no overlays it will advise **No overlays**.

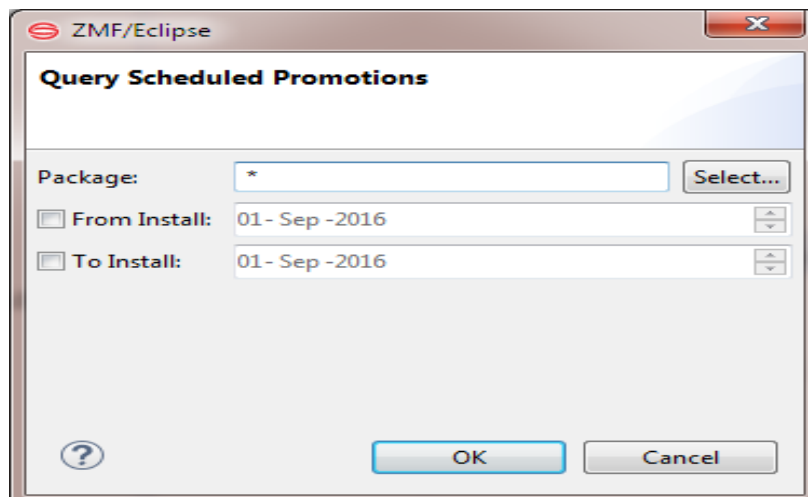


- If the components to be overlaid are those you expect, click **Finish** to submit the promotion job.
- Otherwise, click **Back** to make corrections or **Cancel** to exit the promotion wizard.

## Displaying the Scheduled Package Promotions View

Take the following actions to display a list of scheduled promotions in the **Scheduled Package Promotions** view:

- On the Window Menu item, select Show View, Other and select Scheduled Package Promotions. A new tab will open, titled **Scheduled Package Promotions**.
- Right-click on the area within the tab/view and select refresh. The **Query Scheduled Promotions** dialog appears.



- Specify or select a package name in the **Package** field. You can specify a wildcard for the package name to select all packages whose names match the wildcard specification.
- Check the **From Install** box and specify a From Install date to request the scheduled package promotions from that date forward.
- Check the **To Install** box and specify a To Install date to request the scheduled package promotions up to and including the specified date.

Specify both a From Install and To Install date to request the scheduled package promotions within the From/To time frame.

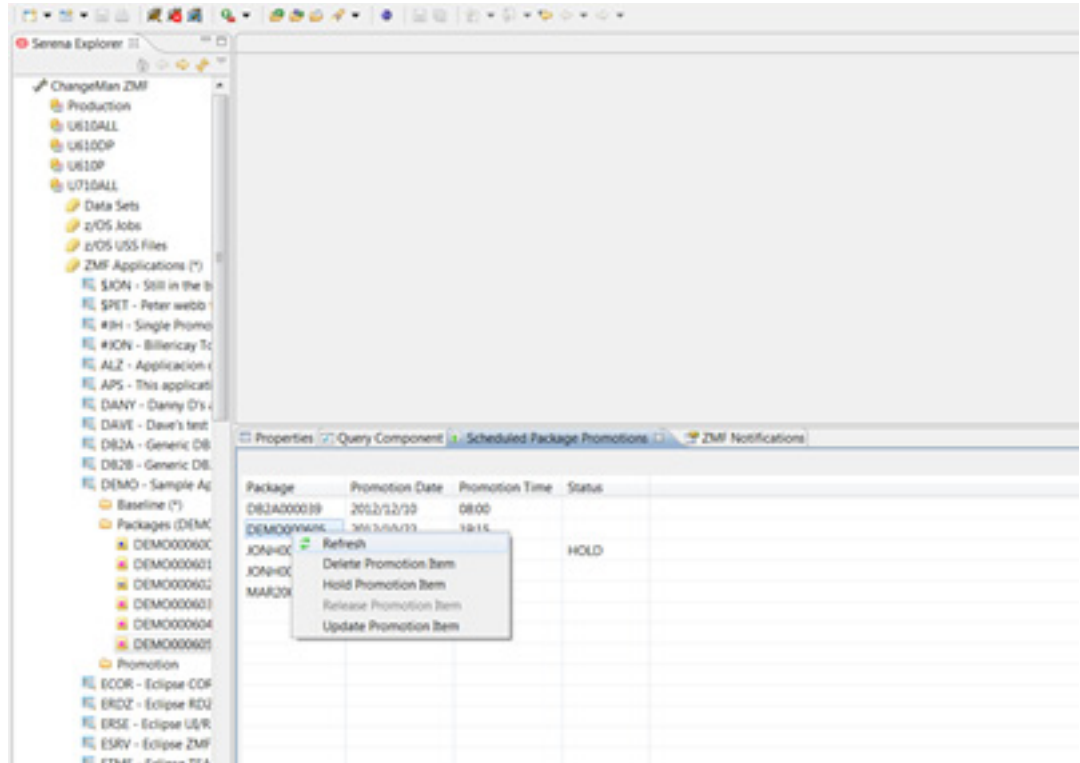
- Click OK.

All scheduled package promotions that match the criteria that you specify on the **Query Scheduled Promotions** dialog are displayed in the **Scheduled Package Promotions** view.

## Deleting, Holding, Releasing, or Updating a Scheduled Package Promotion

Take the following actions to refresh the Scheduled Package Promotions view or delete, hold, release, or update a scheduled package promotion:

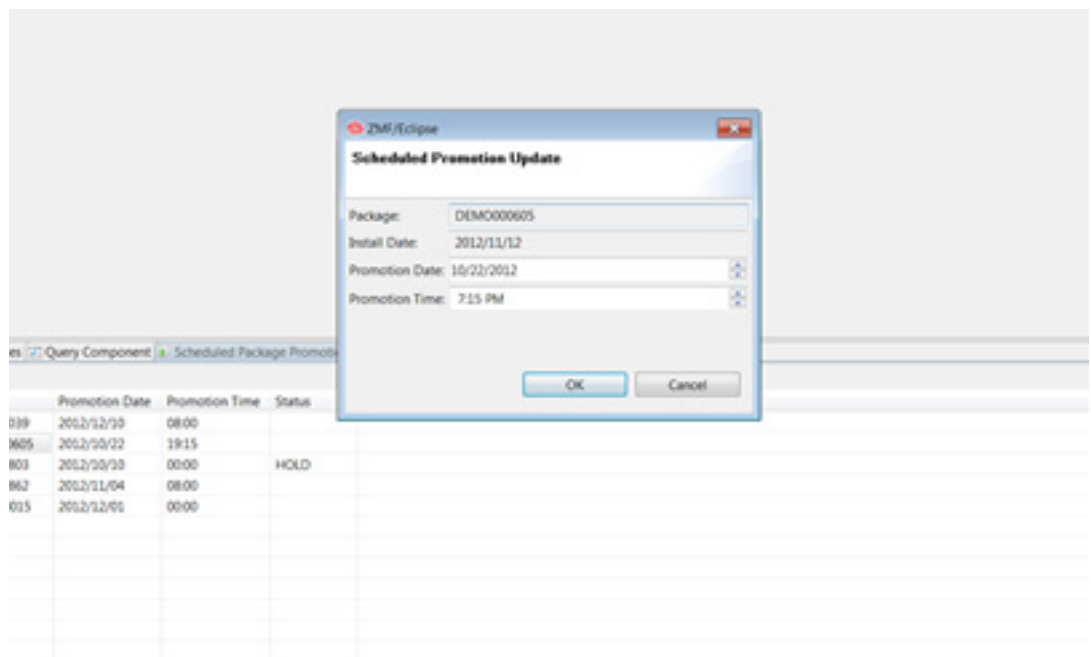
- Right-click on a package that is listed in the Scheduled Package Promotions view. An action pop-up menu is displayed:



- Select the desired action (Refresh, Delete Promotion, Hold Promotion, Release Promotions, or Update Promotion) from the pop-up menu.

### **Updating a Scheduled Package Promotion**

If you select the **Update Promotion Item** from the preceding action list, the Scheduled Promotion Update dialog appears:





- Specify the desired update to the Promotion Date and/or Promotion Time on this dialog and click OK.

## Demoting a Package

### Functional Description

The **Demote** function deletes some or all package components from execution environments that were previously populated by promotion. The promotion level assigned to the demoted package or component is level zero.

The workbench supports *full* demotion of all package components as a group, or *selective* demotion of one or more individual components within the package. The usual promotion rules defined for the application by your administrator remain in force — for example, you may be allowed to selectively demote a component in order to unfreeze and edit it, or you may be required to perform a full demote of the entire package in order to change any part of it.

### Invoking Demotion and Viewing Results

Invoking the  
Demote Function

The **Demote** function is invoked from the following workbench menus:

- **Serena perspective** — In the **Serena Explorer** view, expand the **ZMF Applications** node, then expand the **Packages** node below it. Right-click on the name of the package to be demoted. When the contextual menu displays, select the **Demote** option near the top of the menu.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

- **Java perspective** — In the **Package Explorer** view, right-click on a project that you have shared with a ZMF change package. When the contextual menu displays, open the **Team** submenu, then select the **ZMF Demote** option.

See [Chapter 3, "Project-Level ZMF Functions" on page 102](#) for more information.

Viewing Demote  
Job Status

Successful submission of the **Demote** job to ZMF does not guarantee the job will complete successfully. Job status can be viewed from the workbench as follows:

- **Serena perspective** — In the **Serena Explorer** view, find the node for the system or site where the relevant ZMF repository resides. Expand the **z/OS Jobs** node under that repository. Then expand the node for the job filter associated with your user ID and/or the application to which the demoted package belongs. The demotion job will be listed by job name and the output can be viewed in a workbench editor.

See [Chapter 2, "Working with z/OS Jobs" on page 55](#) for more information.

- **Remote System Explorer perspective (RDz only)** — In the **Remote Systems** navigation view (at right in the default layout), find the node for the z/OS system where the relevant ZMF repository resides. Expand the **JES** node under the system name, then expand the **My Jobs** node to see a list of job filters. Expand the node for the job filter associated with your user ID and/or the application to which the demoted package belongs. The demotion job will be listed by job name and the output can be viewed in a workbench editor.

See [Chapter 4, "z/OS Projects Perspective Overview" on page 110](#) for more information.

- **z/OS Projects perspective (RDz only)** — In the **Remote Systems** navigation view (at left in the default layout), find the node for the z/OS system where the relevant ZMF repository resides. Expand the **JES** node under the system name, then expand the **My Jobs** node to see a list of job filters. Expand the node for the job filter associated with your user ID and/or the application to which the demoted package belongs. The demotion job will be listed by job name and the output can be viewed in a workbench editor.

See [Chapter 4, "z/OS Projects Perspective Overview" on page 110](#) for more information.

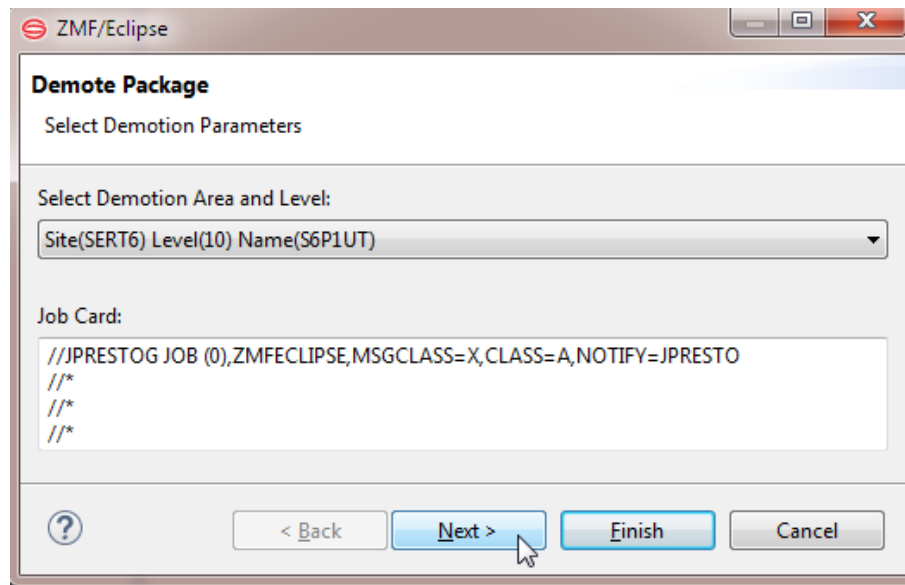
## Procedure for Demoting a Package

The **Demote Package** wizard displays up to three windows that are substantially identical to equivalent windows shown by the **Promote Package** wizard. The steps for demoting a package using the wizard are summarized step-by-step here.

