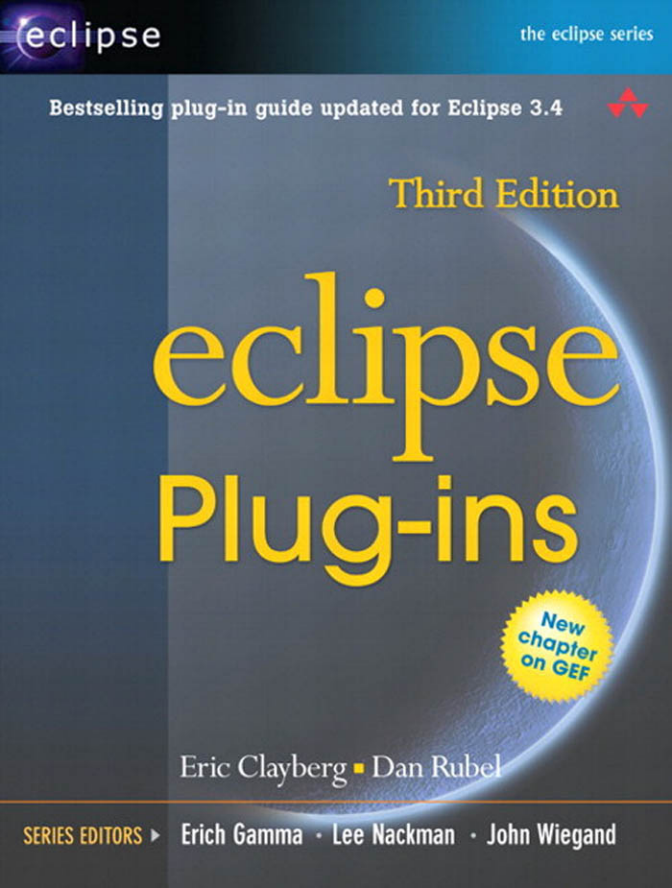


Bestselling plug-in guide updated for Eclipse 3.4



Third Edition



eclipse Plug-ins



New
chapter
on GEF

Eric Clayberg ■ Dan Rubel

SERIES EDITORS ▶ Erich Gamma • Lee Nackman • John Wiegand

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U.S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the United States please contact:

International Sales
international@pearsoned.com

Visit us on the Web: informit.com/aw

Library of Congress Cataloging-in-Publication Data

Clayberg, Eric.

Eclipse : plug-ins / Eric Clayberg, Dan Rubel. -- 3rd ed.
p. cm.

Includes bibliographical references and index.

ISBN 0-321-55346-2 (pbk. : alk. paper)

1. Computer software--Development. 2. Eclipse (Electronic resource) 3. Java (Computer program language) I. Rubel, Dan. II. Title.

QA76.76.D47C574 2008
005.13'3--dc22

2008047409

Copyright © 2009 Pearson Education, Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, write to:

Pearson Education, Inc.
Rights and Contracts Department
501 Boylston Street, Suite 900
Boston, MA 02116
Fax: (617) 671-3447

ISBN-13: 978-0-321-55346-1
ISBN-10: 0-321-55346-2

Text printed in the United States on recycled paper at Courier in Stoughton, Massachusetts.

First printing, December 2008

Foreword

To the millions of developers, engineers, and users all over the world, Eclipse is an extensible platform for tool integration. To the hundreds of thousands of commercial customers using it to develop plug-ins or complete tool platforms, Eclipse represents a proven, reliable, scalable technology on which commercial products can be quickly designed, developed, and deployed.

To the thousands of students and researchers, Eclipse represents a stable platform for innovation, freedom, and experimentation. To all these individuals, groups, and organizations, Eclipse is a vendor-neutral platform for tool integration supported by a diverse Eclipse Ecosystem.

The Eclipse vendor-neutral platform is built on industry standards, which support a wide range of tools, platforms, and languages. The Eclipse Technology is royalty-free and has worldwide redistribution rights. The platform was designed from a clean slate to be extensible and to provide exemplary tools. Eclipse development is based on rules of open source engagements. This includes open, transparent, merit-based, and collaborative development. All individuals can participate and contribute. All plans are developed in the public arena. This platform and the open source development process creates an environment for creativity, originality, and freedom. Eclipse is unparalleled in today's software-tool environment.

The software-tool industry is undergoing massive changes from the commoditization of the technology to the company consolidation. New technology efforts are being redesigned, while a common set of tooling infrastructure is adopted as an industry standard. Successful developers and development paradigms are being challenged to adopt new skills and new, more efficient methods. Old business models are being challenged with free software, and new business models are being developed.

The software-tool industry is deeply appreciative of Eric Clayberg and Dan Rubel for this authoritative book. This book provides the knowledge base so that developers, engineers, and users can learn and use the Eclipse Technology. This enables them to respond to these technology and industry change agents.

Eric and Dan leverage long careers of building software tooling. They each have extensive experience with using Smalltalk for twenty years, Java for thirteen years, and Eclipse for nine years. They have developed extensive vendor and customer relationships that enable them to experience firsthand the necessary elements for building successful software. They are able to combine this direct knowledge of the technology with the experiences of the users to create a book that provides an in-depth description of the process to build commercial-quality Eclipse extensions.

This book provides an introduction and overview to the new developer of the entire process of plug-in development, including all the best practices to achieve high-quality results. This is a reference book for experienced Eclipse developers. It discusses the APIs and demonstrates many samples and examples. Detailed tutorials are provided for both new and experienced developers. Eric and Dan leverage their broad knowledge of user interface (UI) development and present the Eclipse SWT UI. This establishes the building blocks for all Eclipse UI development. These authors articulate the development challenges of building tool software and establish proven in-depth solutions to the problems.

If you are a developer, engineer, or user wishing to build or use Eclipse, this book provides both a foundation and reference. It also provides the intellectual foundation to contribute to the open source Eclipse project and to develop commercial software.

—Skip McGaughey

Foreword

In the 1990s, when Java was in its infancy, learning the Java class libraries involved studying a handful of classes in four or five packages. The Java class libraries have grown in size and complexity, presenting a significant problem to developers wishing to learn Java today. Just like Java, the Eclipse platform has necessarily grown over the years, and therefore considerably more time and effort is required to learn Eclipse 3.4 than its predecessors. One of the principles of the Eclipse platform is that a plug-in should integrate seamlessly with the workbench and with other plug-ins. To achieve seamless integration, it is necessary for plug-in developers to understand the best practices, conventions, and strategies related to building software for Eclipse. *Eclipse Plug-ins* covers everything you need to know to develop Eclipse plug-ins of which you will be proud.

Through the development of a **Favorites** plug-in, the Eclipse Standard Widget Toolkit (SWT) and JFace frameworks are thoroughly discussed, teaching you how to build professional-looking user interfaces such as views, editors, preferences pages, and dialogs. In addition to stock-in-trade subjects, such as user-interface design, lesser-understood Eclipse topics (for example, building features and product branding) are extensively covered, as well as the best discussion I have seen on using Ant to build a product from a single source that targets multiple versions of Eclipse.

Java developers new to Eclipse often have difficulty understanding the extension point mechanism and the critical link between a plug-in's declarative manifest and the Java code necessary to implement a plug-in's functional behavior. This book serves as a roadmap to using the Plug-in Development Environment (PDE) and the extension points defined by the Eclipse platform. It also provides the missing link that developers need to understand the

aspects of a plug-in that should be described in the manifest, how to develop a plug-in using existing extension points, and how to create new extension points to which other developers may further contribute.

When I first saw CodePro, I was both impressed with the productivity gains it brought to Eclipse and the extent to which its plug-ins integrated with the Eclipse platform. Having used CodePro for a while, it has become a part of my development toolkit that I cannot do without. By drawing on their extensive experience gained while developing CodePro, Eric and Dan have done an excellent job of capturing in this book those aspects of plug-in development necessary to create a high-quality and professional-looking Eclipse product.