- 1 Invoke the **Demote** function from the **Serena Explorer** view of the Serena perspective.
- 2 The **Demote Package - Select Demotion Parameters** window is the first to display. Use this screen to:
  - a Identify the target demotion site and level.
  - b Edit the JCL for the demotion job card as needed.
  - c Click **Next**.
- 3 When the **Demote Package - Select Demotion Type** window displays, choose between a full or a selective promotion.
- 4 A list of package components is shown on the **Select Demotion Type** window immediately below the promotion type selection buttons. All promoted package components are shown.
  - a If you chose selective demotion in [Step 3](#), select or deselect the components to demote using the checkboxes in the component list.
  - b Click **Next**.
- 5 Optionally, the **Demote Package - Specify Demotion User Variables** window displays customer-defined package variables for editing. This window displays only if user-defined package variables have been set up in ZMF and enabled for remote editing in the DEMOTE member of the ZDDOPTS parameter library.
- 6 Click **Finish** to generate the promotion job and submit it for execution.  
Otherwise, click **Back** to make corrections or **Cancel** to exit the demotion wizard.

## Demotion Parameters Window

The **Demote Package - Select Demotion Parameters** window is the first to be displayed by the **Demote Package** wizard.



### Selecting a Demotion Site and Level

The **Select Demotion Area and Level** field identifies the *promotion* area that a package or component is being demoted *from*. Some or all package components will be deleted from this execution environment.

Select the desired area from the pull-down menu. The available list of promotion areas are those defined for the application in ZMF.

More information on promotion sites, levels, rules, and paths can be found in the *ChangeMan ZMF User's Guide*.

### Job Card

The **Job Card** field at the bottom of the window displays the JCL JOB statement that will be used to run the **Demote** job on the mainframe. Modify it as necessary.

By default, your TSO user ID is appended with an alphabetic character to create the job name. Each time this dialog displays, the last character of the default job name is incremented alphabetically.

If you change the displayed values, your changes will be saved and used as defaults the next time a job card is displayed.

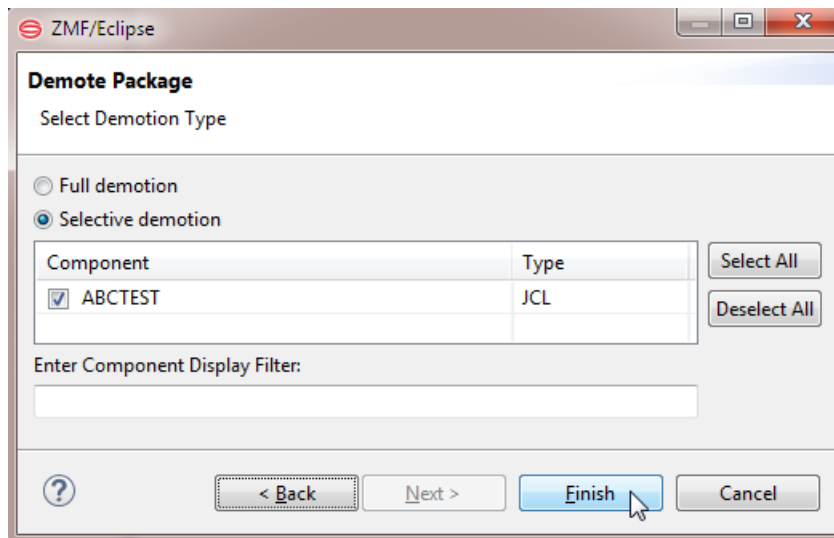
The following syntax conventions apply:

- A maximum of four lines are allowed.
- Lines must begin with double slashes and may not exceed 71 characters in length.
- An asterisk in position three marks a comment.
- A blank in position three continues the JCL command from the previous line. The content of a continuation line must begin in positions 4 to 16. Parameters may not be broken across continuation lines.

## Demotion Type and Component Selection Window

The **Demote Package - Select Demotion Type** window is the second window to be displayed by the promotion wizard. For example:

### Demotion Type



Select the type of demotion to perform:

- **Full Demotion** (to promote the entire package as a whole)
- **Select Demotion Elements Below** (to promote selected components only)

Valid demotion options depend on the current promotion status of the package. The most common situations are summarized in the table below.

Situation	Full Demote	Selective Demote
First promotion of this package at this site.	No	No
Package has previously been promoted to some non-zero level on this site. Demotion is desired to remove the package or some of its components from the site.	Yes	Yes

### Component Selection List

The component selection list displays package components and their associated library types. The checkboxes beside each component name can be selected only for a selective promotion — that is, only if you choose the option to **Select Demotion Elements Below**. Use **Select All** and **Deselect All** to select or deselect components currently displayed in the list. Click the column name to sort the list. Click again to sort in the opposite order.

Check the box beside each components that you wish to demote, then click **Next**.



**NOTE** If the message "No components to demote" shows in title pane of this window, all eligible package components have already been demoted from the promotion level. Choose **Full Demotion** to demote the package, or click **Cancel** to exit the wizard.

## Enter Component Display Filter

For selective demotions, you can use a pattern-matching filter so that only a subset of the components appears in the selection list. For example, to see only the load components that have a type of LOD, you could enter \*.LOD in the **Enter Component Display Filter** field.

## User Variables Window

The **Demote Package - Specify Demotion User Variables** window presents customer-defined package variables for editing. This window displays only if user-defined package variables have been set up in ZMF and enabled for remote editing in the DEMOTE member of the ZDDOPTS parameter library.

Variable Name	Variable Value	Variable Help
Priority	5	
User ID	JPRESTO	
Log dsn		

Select variable in table to update value; Shift+Up / Shift+Down to navigate

Name: Priority

Value: 5

- Update variable values as needed, following the validation rules specified in the DEMOTE member of ZDDOPTS.
- Click **Finish** to initiate the demotion job.

# Auditing a Package

## Functional Description

The ChangeMan ZMF audit function validates the synchronization between changed components in a change package and any related components in:

- The staging and baseline libraries of those components.
- The staging and baseline libraries of components in other participating packages.

- The staging and baseline libraries of components in other applications.

Related components are tracked by ZMF in the impact analysis table. This table identifies relationships between source programs and copybooks, between source programs and load modules, and between statically link-edited load modules and the modules that call them. Many other relationships are also tracked. Audit ensures that the set of changed components in a package invoke modules that actually exist and are mutually consistent.

Refer to the *ChangeMan ZMF User's Guide* for more information about the audit function.

## Invoking Audit and Viewing the Audit Report

Invoking the  
Audit Function

The **Audit** function for a package in the ChangeMan ZMF repository is invoked from the following workbench menus:

- **Serena perspective** — In the **Serena Explorer** view, expand the **ZMF Applications** node, then expand the **Packages** node below it. Right-click on the name of the package to be audited. When the contextual menu displays, select the **Audit** menu option.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

- **Java perspective** — In the **Package Explorer** view, right-click on a project that you have shared with a ZMF change package. When the contextual menu displays, open the **Team** submenu, then select the **ZMF Package Audit** option.

See [Chapter 3, "Project-Level ZMF Functions" on page 102](#) for more information.

A package must be in development (DEV) or frozen (FRZ) status to be eligible for audit.

Viewing the  
Audit Report

The resulting audit report can be viewed in the workbench from the following menus:

- **Serena perspective** — In the **Serena Explorer** view, find the node for the system or site where the relevant ZMF repository resides. Expand the **z/OS Jobs** node under that repository. Then expand the node for the job filter associated with your user ID and/or the application to which the audited package belongs. The audit job will be listed by job name and the output can be viewed in a workbench editor.

See [Chapter 2, "Working with z/OS Jobs" on page 55](#) for more information.

- **Remote System Explorer perspective (RDz only)** — In the **Remote Systems** navigation view (at right in default layout), find the node for the z/OS system where the relevant ZMF repository resides. Expand the **JES** node under the system name, then expand the **My Jobs** node to see a list of job filters. Expand the node for the job filter associated with your user ID and/or the application to which the audited package belongs. The audit job will be listed by job name and the output can be viewed in a workbench editor.

See [Chapter 4, "z/OS Projects Perspective Overview" on page 110](#) for more information.

- **z/OS Projects perspective (RDz only)** — In the **Remote Systems** navigation view (at left in default layout), find the node for the z/OS system where the relevant ZMF repository resides. Expand the **JES** node under the system name, then expand the **My Jobs** node to see a list of job filters. Expand the node for the job filter associated with your user ID and/or the application to which the audited package belongs. The audit job will be listed by job name and the output can be viewed in a workbench editor.

See [Chapter 4, "z/OS Projects Perspective Overview" on page 110](#) for more information.

## Preparing and Submitting the Audit Job

To audit a ZMF change package, perform the following steps.

- 1 Invoke the **Audit** function from a package contextual menu in **Serena Explorer** or from the **Team** submenu of a project contextual menu in **Package Explorer**.
- 2 When the **Audit Package - Specify Audit Parameters** wizard screen displays, select any desired **General Options** that apply to all package levels and types. (See ["Specify Audit Parameters" on page 216.](#))
- 3 If auditing a participating package, also select any desired options from the **Participating Packages** section of the **Specify Audit Parameters** screen. (See ["Specify Audit Parameters" on page 216.](#))
- 4 If this is a **cross-application audit**, also do the following:
- 5 Choose a report header format for cross-application audit reporting. (See ["Specify Audit Parameters" on page 216.](#))
  - a Choose which supplemental applications to include in the audit by clicking the **Select Applications** button on the **Specify Audit Parameters** screen. This will pop up the **Select Applications** wizard screen. (See ["Select Applications for Audit" on page 220.](#))
- 6 Edit the default JCL of the audit **job card** as needed on the **Specify Audit Parameters** screen. (See ["Specify Audit Parameters" on page 216.](#))
- 7 If the **Next** button is enabled, user-defined variables have been specified as editable in the AUDIT member of ZDDOPTS. Click the **Next** button to display the **Audit Package - Variables** wizard screen and provide the required values. (See ["Audit Package - User Variables" on page 221.](#))
- 8 Click the **Finish** button to submit the audit job.

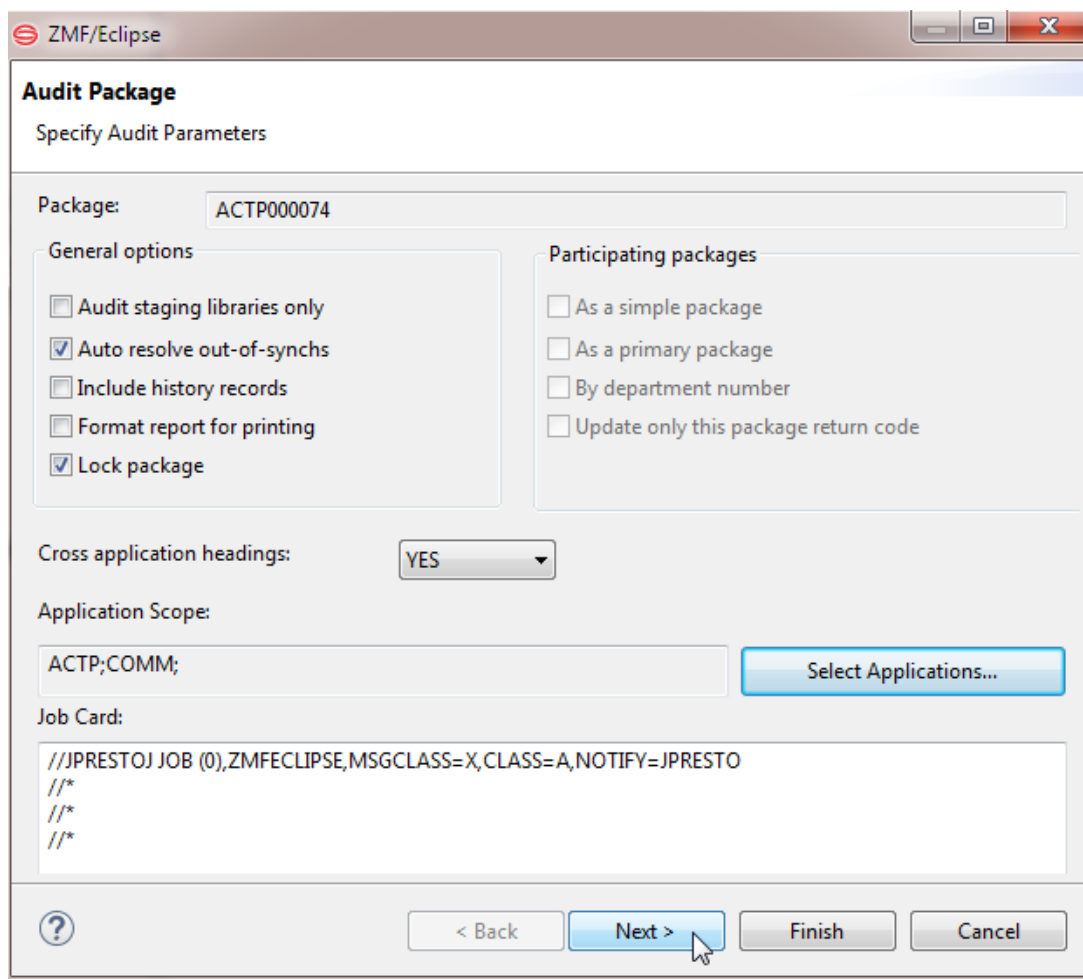
## Audit Wizard Windows

The following wizard windows are displayed for the **Audit** function. Click on the window title to see detailed data entry instructions for that window.

Window	Description
<b>Audit Package - Specify Audit Parameters</b>	Always displayed first. Allows you to select parameters that determine the focus and scope of the audit.
<b>Select Applications</b>	Displayed only if you click the <b>Select Applications</b> button on the <b>Specify Audit Parameters</b> wizard screen. Allows you to select additional applications to be included in the audit synchronization check.
<b>Audit Package - Specify User Variables</b>	Displayed only if user-defined package variables are enabled in the AUDIT member of ZDDOPTS. Allows you to update the values of these variables for the audit.

## Specify Audit Parameters

When you invoke the **Audit** function, the **Audit Package - Specify Audit Parameters** screen displays. Values default to those entered the last time you used this screen.



**Audit Package**  
Specify Audit Parameters

Package: ACTP000074

**General options**

- Audit staging libraries only
- Auto resolve out-of-synchs
- Include history records
- Format report for printing
- Lock package

**Participating packages**

- As a simple package
- As a primary package
- By department number
- Update only this package return code

Cross application headings: YES

Application Scope: ACTP;COMM; Select Applications...

Job Card:  

```
//JPRESTOJ JOB (0),ZMFECCLIPSE,MSGCLASS=X,CLASS=A,NOTIFY=JPRESTO
//*
//*
//*
```

? < Back Next > Finish Cancel

### General Options

The **General Options** section provides audit scope controls and reporting options that apply to all package types and levels. These are always enabled for selection.

Click on a checkbox to select the adjacent option. Click it again to deselect it. Any combination of options in the following table may be chosen.

Option	Description
<b>Audit staging libraries only</b>	<p>Optionally limits the scope of the audit to the package's staging libraries only.</p> <ul style="list-style-type: none"> <li>■ <i>Check this box</i> to audit only staging libraries for this package.</li> <li>■ <i>Leave unchecked</i> to perform a normal package audit that includes staging and baseline libraries.</li> </ul>



Option	Description
<b>Auto resolve out-of-synchs</b>	<p>Certain out-of-synch conditions involving copybooks and load modules can generally be resolved automatically, without human review. These include ZMF SYNCH2, SYNCH4, SYNCH5, SYNCH7, SYNCH8, SYNCH9, SYNCH15, and SYNCH16 conditions. (Refer to the <i>ChangeMan ZMF User's Guide</i> for details.)</p> <ul style="list-style-type: none"> <li>■ <i>Check this box</i> to report the above out-of-synch conditions and automatically submit recompile/relink jobs to resolve them.</li> <li>■ <i>Leave unchecked</i> to report out-of-synch conditions and leave them unresolved pending human review.</li> </ul>
<b>Include history records</b>	<p>ZMF maintains a historical record of component changes. History records identify all packages in which a component was changed.</p> <ul style="list-style-type: none"> <li>■ <i>Check this box</i> to extract component history for each staged component and include it in the audit report.</li> <li>■ <i>Leave unchecked</i> to omit component history information from the audit report.</li> </ul>
<b>Format report for printing</b>	<p>By default, the audit report is formatted for online browsing. However, you can optionally format the audit report for printing.</p> <ul style="list-style-type: none"> <li>■ <i>Check this box</i> to format the audit report for printing. Printer control characters are included to manage report appearance.</li> <li>■ <i>Leave unchecked</i> to browse the report online.</li> </ul>
<b>Lock package</b>	<p>Locks all package components to prevent changes while the audit is running. Package lock is preferred on the final audit run prior to approval and installation. However, it is often not desirable to halt all work on a package during early audits.</p> <ul style="list-style-type: none"> <li>■ <i>Check this box</i> to lock the package and prevent component changes during audit.</li> <li>■ <i>Leave unchecked</i> to allow component changes during audit.</li> </ul>

### Participating Packages

When auditing a participating package, the options in the **Participating Packages** section are enabled for selection on the **Select Audit Parameters** screen. These options are described in the following table. Some options are mutually incompatible.

Option	Description
<b>As a simple package</b>	<p>Limits the scope of the audit to the staging and baseline libraries that are normally audited for a simple package. The audit return code is updated for this package only. Useful for early auditing passes on participating packages.</p> <ul style="list-style-type: none"> <li>■ <i>Check the box</i> to audit the current package only.</li> <li>■ <i>Leave unchecked</i> to perform a normal participating package audit. The audit includes all relationships between this package and the other packages with which it participates in a complex or super package.</li> </ul> <p><b>NOTE:</b> If checked, no other <b>Participating Packages</b> options may be selected.</p>

Option	Description
<b>As a primary package</b>	<p>Audits the participating package with reference to all other packages with which it participates in a complex or super package, but limits the scope of reporting to this package. Useful in early audit passes when a participating package contains the primary functional components of a complex or super package.</p> <ul style="list-style-type: none"> <li>■ <i>Check this box</i> to audit the libraries of all participating packages in the same complex/super package as this one, but report out-of-sync conditions and update the audit return code only for the target participating package.</li> <li>■ <i>Leave unchecked</i> to perform a normal participating package audit. The audit includes all relationships between this package and the other packages with which it participates in a complex or super package. The audit return code will be updated for all packages sharing the same complex/super package as this one.</li> </ul> <p><b>NOTE:</b> This option may NOT be checked if you choose to audit <b>As a simple package</b> or <b>By department number</b>.</p>
<b>By department number</b>	<p>Limits the scope of the audit to only those participating packages that share the same department number as the target participating package. Useful for early audits when department codes are used to identify functions that must work together at all stages of development.</p> <ul style="list-style-type: none"> <li>■ <i>Check this box</i> to restrict the audit to participating packages that share the same department number.</li> <li>■ <i>Leave unchecked</i> to perform a normal participating package audit. The audit includes all relationships between this package and the other packages with which it participates in a complex or super package. The audit return code will be updated for all packages sharing the same complex/super package as this one.</li> </ul> <p><b>NOTE:</b> This option may NOT be checked if you choose to audit <b>As a simple package</b> or <b>As a primary package</b>.</p>
<b>Update only this package return code</b>	<p>Reports the full set of out-of-synch conditions for all participating packages associated with the target package though a complex/super package, but updates the audit return code for the target participating package only.</p> <ul style="list-style-type: none"> <li>■ <i>Check this box</i> to limit audit return code updates to the target participating package only.</li> <li>■ <i>Leave unchecked</i> to perform a normal participating package audit that updates the audit return code for all participating packages.</li> </ul> <p><b>NOTE:</b> This option may NOT be checked if you choose to audit <b>As a simple package</b>.</p>

## Cross Application Audit Options

For cross-application audits, two audit options are available. The first turns on cross-application report headings in the audit report. The second allows you to bring up the **Select Applications** wizard screen and shows the selections currently in effect.

Details are shown in the table below.

Option or Field	Description
<b>Cross-application headings</b>	<p>From the pull-down menu, select the desired option:</p> <ul style="list-style-type: none"> <li>■ YES - Turn on cross-application headings and print them at the bottom of the audit report (default). This option uses the most report space.</li> <li>■ NO - Turn off cross-application headings to save space.</li> <li>■ TOP - Turn on cross-application headings but move them to the top of the audit report.</li> </ul>
<b>Application scope</b>	<p>The <b>Application Scope</b> box displays the application directly associated with this package at creation. It also lists any <i>supplemental</i> applications you want to include in the audit.</p> <p>Supplemental applications are audited <i>in addition</i> to the applications already selected for audit. Those applications vary by package level and the audit options already selected, as follows:</p> <ul style="list-style-type: none"> <li>■ <i>Simple package</i> — The application directly associated with this package at creation is the only application audited.</li> <li>■ <i>Participating package</i> — Varies with audit option. <ul style="list-style-type: none"> <li><b>a</b> If <b>Audit as simple package</b> or <b>Audit as primary package</b> is selected, only the application directly associated with this package at creation is audited.</li> <li><b>b</b> Otherwise, all applications associated with all packages participating in the same complex or super package as this one are audited.</li> </ul> </li> <li>■ <i>Complex or super package</i> — All applications associated with all packages participating in this complex or super package are audited.</li> </ul>
<b>Select Applications</b>	<p>To add or remove supplemental applications to the audit, click the <b>Select Applications</b> button. This action brings up the <b>Select Applications</b> wizard screen. The supplemental applications checked on that screen are added to the application list in the <b>Application Scope</b> box.</p> <p>See "<a href="#">Select Applications for Audit</a>" on page 220 below for a description of the <b>Select Applications</b> wizard screen.</p>

## Edit Job Card

The **Job Card** field at the bottom of the window displays the JCL JOB statement that will be used to run the audit job on the mainframe. Modify it as necessary.

By default, your TSO user ID is appended with an alphabetic character to create the job name. Each time this dialog displays, the last character of the default job name is incremented alphabetically.

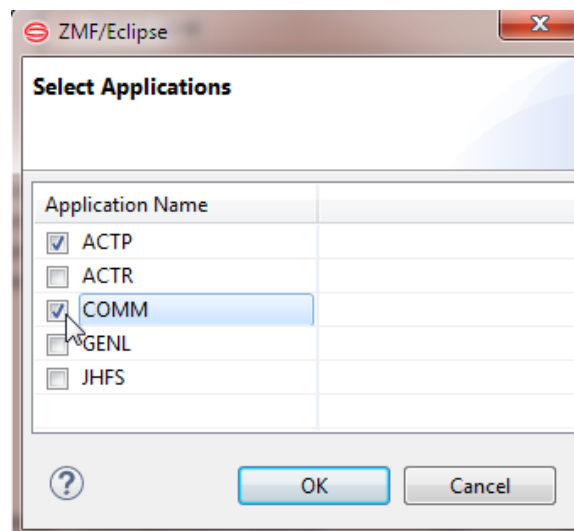
If you change the displayed values, your changes will be saved and used as defaults the next time a job card is displayed.

The following syntax conventions apply:

- A maximum of four lines are allowed.
- Lines must begin with double slashes and may not exceed 71 characters in length.
- An asterisk in position three marks a comment.
- A blank in position three continues the JCL command from the previous line. The content of a continuation line must begin in positions 4 to 16. Parameters may not be broken across continuation lines.

## Select Applications for Audit

The **Select Applications** screen enables you to add any supplemental applications you wish to those already included in the audit. This screen displays when you click the **Select Applications** button on the **Audit Package - Specify Audit Parameters** screen.



The applications listed are all those defined for the ChangeMan instance. However, they are filtered through your currently active application filter in ZMF for Eclipse.

For more information about filters, see [Chapter 2, "Filtering Views in Serena Explorer" on page 29](#).

### Adding Supplemental Applications to the Audit

- 1 Click the checkboxes to add any desired supplemental applications to the audit.
- 2 Uncheck any previously selected applications that you wish to omit from the audit. You may deselect any application except the one in which the package is defined.
- 3 Click **OK** to save the information and return to the **Audit Package** wizard screen.

## Audit Package - User Variables

The **Audit Package - User Variables** screen allows you to enter values for user-defined package variables at audit time. This screen is enabled only if user-defined variables have been specified as editable in the AUDIT member of the ZDDOPTS configuration library in ChangeMan ZMF. When the user variables screen is enabled, the **Next** button is active on the **Audit Package - Specify Audit Parameters** wizard screen.

Variable Name	Variable Value	Variable Help
Priority	5	
User ID	JPRESTO	
Log dsn		

Select variable in table to update value; Shift+Up / Shift+Down to navigate

Name: Priority  
Value: 5

< Back   Next >   Finish   Cancel

### Entering User Variables for the Audit

- 1 On the the **Audit Package - Specify Audit Parameters** wizard screen, click **Next** button to display the **Audit Package - Variables** wizard screen.
- 2 Enter or update the variables as needed for this audit, following the validation rules specified in the AUDIT member of ZDDOPTS.
- 3 Click **Finish** to submit the audit job for execution.

## Resetting Audit Lock

If a change package has previously been locked while performing an audit, the **Reset Audit Lock** function can clear the lock. Resetting the audit lock allows users to make changes to package components while a long-running audit job is executing.