—Simon Archer

Preface

When we were first exposed to Eclipse back in late 1999, we were struck by the magnitude of the problem IBM was trying to solve. IBM wanted to unify all its development environments on a single code base. At the time, the company was using a mix of technology composed of a hodgepodge of C/C++, Java, and Smalltalk.

Many of IBM's most important tools, including the award-winning VisualAge for Java IDE, were actually written in Smalltalk—a wonderful language for building sophisticated tools, but one that was rapidly losing market share to languages like Java. While IBM had one of the world's largest collections of Smalltalk developers, there wasn't a great deal of industry support for it outside of IBM, and there were very few independent software vendors (ISVs) qualified to create Smalltalk-based add-ons.

Meanwhile, Java was winning the hearts and minds of developers worldwide with its promise of easy portability across a wide range of platforms, while providing the rich application programming interface (API) needed to build the latest generation of Web-based business applications. More important, Java was an object-oriented (OO) language, which meant that IBM could leverage the large body of highly skilled object-oriented developers it had built up over the years of creating Smalltalk-based tools. In fact, IBM took its premiere Object Technology International (OTI) group, which had been responsible for creating IBM's VisualAge Smalltalk and VisualAge Java environments (VisualAge Smalltalk was the first of the VisualAge brand family and VisualAge Java was built using it), and tasked the group with creating a highly extensible integrated development environment (IDE) construction set based in Java. Eclipse was the happy result.

OTI was able to apply its highly evolved OO skills to produce an IDE unmatched in power, flexibility, and extensibility. The group was able to replicate most of the features that had made Smalltalk-based IDEs so popular the decade before, while simultaneously pushing the state of the art in IDE development ahead by an order of magnitude.

The Java world had never seen anything as powerful or as compelling as Eclipse, and it now stands, with Microsoft's .NET, as one of the world's premier development environments. That alone makes Eclipse a perfect platform for developers wishing to get their tools out to as wide an audience as possible. The fact that Eclipse is completely free and open source is icing on the cake. An open, extensible IDE base that is available for free to anyone with a computer is a powerful motivator to the prospective tool developer.

It certainly was to us. At Instantiations and earlier at ObjectShare, we had spent the better part of a decade as entrepreneurs focused on building add-on tools for various IDEs. We had started with building add-ons for Digitalk's Smalltalk/V, migrated to developing tools for IBM's VisualAge Smalltalk, and eventually ended up creating tools for IBM's VisualAge Java (including our award-winning VA Assist product and our jFactor product, one of the world's first Java refactoring tools). Every one of these environments provided a means to extend the IDE, but they were generally not well-documented and certainly not standardized in any way. Small market shares (relative to tools such as VisualBasic) and an eclectic user base also afflicted these environments and, by extension, us.

As an Advanced IBM Business Partner, we were fortunate to have built a long and trusted relationship with the folks at IBM responsible for the creation of Eclipse. That relationship meant that we were in a unique position to be briefed on the technology and to start using it on a daily basis nearly a year and half before the rest of the world even heard about it. When IBM finally announced Eclipse to the world in mid-2001, our team at Instantiations had built some of the first demo applications IBM had to show. Later that year when IBM released its first Eclipse-based commercial tool, WebSphere Studio Application Developer v4.0 (v4.0 so that it synchronized with its then current VisualAge for Java v4.0), our CodePro product became the very first commercial add-on available for it (and for Eclipse in general) on the same day.

Currently, the CodePro product adds hundreds of enhancements to Eclipse and any Eclipse-based IDE. Developing CodePro over the last several years has provided us with an opportunity to learn the details of Eclipse development at a level matched by very few others (with the obvious exception of the IBM and OTI developers, who eat, sleep, and breathe this stuff on a daily basis). CodePro has also served as a testbed for many of the ideas and tech-

niques presented in this book, providing us with a unique perspective from which to write.

Goals of the Book

This book, originally titled *Eclipse: Building Commercial-Quality Plug-ins*, provides an in-depth description of the process involved in building commercial-quality extensions for the Eclipse and the IBM Software Development Platform (SDP)—IBM’s commercial version of Eclipse—development environments. To us, “commercial-quality” is synonymous with “commercial-grade” or “high-quality.” Producing a *commercial-quality* plug-in means going above and beyond the minimal requirements needed to integrate with Eclipse. It means attending to all those details that contribute to the “fit and polish” of a commercial offering.