**NOTE** Changes made during an audit run can cause out-of-sync conditions to occur. You should only use the **Reset Audit Lock** function during preliminary audits. The final audit job should lock out user changes when initiated, and users should not reset the lock.

Invoke the **Reset Audit Lock** function from the following menu:

- **Serena perspective** — In the Serena Explorer navigation view, expand the node for the ZMF server where the package of interest resides. Then expand the **z/OS Applications** node, the node for the particular application containing the package, and its **Packages** node. Right-click on the desired package to bring up its contextual menu, then select the **Reset Audit Lock** option.

If the **Reset Audit Lock** option is grayed out, either the package is not locked for audit or you are not authorized by ZMF to reset the lock.

## Approving or Rejecting a Package for Installation

### Functional Description

Before a change package can be installed into production, it must be approved for installation by a predefined hierarchy of approvers. The required approvers vary by application and by the type of change. For example, an emergency fix of temporary duration may use a different approval process than a planned, permanent, functional redesign. Approvers are notified automatically by ChangeMan ZMF when an approval action is required.

The **Approve** function in ZMF for Eclipse is enabled only if you are an authorized package approver and only after approval request notifications have been issued for a package.

Approval Actions Four possible approval actions may be taken by a package approver:

- Approve package installation as-is.
- Reject installation and request reversion to development (DEV) status for changes.
- Flag the package as under review and defer approval or rejection to a future date.
- Comment on the package without approving or rejecting it.

Once an approval action is performed, it cannot be changed in ZMF for Eclipse.

Refer to the *ChangeMan ZMF User's Guide* for more information about package approval.

### Invoking the Approval Function

The **Approve** function is invoked from the following menu:

- **Serena perspective** — In the **Serena Explorer** view, expand the **ZMF Applications** node, the node of the relevant application, and the **Packages** node below it. Right-click on the name of the package for which installation JCL should be rebuilt. When the contextual menu displays, select the **Approve** option.

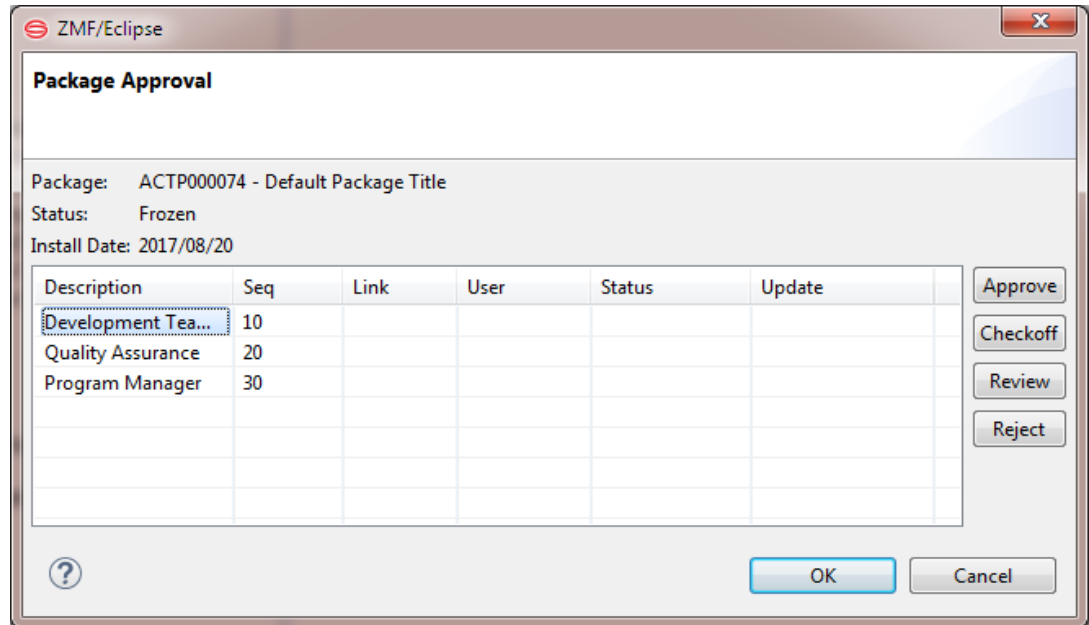
See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

The success or failure of the **Approve** action is reported in a dialog window.

### Approval Actions Step-by-Step

To approve a package, reject a package, or take other approval actions on a package, perform the following steps.

- 1 Invoke the **Approve** function from the contextual menu of the package you want to approve or reject.
- 2 The **Package Approval** window displays the package name and description, the package status, and a list of approval actions currently recorded for the package. The approver category in use for the package is shown in the **Description** column.



- a Select the description for the incomplete approval entry in the table. This enables the approval action buttons at right.
  - b Click one of the following buttons to register your approval action:
    - **Approve** — You approve the package for installation as-is. No further prompt window will display.
    - **Checkoff** (or **Comment**) — You neither approve nor reject the package, but merely "check off" the requirement to respond. You will be prompted to register comments about the package.
    - **Review** — You wish defer an approval decision to a later date while reviewing the package or researching a possible issue with it. No further prompt window will display.
    - **Reject** — You reject the installation of the package into production. You will be prompted to supply a reason for rejecting the package.
  - c Click **OK** to record your approval action or request the next prompt. No further action is required if you approved the package or flagged it for review.
- 3 If you clicked the **Checkoff** (or **Comment**) action button, the **Check Off List** window displays.

- a Type a comment about the package in the text box. At least one character is required.
  - b Click **OK** to record your comment. No further action is required for **Checkoff**.
- 4 If you clicked the **Reject** action button, the **Reject Package** window displays.

- a In the text box, type a reason for rejecting the package for installation in production. At least one character is required.
- b Click **OK** to record your rejection action. No further action is required.

## Rebuilding Installation JCL

### Functional Description

The JCL install jobs that move a change package into production must be built using z/OS file tailoring prior to install time. Your ZMF administrator configures the repository to perform this file tailoring automatically — either when the package is frozen or when the final package approval action is received. However, changes to a package may be required after the initial file tailoring is performed, and these changes may require a rebuild of the installation JCL. This may be done on demand using the **Rebuild Install JCL** function.



The following rules apply to the **Rebuild Install JCL** function:

- The package must be in frozen (FRZ) status.
- The scheduled install date must be in the future.
- An initial file tailoring of the installation JCL must already have been performed for the change package. Look for file tailoring program CMN\_ADSP in the job log (where the underscore represents a ZMF started task identifier) if the prior execution of file tailoring is in doubt.

Refer to the *ChangeMan ZMF User's Guide* for more information about file tailoring and rebuilding the JCL install jobs.

## Invoking the Job and Viewing Results

The **Rebuild Install JCL** function is invoked from the following menu:

- *Serena perspective* — In the **Serena Explorer** view, expand the **ZMF Applications** node, the node of the relevant application, and the **Packages** node below it. Right-click on the name of the package for which installation JCL should be rebuilt. When the contextual menu displays, select the **Rebuild Install JCL** option.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

The success or failure of the **Rebuild Install JCL** request is reported in a dialog window.

# Backing a Package Out of Production

## Functional Description

The **Backout** function performs two operations. First, it backs a change package out of all the production environments where it has been installed. Then, after all production environments have been restored to their pre-change states, the baseline libraries in the ZMF repository are "reverse rippled" to back the change package out of baseline.

The following rules apply to the **Backout** function:

- The package must be in baselined (BAS) or installed (INS) status.
- The staging libraries containing the change package components must still exist. If those libraries have been aged and physically scratched by housekeeping, a **Backout** is no longer possible.

Refer to the *ChangeMan ZMF User's Guide* for additional rules that apply to backing out a package.

## Invoking the Backout Function

The **Backout** function is invoked from the following workbench menus:

- *Serena perspective* — In the **Serena Explorer** view, expand the **ZMF Applications** node, the node of the relevant application, and the **Packages** node below it. Right-click on the name of the package to be backed out. When the contextual menu displays, select the **Backout** option.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

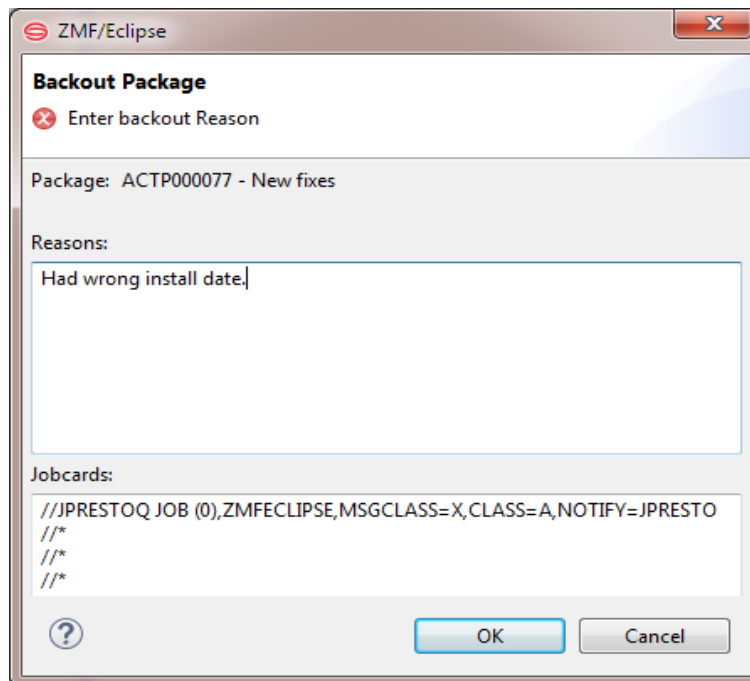
- **Java perspective** — In the **Package Explorer** navigation view, navigate to a project that is shared with a ZMF package. Right-click on the project to bring up its contextual menu, then open the **Team** submenu and select the **ZMF Package Backout** option.

See [Chapter 3, "Project-Level ZMF Functions" on page 102](#) for more information.

A dialog box reports the success or failure of the **Backout** operation.

## Backout Parameters Window

When you invoke the **Backout** function, the **Backout Package - Enter/Verify Backout Parameters** window displays.



Enter the following information:

- **Reasons** — Type a reason for backing out the package. You must type at least one character.
- **Jobcards** — Displays the JCL JOB statement that will be used to run the backout job on the mainframe. Modify it as necessary.

By default, your TSO user ID is appended with an alphabetic character to create the job name. Each time this dialog displays, the last character of the default job name is incremented alphabetically. If you change the displayed values, your changes will be saved and used as defaults the next time a job card is displayed.

The following syntax conventions apply:

- A maximum of four lines are allowed.
- Lines must begin with double slashes and may not exceed 71 characters in length.
- An asterisk in position three marks a comment.

- A blank in position three continues the JCL command from the previous line. The content of a continuation line must begin in positions 4 to 16. Parameters may not be broken across continuation lines.

Click **OK** to submit the **Backout** request.

## Reverting a Package to Development Status

### Functional Description

A package that has been frozen (FRZ), partially or fully approved (APR), or backed out of production (BAK) can be returned to development (DEV) status using the **Revert** function. **Revert** removes all previously entered approvals, unlocks package components and metadata, and opens the package to new development.

Refer to the *ChangeMan ZMF User's Guide* for the rules that apply to reverting a package.

### Invoking the Revert Function

The **Revert** function is invoked from the following menu:

- **Serena perspective** — In the **Serena Explorer** view, expand the **ZMF Applications** node, the node of the relevant application, and the **Packages** node below it. Right-click on the name of the package to be reverted. When the contextual menu displays, select the **Revert** option.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

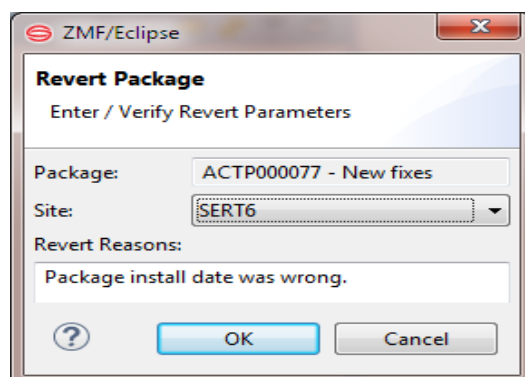
- **Java perspective** — In the **Package Explorer** navigation view, navigate to a project that is shared with a ZMF package. Right-click on the project to bring up its contextual menu, then open the **Team** submenu and select the **ZMF Package Revert** option.

See [Chapter 3, "Project-Level ZMF Functions" on page 102](#) for more information.

A dialog box reports the success or failure of the **Revert** operation.

### Revert Parameters Window

When you invoke the **Revert** function, the **Revert Package - Enter/Verify Revert Parameters** window.



Type a reason for reverting the package. You must type at least one character.

Click **OK** to submit the **Revert** request.

## Querying Packages

**Functional Description** The **Query Package** function lists all change packages in a ZMF repository that conform to your selection criteria. Selection criteria can include package level, type, and status; creation and installation characteristics; associated release (if the Enterprise Release Option is installed); and included components or component types.

The queried packages may belong to multiple applications. Package components may reside in baseline libraries as well as package staging libraries.

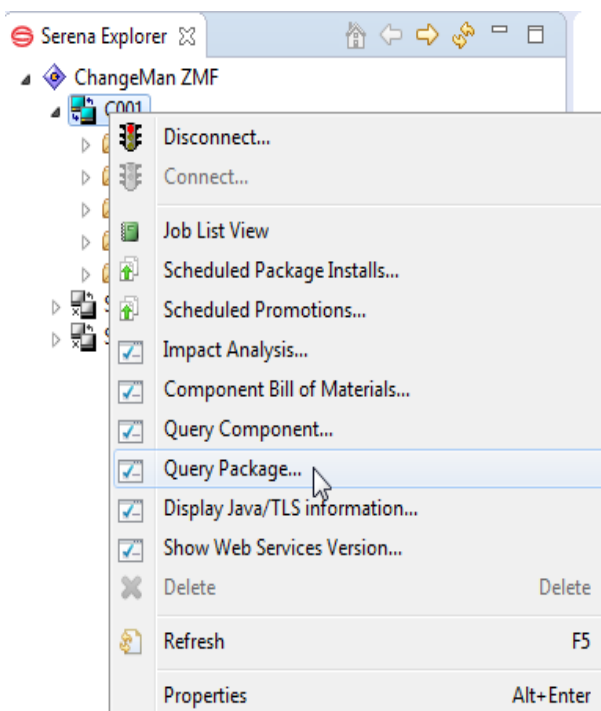
See the *ChangeMan ZMF User's Guide* for more information about package queries.

## Invoking a Package Query and Viewing Results

**Invoking a Package Query** Because a package query may span multiple applications and multiple libraries, the **Query Package** function is requested at the ZMF server level. It is invoked from the following menu:

- **Serena perspective** — In the **Serena Explorer** navigation view of the **Serena** perspective, right-click on the ZMF server hosting the repository where the target packages reside. When the contextual menu displays, select **Query Package**.

See [Chapter 2, "ZMF Server Functions" on page 36](#) for more information.



**Viewing Package Query Results** The results of a package query are displayed in a tabbed ZMF table view in the bottom right pane of the perspective. Any column may be ASCII text sorted by clicking on the heading. The title in the results tab is **Query Package**.

Package	Status	Install	Level	Type	Work request	Dept	Promotion	Audit	Creator	Title
GENL000001	Baselined	2014/12/23	Simple	Planned Permanent	100001000106	IDD	0	00	JPRESTO	Package for testing a
COMM000003	Baselined	2015/01/05	Simple	Planned Permanent	100001000106	IDD	0		JPRESTO	Package for testing a
COMM000002	Baselined	2014/12/17	Simple	Planned Permanent	100001000106	IDD	0	00	JPRESTO	Package for testing a
COMM000001	Baselined	2014/12/17	Simple	Planned Permanent	100001000106	IDD	0	20	JPRESTO	Package for loading a
ACTR000001	Development	2017/08/15	Simple	Planned Permanent	100001000106	IDD	0		JPRESTO	Package #4 for testin
ACTP000077	Backout	2016/11/19	Simple	Planned Permanent	1907D90	IDD	0	00	JPRESTO	New fixes
ACTP000076	Development	2016/08/22	Simple	Planned Permanent	TN3270	IDD	0		JPRESTO	Default Package Title
ACTP000075	Distributed	2017/08/22	Simple	Planned Permanent	TN3270	IDD	0	00	JPRESTO	Default Package Title
ACTP000074	Distributed	2017/08/20	Simple	Planned Permanent	WRQ3270	IDD	10	S6P1UT	JPRESTO	Default Package Title
ACTP000073	Development	2016/08/29	Simple	Planned Permanent	100001000106	IDD	0		JPRESTO	Package #4 for testin
ACTP000072	Development	2016/08/29	Simple	Planned Permanent	100001000106	IDD	10	S6P3UT3	JPRESTO	Package #3 for testin
ACTP000071	Development	2016/08/29	Simple	Planned Permanent	100001000106	IDD	0		JPRESTO	Package #1 for testin
ACTP000070	Distributed	2016/09/12	Simple	Planned Permanent	100001000106	IDD	0	04	JPRESTO	Package #1 for testin
ACTP000069	Development	2016/01/31	Simple	Planned Permanent	100001000106	IDD	0		JPRESTO	Package #1 for testin
ACTP000068	Development	2016/01/31	Simple	Planned Permanent	100001000106	IDD	0		JPRESTO	Package #1 for testin
ACTP000067	Development	2016/01/31	Simple	Planned Permanent		IDD	0		DJACOBS	mass stage test
ACTP000066	Development	2016/01/31	Simple	Planned Permanent		IDD	0		DJACOBS	This is a test package

Displaying Site Activities Information

You can right-click on a Package ID in the **Query Package** view to display the **Site Activities** dialog. See [Chapter 6, "Displaying Site Activities" on page 241](#) for more information about the **Site Activities** dialog.

## Selection Criteria

All selection criteria are optional. If the package list is left blank then all packages are queried.

Selection criteria are related logically as follows:

- Multiple values for the *same* field label are related using logical OR.
- Fields with *different* field labels are related using logical AND.

## Query Package Wizard

The **Query Package** wizard displays three windows containing package selection criteria. Use the **Next** and **Back** buttons to cycle through the wizard windows and find the criteria you need to enter. When ready to run the query, click **Finish**.

- 1 When you select the **Query Package** option from the ZMF server contextual menu, the first **Package Query** wizard screen displays. It requests search parameters concerning package names, package status, package level, and package type.

- a Package List** — If used, this field will restrict the package search by package ID according to the values used. Multiple entries are related by logical OR.



**NOTE** Package IDs consist of a 4-character alphanumeric application ID followed by a 6-digit package number.

Type any combination of the following:

- *Omit entirely to report on all packages.*
- *One or more package IDs* separated by semicolons. These packages will be included in the package search. Example:

ACPT000011;ACPT000022;ACPT000033

- *One or more pattern strings* to match against package IDs, separated by semicolons. Packages with IDs that match a pattern will be included in the package search results. Example:

ACPT00007\*

Patterns follow mainframe conventions. A letter or number requires itself as a value in the relative position where it appears. A question mark (?) is a wild card that stands for any one alphanumeric character in the position where it appears. An asterisk (\*) is a wild card that stands for a variable number of alphanumeric characters in the position where it appears.

- b** Enter the following information as relevant to your search.

- **Package Status** — Check the box beside each package status that you want to include in the search results. If no value is checked, all statuses are included in the search results.
- **Package Level** — Check the box beside each package complexity level that you want to include in the search results. If no value is checked, all package levels are included in the search results. (Note that some package levels are incompatible with some package types.)
- **Package Type** — Check the box beside each package type that you want to include in the search results. If no value is checked, all package types are included in the search results. (Note that some package types are incompatible with some package levels.)
- **Creator's id list** — Type one or more TSO user IDs or pattern strings to restrict search results to packages created by particular users. If omitted, all values are included in the search results.

Any combination of the following is allowed in this field:

One or more user IDs separated by semicolons. Packages created by these users will be included in the package search results.

One or more pattern strings to match against user IDs, separated by semicolons. Packages created by user IDs that match a pattern will be included in the package search results.

Patterns follow mainframe conventions. A letter or number requires itself as a value in the relative position where it appears. A question mark (?) is a wild card that stands for any one alphanumeric character in the position where it appears. An asterisk (\*) is a wild card that stands for a variable number of alphanumeric characters in the position where it appears.

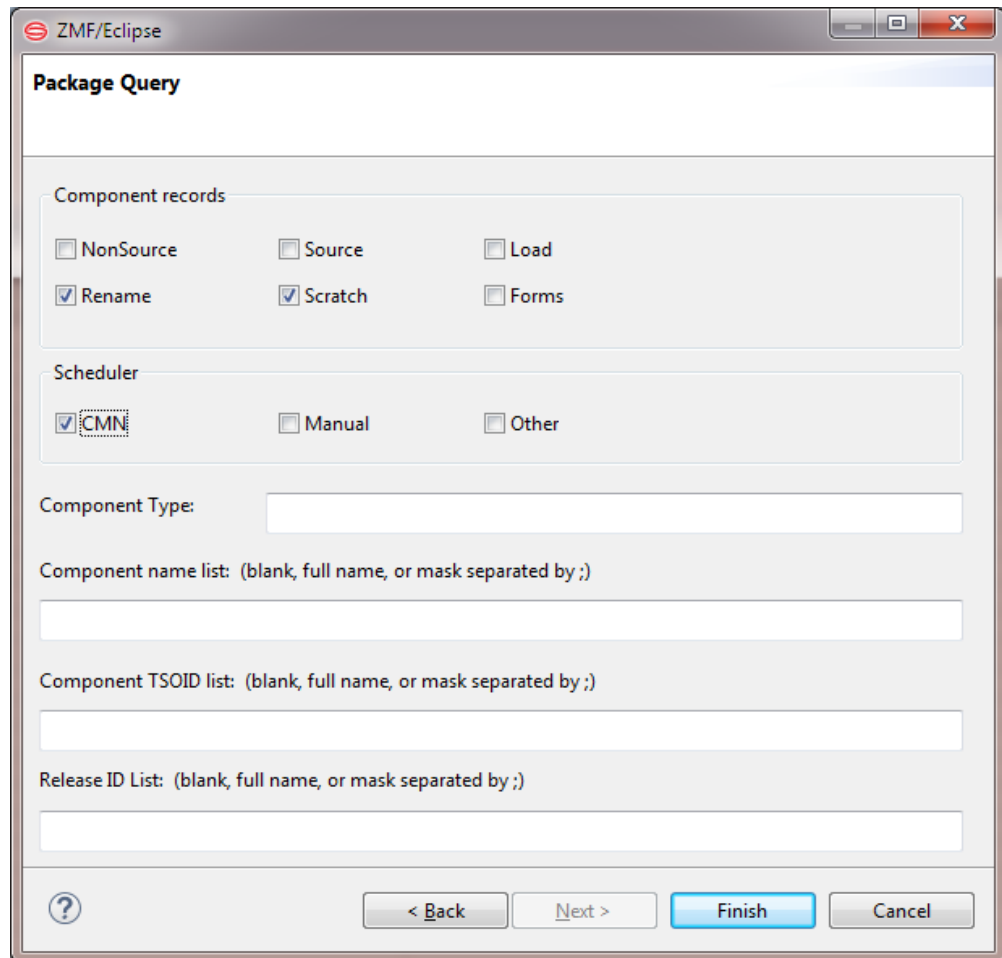
- **Work Request** — Type a value to require it in the search results. If an entry is omitted, all values are included in the search results. This field is limited to 12 characters. You may use one value with a wildcard e.g. 1\*.
  - **Department** — Type a value to require it in the search results. If an entry is omitted, all values are included in the search results. Limited to 4 characters. You may wildcard, e.g. D\*.
- c Click **Next** to bring up the next wizard window.
- 2 The second **Package Query** wizard screen requests search parameters concerning the work request and department, create dates and install dates, the install site targeted, and/or the package creator and package approver.

Enter the following information as relevant to your search. All values are optional.

- **From/To Install Date** — To enter a value, click on the month, day, or year segment of the date to select it. Click the up or down arrow at the right of the field to increment or decrement it. Alternatively, type a new value on top of the selected value. Default is current date.
- **From/To Create Date** — To enter a value, click on the month, day, or year segment of the date to select it. Click the up or down arrow at the right of the field to increment or decrement it. Alternatively, type a new value on top of the selected value. Default is current date.
- **Site Name** — Type an installation site name to use it as a filter for the search results. If this entry is omitted, all values are included in the search results.
- **Approvers Entity** — Must be a security entity defined in the package approver list. Type a value to require it in the search results. If an entry is omitted, all values are included in the search results. (Note that a value in this field is incompatible with a search for unplanned packages that have not been baselined.)

3 The final screen displayed:





Enter the following information as relevant to your search. All values are optional.

- **Component Records** — Check the box beside each component record type that a package must contain to be included in the search results. Multiple values are related by logical OR. If no value is checked, all component record types are included in the search results.
- **Scheduler** — Check the box beside each installation scheduler that a package must be associated with to be included in the search results. Multiple values are related by logical OR. If no value is checked, all schedulers are included in the search results.
- **Component Type** — Type one or more values or pattern strings (also known as *masks*) to restrict the search to packages containing components of that type. Separate multiple values with semicolons. If an entry is omitted, all values are included in the search results.