In the world of Eclipse plug-ins, very few people take the time to really go the extra mile, and most plug-ins fall into the open source, amateur category. For folks interested in producing high-quality plug-ins (which would certainly be the case for any software company wanting to develop Eclipse-based products), there are many details to consider. Our book is meant to encompass the entire process of plug-in development, including all the extra things that need to be done to achieve high-quality results. This book has several complementary goals:

- Provide a quick introduction to using Eclipse for new users
- Provide a reference for experienced Eclipse users wishing to expand their knowledge and improve the quality of their Eclipse-based products
- Provide a detailed tutorial on creating sophisticated Eclipse plug-ins suitable for new and experienced users

The first three chapters introduce the Eclipse development environment and outline the process of building a simple plug-in. The intention of these chapters is to help developers new to Eclipse quickly pull together a plug-in they can use to experiment with.

The first chapter, in particular, introduces the reader to the minimum set of Eclipse tools that he or she will need to build plug-ins. It is a fairly quick overview of the Eclipse IDE and relevant tools (one could write an entire book on that topic alone), and we would expect expert Eclipse users to skip that chapter entirely.

The second chapter introduces the example that we will use throughout most of the book and provides a very quick introduction to building a working plug-in from start to finish. The third chapter presents a high-level overview of the Eclipse architecture and the structure of plug-ins and extension points.

The fourth and fifth chapters cover the Standard Widget Toolkit (SWT) and JFace, which are the building blocks for all Eclipse user interfaces (UIs). These chapters can act as a stand-alone reference; they are intended to provide just enough detail to get you going. Both of these topics are rich enough to warrant entire books and several are currently available.

The subsequent chapters, comprising the bulk of this book, focus on describing each of the various aspects of plug-in development and providing the reader with in-depth knowledge of how to solve the various challenges involved. Each chapter focuses on a different aspect of the problem, and includes an overview, a detailed description, a discussion of challenges and solutions, diagrams, screenshots, cookbook-style code examples, relevant API listings, and a summary.

We have structured the book so that the most important material required for every plug-in project appears in the first half of it. Some of the packaging- and building-oriented material is placed at the end (for example, features and product builds). This organizational scheme left several topics that, while not critical to every plug-in, were important to the creation of commercial-quality plug-ins. These topics have been placed in the second half of the book in an order based on the importance of each and how it related to earlier material. Internationalization, for example, is one of those topics. It isn't critical, and it isn't even all that complicated when you get right down to it. It is, however, important to the book's premise, so we felt it was a topic we needed to include. Since we aren't assuming that the reader is an Eclipse expert (or even a plug-in developer), we have tried to take the reader through each of the important steps in as much detail as possible. While it is true that this is somewhat introductory, it is also an area that most plug-in developers totally ignore and have little or no experience with.

Sometimes a developer needs a quick solution, while at other times that same developer needs to gain in-depth knowledge about a particular aspect of development. The intent is to provide several different ways for the reader to absorb and use the information so that both needs can be addressed. Relevant APIs are included in several of the chapters so that the book can be used as a stand-alone reference during development without requiring the reader to look up those APIs in the IDE. Most API descriptions are copied or paraphrased from the Eclipse platform Javadoc.

As the originators of Eclipse and a major consumer of Eclipse-based technology, IBM is justifiably concerned that new plug-ins meet the same high-quality standards that IBM adheres to. To that end, IBM has established a rigorous *Ready for Rational Software* (RFRS) certification program meant to

ensure the availability of high-quality add-ons to Eclipse and the IBM Software Development Platform. RFRS certification should be one of the ultimate goals for anyone wishing to build and market Eclipse plug-ins. Every chapter covers any relevant RFRS certification criteria and strategies.