The following values are valid:

Component Type	Description
CPY	Copybooks
DOC	Documentation
JCL	JCL modules
LCT	Linkage control cards

Component Type	Description
LOD	Load modules
PRC	Compilation procedures
SRC	Source code modules
LIKE CPY	Library types defined as "like copybooks" to ZMF
LIKE LOD	Library types defined as "like load modules" to ZMF
LIKE SRC	Library types defined as "like source modules" to ZMF
LIKE N	Like NCAL load subroutines
LIKE O	Like object code
OTHER	Custom processing is associated with these components

- **Component Name List** — Type one or more component name or pattern strings to restrict search results to packages that include those components. If an entry is omitted, all values are included in the search results.

Any combination of the following is allowed in this field:

- *One or more component names* separated by semicolons. Packages including these components will be included in the package search results.
- *One or more pattern strings (masks)* to match against component names, separated by semicolons. Packages including components with names that match a pattern will be included in the package search results.

Patterns follow mainframe conventions. A letter or number requires itself as a value in the relative position where it appears. A question mark (?) is a wild card that stands for any one alphanumeric character in the position where it appears. An asterisk (\*) is a wild card that stands for a variable number of alphanumeric characters in the position where it appears.

- **Component TSO ID List** — Type one or more TSO user IDs or pattern strings to restrict search results to packages with components created by particular users. If an entry is omitted, all values are included in the search results.

Any combination of the following is allowed in this field:

- *One or more user IDs* separated by semicolons. Packages created by these users will be included in the package search results.
- *One or more pattern strings* to match against user IDs, separated by semicolons. Packages created by user IDs that match a pattern will be included in the package search results.

- **Release ID List** — Customers who use the Enterprise Release Option (ERO) of ChangeMan ZMF can type one or more release IDs or pattern strings to match against release IDs, separated by semicolons. An entry in this field restricts search results to packages associated with the named release(s). If an entry is omitted, all values are included in the search results.

- 4 Click **Finish** to submit the package query for execution.

# Querying Package Components

The **Package Components** function lists all the components in a change package, or any subset of those components that satisfy your search criteria.

## Invoking a Package Component Query

The **Package Components** query is invoked from the following menu:

- Serena perspective** — In the **Serena Explorer** navigation view, expand the node for the ZMF server where the package of interest resides. Then expand the **z/OS Applications** node, the node for the particular application containing the package, and its **Packages** node. Right-click on the desired package to bring up its contextual menu, then select the **Package Components** option.

The requested components are listed in a table view under the **Package Components View** tab in the lower right pane of the perspective. For example:

Package: WSRV000005

Component Name	Type	Status	Changed	Procedure	Language	Userid	Request	Source Data Set
com/serena/sercmn/zmf/constants/IAccessTypes.java	JV1	Active	2010/02/09 16:45:04	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/IApprovalActionCodes.java	JV1	Active	2010/02/09 16:49:46	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/IApprovalStatuses.java	JV1	Active	2010/02/09 16:48:16	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/IApproverListTypes.java	JV1	Active	2010/02/09 16:46:35	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/IBrowseOptions.java	JV1	Active	2010/02/09 16:54:50	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/ICheckoutRules.java	JV1	Active	2010/02/09 16:51:13	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/ICmpLocations.java	JV1	Active	2010/02/09 16:56:32	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/IComponentActions.java	JV1	Active	2010/02/09 16:53:02	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/IComponentStatuses.java	JV1	Active	2010/02/09 16:59:46	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/IComponentTypes.java	JV1	Active	2010/02/09 17:02:15	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/ICreateMethods.java	JV1	Active	2010/02/09 16:57:55	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/IDataTypes.java	JV1	Active	2010/02/09 17:03:45	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/IDatasetOrgs.java	JV1	Active	2010/02/09 17:07:46	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1
com/serena/sercmn/zmf/constants/IDirTypes.java	JV1	Active	2010/02/09 17:10:12	CMNJAVA	JAVA	WSER27		HFS - /cmndev/cmng7/WSRV/#000005/JV1

## Package Component List Wizard

The **Package Components** list wizard displays just one window, titled **Select Package List Parameters**.

The screenshot shows a dialog box titled "Select Package List Parameters" with the following fields and options:

- Package: ACTP000075
- Component Name: \*
- Library Type: \*
- Language: (empty)
- Procedure Name: (empty)
- Userid: (empty)
- Changed From: 04-Sep-2016
- Changed To: 22-Sep-2017
- Component Status:
  - Active
  - Checked Out
  - Frozen
  - Inactive
  - Incomplete
  - Unfrozen
- Component Type:
  - All Component Types
  - Source Components
  - Non Source Components
- Mixed Case

Buttons: OK, Cancel

Enter the following information as relevant to your search.

- **Package** — ID of package whose components will be listed. Defaults to the selected package, but may be edited in the text box.
- **Component Name** — Name of desired component(s). May be full component name or a search pattern with wildcards. Defaults to the asterisk wildcard, which matches any component name.
- **Library Type** — Library type of desired components. May be full library type identifier or a search pattern with wildcards. Defaults to the asterisk wildcard, which matches any library type.
- **Language** — Source code language of desired components.
- **Procedure Name** — Compile procedure associated with desired components.
- **User ID** — TSO ID of user who staged the desired components into the change package (either by checkout or checkin).
- **Changed From** — If searching by date range for the most recent change, enter the oldest (or "from") date for the desired components. Select from pull-down list or type.

- **Changed To** — If searching by date range for the most recent change, enter the newest (or "to") date for the desired components. Select from pull-down list or type.
- **Component Status** — Check the box beside each component status desired. Uncheck any component statuses that should be excluded from the search results.
- **Component Type** — Optionally restrict the component list to like-source components only, non-source components only, or all component types.
- **Mixed Case** — Check this box to match case in names during the search. Deselect the checkbox to disregard case in names.

Click **OK** to submit the query for execution.

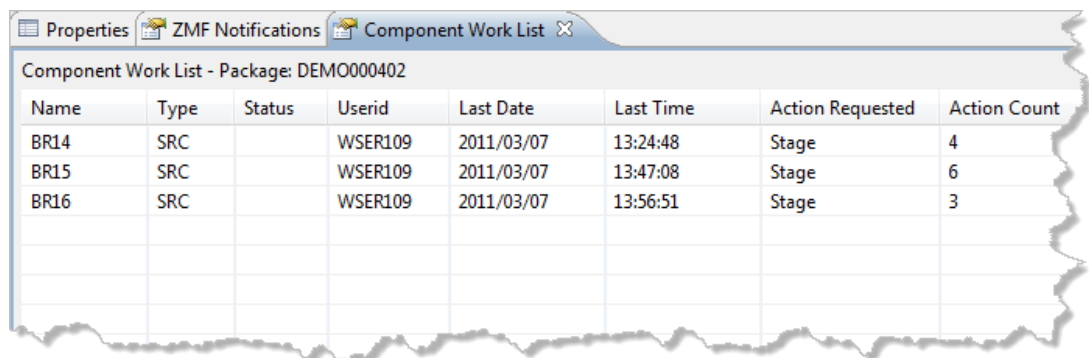
## Viewing the Component Work List

The **Component Work List** function identifies the most recent users to change each component in the selected change package. Results are listed by component and include the TSO user ID, the action requested, and the date and time that action was most recently performed. An action count shows the total number of changes performed on the component by the named user.

The **Component Work List** function is invoked from the following menu:

- **Serena perspective** — In the **Serena Explorer** navigation view, expand the node for the ZMF server where the package of interest resides. Then expand the **z/OS Applications** node, the node for the particular application containing the package, and its **Packages** node. Right-click on the desired package to bring up its contextual menu, then select the **Component Work List** option.

The requested relationships are listed in a table view under the **Component Work List** tab in the lower right pane of the perspective. For example:



Name	Type	Status	Userid	Last Date	Last Time	Action Requested	Action Count
BR14	SRC		WSER109	2011/03/07	13:24:48	Stage	4
BR15	SRC		WSER109	2011/03/07	13:47:08	Stage	6
BR16	SRC		WSER109	2011/03/07	13:56:51	Stage	3

## Viewing Source-to-Load Relationships

The **Source-to-Load Relationships** function lists the source-to-load relationships that exist among components in the selected change package. Use this function to find the compiled modules in a package, to identify components that have not yet been built, or to identify components that require a recompile based on change timestamp.

The **Source-to-Load Relationships** function is invoked from the following menu:

- Serena perspective** — In the **Serena Explorer** navigation view, expand the node for the ZMF server where the package of interest resides. Then expand the **z/OS Applications** node, the node for the particular application containing the package, and its **Packages** node. Right-click on the desired package to bring up its contextual menu, then select the **Source-to-Load Relationships** option.

The requested relationships are listed in a table view under the **Source-to-Load Relationships** tab in the lower right pane of the perspective. For example:

Source to load - package -

Name	Type	Target Name	Target Type	Status	Promotion	Changed	Userid	Setssi
ACPSRC90	SRC	ACPSRC90	LOD	Active	0 Staging	2016/08/11 21:18:14	JPRESTO	6A7CE2FD
ACPSRC90	SRC	ACPSRC90	LST	Active	0 Staging	2016/08/11 21:18:17	JPRESTO	6A7CE2FD
ACPSRC4A	SRC	ACPSRC4A	LOD	Active	0 Staging	2016/08/04 22:16:31	JPRESTO	6A73B627
ACPSRC4A	SRC	ACPSRC4A	LST	Active	0 Staging	2016/08/04 22:16:35	JPRESTO	6A73B627

## Viewing Scratch/Rename Components

The **Scratch/Rename Components** function lists all control records in a package that will manage the execution of versioned deletions (scratches) or versioned renames of baselined components. The scratch or rename actions are not actually performed until the package is baselined.

The **Scratch/Rename Components** function is invoked from the following menu:

- Serena perspective** — In the **Serena Explorer** navigation view, expand the node for the ZMF server where the package of interest resides. Then expand the **z/OS Applications** node, the node for the particular application containing the package, and its **Packages** node. Right-click on the desired package to bring up its contextual menu, then select the **Scratch/Rename Components** option.

The requested control records are shown in a table view under the **Package Scratch/Rename Components** tab in the lower right pane of the perspective. For example:

Package Scratch/Rename Components - Package: ACTP000076

Request	Name	Type	Status	Changed	Userid	Rename
Rename	ACPCPY3X	CPY	Active	2016/08/22 03:17:38	JPRESTO	ACP908
Scratch	ACPCPY2X	CPY	Active	2016/08/22 03:17:05	JPRESTO	

Related functions are described in the following topics.

- See [Chapter 5, "Scratching a Component under Change Control" on page 125](#) for step-by-step instructions on performing a versioned deletion (scratch) of a component.
- See [Chapter 5, "Renaming a Component under Change Control" on page 127](#) for step-by-step instructions on performing a versioned rename of a component.
- See [Chapter 6, "Removing Scratch Records from a Package" on page 193](#) for step-by-step instructions on removing scratch control records from a change package.

- See [Chapter 6, "Removing Rename Records from a Package" on page 193](#) for step-by-step instructions on removing rename control records from a change package.

## Viewing Promotion History

The **Promotion History** function displays a history of promotion and demotion actions performed on the selected change package at a particular promotion site. The resulting table lists the promotion/demotion actions performed, a count of the components involved in each action, the promotion level and nickname targeted by the promotion/demotion action, the type of action performed, the user ID of the action requestor, a timestamp for the action, and the resulting package status.

The **Promotion History** function is invoked from the following menu:

- **Serena perspective** — In the **Serena Explorer** navigation view, expand the node for the ZMF server where the package of interest resides. Then expand the **z/OS Applications** node, the node for the particular application containing the package, and its **Packages** node. Right-click on the desired package to bring up its contextual menu, then select the **Promotion History** option.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

- **Java perspective** — In the **Package Explorer** navigation view, navigate to a project that is shared with a ZMF package. Right-click on the project to bring up its contextual menu, then open the **Team** submenu and select the **ZMF Package Promotion History** option.

See [Chapter 3, "Project-Level ZMF Functions" on page 102](#) for more information.

The **Promotion History** results are displayed under the **Package Promotion History** tab in the lower right pane of the perspective.

Related functions are described in the following topics.

- To see the promotion history of one or more selected components, regardless of associated package, use the **Component History** function instead of the **Promotion History** function. See [Chapter 5, "Viewing Component History" on page 138](#) for more information.
- To see only the promotion libraries currently associated with a change package, use the **Promotion Libraries** function instead of the **Promotion History** function. See ["Viewing Promotion Libraries" on page 239](#) for more information.