The examples provided as part of the chapters describe building various aspects of a concrete Eclipse plug-in that you will see evolve over the course of the book. When used as a reference rather than read cover-to-cover, you will typically start to look in one chapter for issues that are covered in another. To facilitate this type of searching, every chapter contains numerous cross-references to related material that appears in other chapters.

Intended Audience

The audience for this book includes Java tool developers wishing to build products that integrate with Eclipse and other Eclipse-based products, relatively advanced Eclipse users wishing to customize their environments, or anyone who is curious about what makes Eclipse tick. You do not need to be an expert Eclipse user to make use of this book because we introduce most of what you need to know to use Eclipse in Chapter 1, Using Eclipse Tools. While we don't assume any preexisting Eclipse knowledge, we do anticipate that the reader is a fairly seasoned developer with a good grasp of Java and at least a cursory knowledge of extensible markup language (XML).

Conventions Used in This Book

The following formatting conventions are used throughout the book.

Bold—the names of UI elements such as menus, buttons, field labels, tabs, and window titles

Italic—emphasize new terms and Web site addresses

`Courier`—code examples, references to class and method names, and filenames

Courier Bold—emphasize code fragments

“Quoted text”—quotation marks surrounding text indicates words to be entered by the user

What's New in the Third Edition

In this edition, we use the same **Favorites** view example as in the first and second editions, but have reworked much of the content and recreated the code from scratch. Some Eclipse concepts, such as views and editors, are similar but with additional functionality and capabilities; other areas, such as commands, GEF and PDE Build have been added. The following are some of the major changes in this third edition:

Eclipse Command Framework

The Eclipse command framework replaces the older Action framework. Throughout the book, use of the older Action framework has been replaced with new content describing how to accomplish the same thing with the new command framework. This is most obvious in Chapter 6 where the first half of the chapter is entirely devoted to the command framework. Chapter 7 and Chapter 8 also have lots of new material describing use of commands with views and editors.

Eclipse 3.4 and Java 5

All of the screen shots, text and code examples throughout the book have been updated to use the latest Eclipse 3.4 API and Java 5 syntax. Chapter 1 has been overhauled to include descriptions of new capabilities in Eclipse 3.4 including a new overview of using Mylyn and discussion of new preferences. Both Chapter 1 and Chapter 2 cover cool PDE and SWT tools available in Eclipse 3.4.

New GEF Chapter

GEF, the Graphical Editing Framework from Eclipse.org, provides a toolkit for building dynamic interactive graphical user interface elements. Chapter 20 takes you step by step through the process of building a GEF-based view for graphically presenting the relationships between the favorites items and their underlying resources. We then go further, building a GEF-based editor with the ability to add, move, resize, and delete the graphical elements representing those favorites items.

Put PDE Build through its paces

Over the past several years, the PDE build process has been steadily maturing. The proprietary Ant scripts in Chapter 19 of earlier versions of our book have been completely replaced with step-by-step instructions for getting an Eclipse PDE Build process up and running to automate assembly of your product for distribution.

New “p2” update site creation description

“p2” debuts in Eclipse 3.4, replacing the older Update Manager. We take you through the process of using update sites and then building your own in Section 18.3, Update Sites, on page 679.

Acknowledgments

The authors would like to thank all those who have had a hand in putting this book together or who gave us their support and encouragement throughout the many months it took to create.

To our comrades at Instantiations, who gave us the time and encouragement to work on this book: Rick Abbott, Brent Caldwell, Devon Carew, Jim Christensen, Taylor Corey, Dianne Engles, Marta George, Nick Gilman, Seth Hollyman, Mark Johnson, Ed Klimas, Tina Kvavle, Alexander Lobas, Warren Martin, David Masulis, Nancy McClure, Steve Messick, Alexander Mitin, Gina Nebling, John O’Keefe, Keerti Parthasarathy, Phil Quitslund, Mark Russell, Rob Ryan, Andrey Sablin, Balaji Saireddy, Konstantin Scheglov, Chuck Shawan, Bryan Shepherd, Julie Taylor, Mike Taylor, Solveig Viste, Brian Wilkerson, and Jaime Wren.

A special thanks to Jaime Wren, who provided much of the basic research and initial content for Chapter 20.