## Viewing Promotion Libraries

The **Promotion Libraries** function displays a list of promotion libraries currently associated with a selected change package. The library names, library types, shadow libraries, promotion sites, and promotion levels associated with the package are displayed for each library in the selected package.

To see the complete promotion history associated with a change package and its components, use the **Promotion History** function instead of the **Promotion Libraries** function. See ["Viewing Promotion History" on page 239](#) for more information.

The **Promotion Libraries** function is invoked from the following menu:

- **Serena perspective** — In the **Serena Explorer** navigation view, expand the node for the ZMF server where the package of interest resides. Then expand the **z/OS Applications** node, the node for the particular application containing the package, and its **Packages** node. Right-click on the desired package to bring up its contextual menu, then select the **Promotion Libraries** option.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.

LibType	SiteName	Level	Promotion Library 01	Promotion Library 02	Promotion Library 03	Shadow Library
CTC	SERT6	10	CMNTP.S6.V810.PROM.S6P1UT.CTC			CMNTP.S6.V810.PROM.S6P1UT.CTC
JCL	SERT6	10	CMNTP.S6.V810.PROM.S6P1UT.JCL			CMNTP.S6.V810.PROM.S6P1UT.JCL
JVS	SERT6	10	/cmntp/s6/actp/prom10/jvs			/cmntp/s6/actp/prom10/jvs
JVT	SERT6	10	/cmntp/s6/actp/prom10/jvt			/cmntp/s6/actp/prom10/jvt
JVL	SERT6	10	/cmntp/s6/actp/prom10/jvl			/cmntp/s6/actp/prom10/jvl
LOD	SERT6	10	CMNTP.S6.V810.PROM.S6P1UT.LOD			CMNTP.S6.V810.PROM.S6P1UT.LOD
PRC	SERT6	10	CMNTP.S6.V810.PROM.S6P1UT.PRC			CMNTP.S6.V810.PROM.S6P1UT.PRC
SRC	SERT6	10	CMNTP.S6.V810.PROM.S6P1UT.SRC			CMNTP.S6.V810.PROM.S6P1UT.SRC
CTC	SERT6	20	CMNTP.S6.V810.PROM.S6P1IT.CTC			CMNTP.S6.V810.PROM.S6P1IT.CTC
JCL	SERT6	20	CMNTP.S6.V810.PROM.S6P1IT.JCL			CMNTP.S6.V810.PROM.S6P1IT.JCL
LOD	SERT6	20	CMNTP.S6.V810.PROM.S6P1IT.LOD			CMNTP.S6.V810.PROM.S6P1IT.LOD
PRC	SERT6	20	CMNTP.S6.V810.PROM.S6P1IT.PRC			CMNTP.S6.V810.PROM.S6P1IT.PRC
SRC	SERT6	20	CMNTP.S6.V810.PROM.S6P1IT.SRC			CMNTP.S6.V810.PROM.S6P1IT.SRC
CTC	SERT6	30	CMNTP.S6.V810.PROM.S6P1AT.CTC			CMNTP.S6.V810.PROM.S6P1AT.CTC
JCL	SERT6	30	CMNTP.S6.V810.PROM.S6P1AT.JCL			CMNTP.S6.V810.PROM.S6P1AT.JCL
LOD	SERT6	30	CMNTP.S6.V810.PROM.S6P1AT.LOD			CMNTP.S6.V810.PROM.S6P1AT.LOD
PRC	SERT6	30	CMNTP.S6.V810.PROM.S6P1AT.PRC			CMNTP.S6.V810.PROM.S6P1AT.PRC
SRC	SERT6	30	CMNTP.S6.V810.PROM.S6P1AT.SRC			CMNTP.S6.V810.PROM.S6P1AT.SRC
CTC	SERT6P1	10	CMNTP.S6.V810.PROM.S6P1UT1.CTC			CMNTP.S6.V810.PROM.S6P1UT1.CTC

The promotion library list displays in a pop-up window. When finished viewing, click **OK**.

## Viewing Baseline Libraries

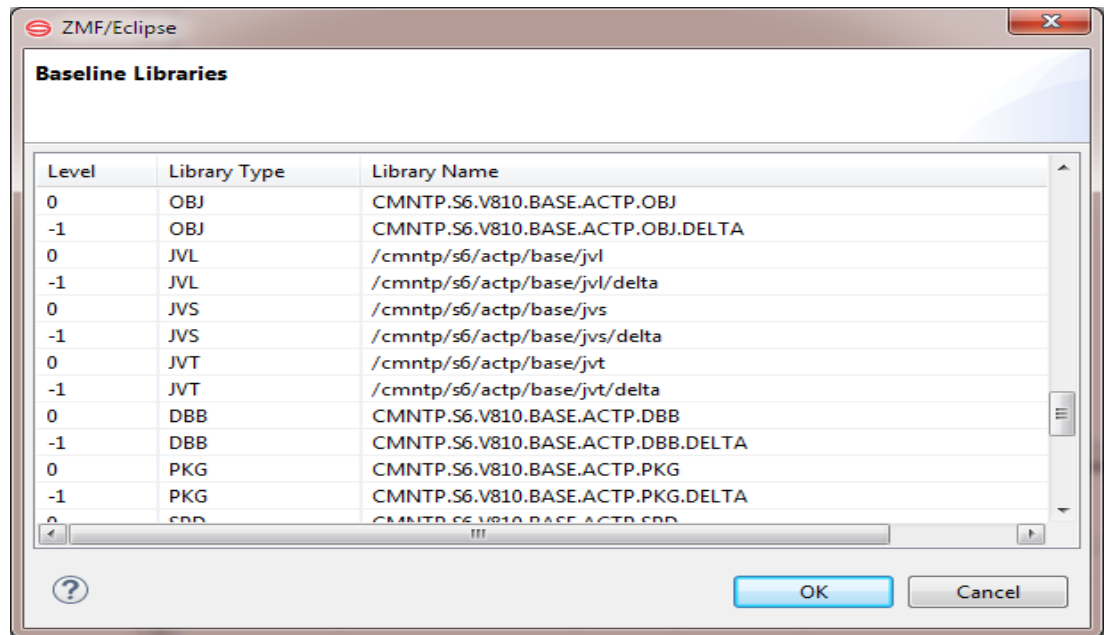
The **Baseline Libraries** function displays a list of baseline libraries associated with a selected change package. The library names, library types, and baseline levels associated with the package are displayed for each library in the selected package.

The **Baseline Libraries** function is invoked from the following menu:

- **Serena perspective** — In the **Serena Explorer** navigation view, expand the node for the ZMF server where the package of interest resides. Then expand the **z/OS Applications** node, the node for the particular application containing the package, and its **Packages** node. Right-click on the desired package to bring up its contextual menu, then select the **Baseline Libraries** option.

See [Chapter 2, "Working with ZMF Packages" on page 62](#) for more information.



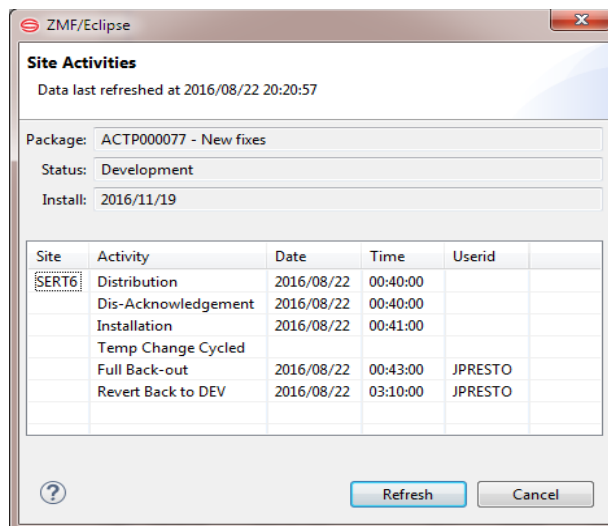


The baseline library list displays in a pop-up window. When finished viewing, click **OK**.

## Displaying Site Activities

Take the following actions to display the starting date and time stamp of activities that have been performed at the sites where the selected package was or will be installed:

- In the Serena Explorer navigation view, right-click on a package ID to bring up the package contextual menu.
- Select **Site Activity** from the menu. The **Site Activities** dialog is displayed:



The dialog is a display-only dialog that has the following fields:

<b>Field</b>	<b>Description</b>
Package	Package ID and description.
Status	Current status of the package.
Install	Installation date.
Site	Site ID where the package has been or will be installed.
Activity	List of activities that have been performed against the package at the site.
Date	Date the associated activity was performed.
Time	Time the associated activity was performed.
Userid	TSO userid of the user who performed the activity.

To refresh the information in the display, click the **Refresh** button or press the **F5** key.

## Appendix A

---

# Error Messages and Troubleshooting

Unidentified Nested Exception at ZMF Logon	244
Logoff Fails Due to Invalid Session	244
Component Browse Function Fails	244
Checkout to RDz Project Fails	245
Exception When Sharing an Eclipse Project	245
Code Page Issues with Comments in ZDDOPTS	245
ZDDOPTS Updates Do Not Affect Workbench	247

## Unidentified Nested Exception at ZMF Logon

- Error Message** The following unidentified message displays when the ZMF web services are unable to establish a connection to ChangeMan ZMF.
- ```
Return Code: 8
Reason Code: 8
Message: ;nested exception is:
```
- Explanation** The most common cause of this error is a web application server that has not been started as a service under Windows.
- Resolution** To fix this problem, start — or stop and then restart — the web application server used by ZMF for Eclipse.
- From the Windows **Start** menu, navigate to **Administrative Tools | Services**. From the list of services shown, select your web application server (for example, Apache Tomcat or WebSphere Application Server), then click **Start** or **Restart**.

## Logoff Fails Due to Invalid Session

- Error Message** Depending on the security settings on your z/OS mainframe, you may get the following message when you select the **Disconnect** option to logoff from ZMF:
- ```
Logoff failed
Reason: Invalid session.
```
- Resolution** This occurs if your ZMF for Eclipse session with z/OS has timed out. You have already been logg off. No corrective action is required.

## Component Browse Function Fails

- Error Message** When attempting to browse ChangeMan ZMF components, the following error occurs:
- ```
Could not initialize the editor
```
- Explanation** This error occurs when a network firewall blocks access to the XCH (Sernet Connect) and/or ZMF (CMN) ports for the target ChangeMan ZMF instance on the mainframe.
- Resolution** To resolve this problem, first determine the XCH and ZMF port numbers for the ZMF instance you are trying to access. These can be found on the ZMF server information tab in the Serena perspective of ZMF for Eclipse.
- Provide the XCH and ZMF port numbers to your network administrator and request that these be opened for client access over the TCP/IP network where your web application server resides.

## Checkout to RDz Project Fails

|               |                                                                                                                                                                                                                                                                                                                                                                 |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Error Message | After selecting <b>ZMF-RDz   Checkout</b> from the <b>Serena Explorer</b> contextual menu and providing a valid RDz project and subproject name, you may encounter a message like the following:<br><br>Project undefined or no data set                                                                                                                        |
| Explanation   | This error occurs when no PDS libraries or HFS directories have been linked to the subproject targeted by the checkout. An RDz project or subproject is a logical collection, not a physical container for checked out files. A physical library or directory from the RDz <b>Remote Explorer</b> view must be linked to an RDz subproject before ZMF checkout. |
| Resolution    | Link a physical library or directory in the RDz <b>Remote Explorer</b> view to an appropriate RDz subproject. Then retry the ZMF checkout.                                                                                                                                                                                                                      |

## Exception When Sharing an Eclipse Project

|               |                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Error Message | If the ZDDOPTS configuration library is not installed with ChangeMan ZMF on the mainframe, or if the ECLIPSE member of that library is missing or incorrect, you will encounter the following message when you attempt to share an Eclipse project with ZMF:<br><br>‘Share Eclipse project with ZMF’ has encountered a problem -<br>Exception running background job Share Eclipse with ZMF                           |
| Explanation   | Each ChangeMan ZMF instance on the mainframe must be configured to support access from ZMF for Eclipse. To do this, your ZMF administrator must install the ZDDOPTS configuration library on each ZMF instance that will be accessed using ZMF for Eclipse. In addition, the ECLIPSE member of each library must be appropriately configured for library type mapping between the mainframe and your Eclipse project. |
| Resolution    | Instructions for ZDDOPTS installation and setup are provided in the <i>ChangeMan ZMF for Eclipse Installation and Setup</i> document.                                                                                                                                                                                                                                                                                 |