To our agent, Laura Lewin, and the staff at Studio B, who encouraged us from day one and worked tirelessly on our behalf.

To our editors, Greg Doench and John Neidhart, our production editors, Elizabeth Ryan and Kathleen Caren, our copy editors, Marilyn Rash and Camie Goffi, our editorial assistants, Michelle Housley and Mary Kate Murray, our art director, Sandra Schroeder, our marketing managers, Brandon Prebyski and Beth Wickenhiser, and the staff at Pearson, for their encouragement and tremendous efforts in preparing this book for production.

To Simon Archer and Robert Konigsberg who contributed an unparalleled number of changes and suggestions to various editions of the book, and helped us improve them in almost every dimension.

To Linda Barney who helped us polish and edit the second edition.

To our technical reviewers who helped us enhance the book in many ways: Matt Lavin, Kevin Hammond, Mark Russell, Keerti Parthasarathy, Jaime Wren, Joe Bowbeer, Brian Wilkerson, Joe Winchester, David Whiteman, Boris Pruesmann, Balaji Saireddy, and Raphael Enns.

To the many readers of the first and second editions who contributed errata that have gone into this third edition: Bruce Gruenbaum, Tony Saveski, James Carroll, Tony Weddle, Karen Ploski, Lori Kissell, Brian Vosburgh, Peter Nye, Chris Lott, David Watkins, Simon Archer, Mike Wilkins, Brian Penn, Bernd Essann, Eric Hein, Dave Hewitson, Frank Parrott, Catherine Suess,

William Beebe, Janine Kovack, David Masulis, Jim Norris, Jim Wingard, Roy Johnston, David Karr, Chris Gage, Paul Tamminga, Asim Ullah, and Ebru Aktuna.

To the series editors, Erich Gamma, Lee Nackman, and John Weigand, for their thoughtful comments and for their ongoing efforts to make Eclipse the best development environment in the world.

We would also like to thank our wives, Karen and Kathy, for their endless patience, and our children, Beth, Lauren, Lee, and David, for their endless inspiration.

About the Authors



Eric Clayberg is Senior Vice President for Product Development for Instantiations, Inc. Eric is a seasoned software technologist, product developer, entrepreneur, and manager with more than seventeen years of commercial software development experience, including twelve years of experience with Java and nine years with Eclipse. He is the primary author and architect of more than a dozen commercial Java and Smalltalk add-on products, including the popular WindowBuilder Pro, CodePro, and the award-winning VA Assist product lines. He has a B.S. from MIT, an MBA from Harvard, and has cofounded two successful software companies—ObjectShare and Instantiations.



Dan Rubel is Chief Technology Officer for Instantiations, Inc. He is an entrepreneur and an expert in the design and application of OO technologies with more than fifteen years of commercial software development experience, including thirteen years of experience with Java and nine years with Eclipse. He is the architect and product manager for several successful commercial products, including RCP Developer, WindowTester, jFactor and jKit, and has played key design and leadership roles in other commercial products such as VA Assist, and CodePro. He has a B.S. from Bucknell and is a cofounder of Instantiations.

Instantiations is an Advanced IBM Business Partner and developer of many commercial add-ons for Eclipse and IBM's VisualAge, WebSphere, and Rational product lines. Instantiations is a member of the Eclipse Foundation and a contributor to the Eclipse open source effort with responsibility for the Eclipse Collaboration Tools project known as Koi and for the Eclipse Pollinate project (Beehive).

How to Contact Us

While we have made every effort to make sure that the material in this book is timely and accurate, Eclipse is a rapidly moving target and it is quite possible that you may encounter differences between what we present here and what you experience using Eclipse. The Eclipse UI has evolved considerably over the years, and the latest 3.4 release and upcoming 3.5 release are no exceptions. While we have targeted it at Eclipse 3.4 and used it for all of our examples, this book was completed after Eclipse 3.4 was finished and during the initial phases of development of Eclipse 3.5. If you are using an older or newer version of Eclipse, this means that you may encounter various views, dialogs, and wizards that are subtly different from the screenshots herein.

- Questions about the book's technical content should be addressed to:
info@qualityeclipse.com
- Sales questions should be addressed to Addison-Wesley at:
www.informit.com/store/sales.aspx
- Source code for the projects presented can be found at:
www.qualityeclipse.com/projects
- Errata can be found at:
www.qualityeclipse.com/errata
- Tools used and described can be found at:
www.qualityeclipse.com/tools



CHAPTER 2

A Simple Plug-in Example

Before covering the Eclipse infrastructure (see Chapter 3) and each area of plug-in construction in-depth, it is useful to create a simple plug-in on which discussion and examples can be based. This chapter takes a step-by-step approach to creating a simple but fully operational plug-in that will be enhanced bit-by-bit during the course of this book. This process provides valuable firsthand experience using the Eclipse IDE and touches on every aspect of building and maintaining a plug-in.

2.1 The Favorites Plug-in

The **Favorites** plug-in, which you'll build over the course of this book, displays a list of resources, lets you add and remove resources from the list, easily opens an editor on a selected resource, updates the list automatically as a result of events elsewhere in the system, and more. Subsequent chapters discuss aspects of plug-in development in terms of enhancements to the **Favorites** plug-in.

This chapter starts the process by covering the creation of the **Favorites** plug-in in its simplest form using the following steps:

- Creating a plug-in project
- Reviewing the generated code
- Building a product
- Installing and running the product

2.2 Creating a Plug-in Project

The first step is to create a plug-in project using the Eclipse **New Project** wizard. In addition to creating a new project, this wizard has a number of different code generation options, such as views, editors, and actions, for creating sample plug-in code. To keep things simple and focus only on the essentials of plug-in creation, select the **Plug-in with a view** option, which is discussed in the next subsection.

2.2.1 New Plug-in Project wizard

From the **File** menu, select **New > Project** to launch the **New Project** wizard (see Figure 2–1). On this first page of the wizard, select **Plug-in Project** from the list and then click the **Next** button.

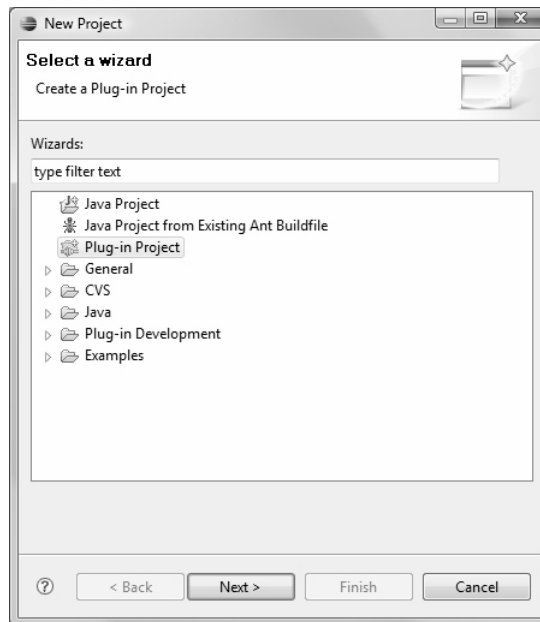


Figure 2–1 New Project wizard page 1—selecting a project type.

On the next page of the wizard (see Figure 2–2), enter the name of the project; in this case, it's `com.qualityeclipse.favorites`, which is the same as the Favorites plug-in identifier. Chapter 3, *Eclipse Infrastructure*, discusses

plug-in identifiers and other aspects of plug-in architecture in more detail. Fill in the other fields as shown and then click the **Next** button.