## Code Page Issues with Comments in ZDDOPTS

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Error Message | The first time you execute a ZMF process that uses the XML specification in a ZDDOPTS configuration library, you may encounter an error such as the following:<br><br>ChangeMan [ChangeMan ZMF instance] on server [ChangeMan ZMF server]<br>has invalid XML data specified for ZDDOPTS in [library(member)].<br><br>Missing equals sign between attribute and attribute value.                                                                                                                                         |
| Explanation   | This message occurs when in an XML comment in ZDDOPTS is processed using a non-English character code page.<br><br>Serena supplies sample ZDDOPTS members on the product media for ChangeMan ZMF and ZMF for Eclipse. The sample members were created for use with English-language character code pages on the mainframe. If you are using different code pages, you may need to modify the XML comments in these sample ZDDOPTS members.<br><br>All ZDDOPTS members use standard XML syntax for comments, as follows: |



## ZDDOPTS Updates Do Not Affect Workbench

|             |                                                                                                                                       |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------|
| Symptom     | Changes are made to a ZDDOPTS configuration member on the mainframe, but the behavior of ZMF for Eclipse does not reflect the change. |
| Explanation | You must refresh the ZMF server settings on the client for the new settings to take effect in ZMF for Eclipse.                        |
| Resolution  | Disconnect from and reconnect to the ZMF server. It is <b>not</b> necessary to restart the ZMF started task.                          |





# Index

---

## A

- Add to Workspace option 101
- Adobe URL 11
- Apache Software Foundation URL 9
- application name 178
- Application/Package Filters window 88
- applications
  - adding to workspace 60
  - filtering 30
  - filtering views 87
  - function list 59
  - mass component download 60
  - navigating 36, 57
  - navigating in Serena Explorer 58
  - Packages node 176
  - ZMF Applications node 57
- auditing a change package
  - cross-application audit options 219
- auditing a package
  - auto resolve synch errors 217
  - description 66
  - reset audit lock 65

## B

- backing out a change package 65
- baseline libraries
  - component functions 67, 69
  - viewing 240
- bill of materials 37

## C

- ChangeMan ZMF 14
  - described 14
  - Web Services 10, 15
- character code pages 23
- checkout to RDz project fails 245
- complex or super package 179, 180
- component history 217
- components
  - bill of materials 37, 165–167
  - browsing 59, 141
  - browsing a listing 142
  - browsing staging versions 162
  - build jobs 49, 55, 118

- building 103, 106, 108, 134, 143–150
- change description 137
- checking in 49, 55, 103, 106, 108, 111, 118, 134–138
- checking out 103, 106, 108, 129–133
- comparing 163–165
- comparing staging versions 162–163
- comparing to baseline or promotion 163–164
- component history list 108
- component types 233
- component work list 237
- contextual menus, baseline 67–69
- creating 122
- deleting immediately 124–125
- deleting under change control 125–127
- editing 143
- functions in baseline libraries 67–69
- history list 138–141
- impact analysis 36, 168–171
- listing 123
- locking 103, 105, 108, 141
- mass checkin 101
- mass download 60
- querying 235
- recompiling 150–155
- refreezing 103, 198–199
- relinking 155–159
- removing rename control records 128
- removing scratch control records 125
- renaming under change control 127
- reverting to package level 102, 105, 108
- scratch/rename records 238
- scratching 125–127
- staging (see checking in)
- staging versions 108, 142, 159–163
- unfreezing 103, 197–198
- unlocking 103, 105, 108, 141

- contextual menus
  - applications 59
  - baseline components 68
  - dataset libraries 38
  - dataset members 47
  - Java components 107
  - Java folders 105
  - Java projects 102
  - package components 70
  - packages 63
  - promotion components 73
  - Team submenu 102, 105, 107

- Unix HFS files 53
- Unix HFS folders 51
- ZMF servers 36

cross-application audits 220

## D

dataset members

- browsing 48
- build jobs 49, 118
- checking in to a package 49, 118
- creating 40
- deleting 50
- described 46
- downloading 48
- editing 47
- function list 46
- refreshing 50
- submitting for execution 49
- submitting XML Services requests 49
- ZMF function list 47

dataset naming conventions 78

datasets

- creating new members 40
- Default Data Set Filter 37
- deleting 41
- described 37
- downloading libraries 41
- downloading single members 48
- filtering 30, 37
- filtering views 77–80
- function list 37
- migrating offline 42
- navigating 35
- recalling from offline storage 42
- refreshing a view 39

demotion level 211

demotion site 211

## E

EBCDIC to Unicode 23

Eclipse Project URL 9

editors 23

error messages

- could not initialize editor 244
- exception running Share Eclipse with ZMF 245
- invalid session 244
- invalid XML specified in ZDDOPTS 245
- logoff failed 244
- missing equal sign 245
- nested exception, unidentified 244
- project undefined or no data set 245

executing mainframe jobs 49

## F

filter strings

- application filter string syntax 88

filters

- application views 59
- applications 87
- baseline library views 59
- dataset filter string syntax 78
- dataset filters, creating 77–79
- dataset filters, deleting 80
- dataset filters, modifying 79
- dataset filters, refreshing 80
- dataset filters, renaming 79
- datasets 77–80
- default 29
- described 29
- job filter string syntax 86
- job filters, creating 85–86
- job filters, deleting 87
- job filters, modifying 86
- job filters, refreshing 87
- job filters, renaming 86
- job name 85
- job owner 85
- jobs 84–87
- packages 88
- Unix HFS files 81–84
- Unix HFS filter string syntax 82
- Unix HFS filters, creating 81–??
- Unix HFS filters, deleting 84
- Unix HFS filters, modifying 83
- Unix HFS filters, refreshing 84
- Unix HFS filters, renaming 83

## H

HFS files

- see Unix HFS files or folders

Hierarchical File System (HFS) 50

History button 130, 135

## I

IBM Rational developerWorks URL 9

IBM WebSphere Application Server (WAS) URL 10

impact analysis 36

installation JCL 224

invalid session 244

**J**

Java perspective  
 dynamic integration with ZMF 62, 100  
 enabling ZMF functions 99  
 function list 97  
 mass checkin 101  
 navigation 96  
 opening 97  
 Package Explorer view 96  
 sharing projects with ZMF 99  
 viewing 98  
 ZMF functions 101

**jobs**

cancelling 57  
 deleting output 57  
 filtering 30  
 filtering views 84–87  
 navigating 55  
 output 55  
 viewing output 56

**L**

library mapping 60  
 listings 59  
 listings, viewing 23  
 lock package for audit 217  
 log off from ZMF 35  
 log on to ZMF 33  
 logoff failed 244

**M**

mass checkin 101

**N**

nested exception, unidentified 244  
 notifications 75

**P**

package audit parameters 216  
 Package Explorer  
 described 96  
 function list 101  
 package properties wizard 64  
 package staging libraries  
 component functions 69  
 package title 178  
 packages

affected applications 183  
 approving 222–224  
 audit 66  
 auditing 104, 213–221  
 back out of production 65  
 backing out 225–227  
 baseline libraries 240  
 complex 179  
 component list 235  
 component work list 237  
 contextual menu 63  
 Create Package wizard 176, 192  
 creating 176–190  
 defined 176  
 deleting 190–191  
 demoting 64, 103, 209–213  
 described 62  
 editing 192  
 editing properties 64  
 filtering views 88  
 freeze 65  
 freezing 103, 194–195  
 installation contingencies 182  
 installation instructions 182  
 installation scheduler 182  
 level 179  
 lifecycle 180  
 listing components of 123  
 long creation method 179  
 package IDs 230  
 participating 179  
 permanent 180  
 planned change 179  
 promote 64  
 promoting 103, 199–206  
 promotion history 103, 106, 108  
 promotion history list 239  
 promotion libraries 239  
 properties 192  
 query 37  
 querying 37, 228–234  
 querying components in 235  
 quick creation method 177, 179, 192  
 rebuilding installation JCL 224  
 refreeze component 65  
 refreezing components 197  
 rejecting 222–224  
 removing rename records 193–??  
 removing scratch records 193  
 requestor 178  
 reset audit lock 65  
 resetting audit lock 221  
 revert to development status 65  
 reverting 227–228  
 scheduling dependencies 182  
 scratch/rename components list 238

- searching 228–234
- short creation method 177, 179, 192
- simple 179
- site information 188
- source-to-load relationships 237
- super 179
- temporary 180
- temporary install duration 180
- types 179
- undeleting 191
- unfreeze component 64
- unfreezing components 197
- unplanned change 179
- user variables 185
- work requests 178
- participating package 179, 180
- patterns in string searches 230, 234
- perspectives
  - Java perspective 95, 96
  - RDz perspectives 109
  - Remote Systems perspective (RDz) 110
  - Serena 26
  - Serena perspective 21
  - switching 27, 98
  - z/OS Projects perspective 110
  - z/OS Projects perspective (RDz) 110
- plug-in 15
- plug-in differences by IDE 18
- project undefined 245
- projects
  - build components 103, 106, 108
  - check in components 103, 106, 108
  - check out components 103, 106, 108
  - lock components 103, 105, 108
  - revert components to package level 102
  - sharing with ZMF 99
  - unlock components 103, 105, 108
  - unshare 102, 105, 107
- promoting a package 64
- promotion level 202
- promotion libraries
  - component functions 73–75
  - viewing for packages 239
- promotion site 202
- Properties view 23

## Q

- query package 37

## R

- RDz projects
  - checking in a component from 111, 118

- logical collections 110
- organization 111
- z/OS Projects view 110
- refreezing a package component 65
- repository 15
- reset audit lock 65
- revert a package to development status 65

## S

- scheduler
  - ChangeMan ZMF, automatic 182
  - external 182
- Serena Explorer 22
  - applications 58
  - filtering views in 29
  - hiding 29
  - navigating 35
  - showing 28
  - Unix HFS files 35
  - z/OS USS files 35
  - ZMF applications 36
- Serena perspective
  - contextual menus 59
  - function list 26
  - Serena Explorer view 22
- Share Project option 101
- sites
  - DP site type 188

## T

- table views
  - package components 123
  - staging versions 160
- Team contextual menu 101
- The contextual menu for a Release. 92
- The contextual menu for a specific Release. 92
- The contextual menu for Releases. 91
- troubleshooting
  - code page issues 245
  - component browse function fails 244
  - editor won't initialize 244
  - exception when sharing Eclipse project with ZMF 245
  - ZDDOPTS changes don't show 247
  - ZDDOPTS issues 245

## U

- unfreezing a package component 64
- Unicode to EBCDIC 23
- Unix HFS files

- browsing 55
  - build jobs 55
  - building 55
  - checking in to a package 55
  - checking in to a ZMF package 55
  - default HFS filter 50
  - deleting 54
  - downloading 54
  - editing 55
  - filtering 30
  - filtering views 81–84
  - function list 53
  - navigating 35, 50
  - refreshing 54
  - viewing 56
  - Unix HFS folders
    - deleting 51
    - downloading 51
    - function list 50
    - refreshing a view 51
  - Unshare Project option 101
  - use cases
    - COBOL application maintenance 17, 18
    - enterprise e-commerce with Java 15, 16
  - USS files
    - see Unix HFS files or folders
- W**
- web application server 15
  - web services
    - showing version 37
  - work requests 178
  - Working with ZMF Releases 91
  - workspaces
    - integration with ZMF 60
    - mass download from ZMF 60
- X**
- XML Services
    - submitting XML requests 49
    - viewing XML replies 49
- Z**
- z/OS jobs
    - navigating 35
  - z/OS Projects perspective
    - Remote Systems view 110
    - z/OS Projects view 110
  - z/OS USS Files node 50
  - ZDDOPTS parameter library 177, 192
  - code page issues 245
  - downloaded file extensions and 42, 48
  - library mapping 60
  - package variables 185
  - ZMF for Eclipse
    - benefits 14
    - described 9
    - features and functions 14
    - plug-in differences by IDE 18
    - software components 15
    - use cases 15, 16, 17, 18
  - ZMF functions
    - applications 59
    - component function list 121
    - components, baseline 68
    - components, package 70
    - components, promotion 74
    - dataset libraries 38
    - dataset members 47
    - dynamic integration 100
    - enabling in Java perspective 99
    - invoking with mainframe libraries 57
    - Java components and 107
    - Java folders 105
    - Java folders and 105
    - Java projects 102
    - Java projects and 102
    - Package Explorer view 101
    - package function list 175
    - packages 64
    - Serena Explorer view 33, 35
    - Unix HFS files 53
    - Unix HFS folders 50, 51
    - ZMF servers 36
  - ZMF repository
    - logging off 35
    - logging on 33
  - ZMF servers
    - accessing 33
    - deleting 36
    - properties 36
    - resources 35
    - viewing 29
    - web services version 37
  - ZMF table views
    - notifications 75
    - opening 31
  - ZMF Applications node
    - see applications 57