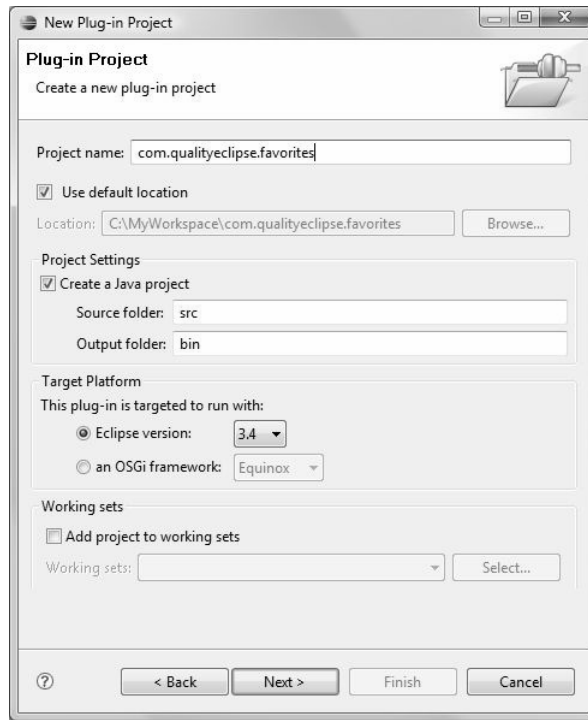


Figure 2–2 New Project wizard page 2—naming the project..

Tip: A project can be named anything, but it is easier to name it the same as the plug-in identifier. By convention, this is the plug-in project-naming scheme that the Eclipse organization uses for most of its work. Because of this, the **New Project** wizard assumes that the project name and the plug-in identifier are the same.

2.2.2 Define the plug-in

Every plug-in has a `META-INF/MANIFEST.MF` file. In addition, it may contain a `plugin.xml` file and/or a Java class that represents the plug-in programmatically. The next wizard page displays options for generating both the plug-in manifest and plug-in Java class. Supply the **Plug-in ID**, **Plug-in Version**, **Plug-in Name** and more for the plug-in as shown in Figure 2–3 then click the **Next** button.



Figure 2–3 New Project wizard page 3—describing the plug-in.

Next, the **New Plug-in Project** wizard next displays the various plug-in pieces that can be automatically generated by the wizard (see Figure 2–4). There are many different options on this page for generating quite a bit of sample code. It is useful to try out each option and review the code that is generated; however for this example, select **Plug-in with a view** and then click the **Next** button.

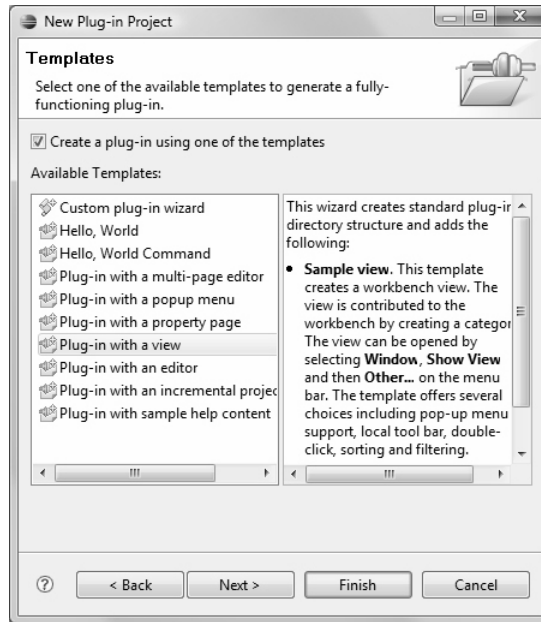


Figure 2-4 New Plug-in Project wizard page 4—selecting a plug-in type.

2.2.3 Define the view

Selecting view code generation options is the next step in this process. Enter the values for this page (see Figure 2-5), uncheck the **Add the view to the resource perspective** and **Add context help to the view** (Eclipse 3.4 only) checkboxes to simplify the generated plug-in manifest file.

If you are in Eclipse 3.3, then click the **Next** button and uncheck each of the code generation options (see Figure 2-6). Each of these checkboxes represents code that could be generated as part of the **Favorites** view. These are covered in subsequent chapters. This wizard page has been removed in Eclipse 3.4 and thus the `FavoritesView` class generated by the wizard will contain more code than is shown in the book (see Section 2.3.3, The Favorites view, on page 84).

When you click the **Finish** button, the new plug-in project is created and the plug-in manifest editor is automatically opened (see Figure 2-9).

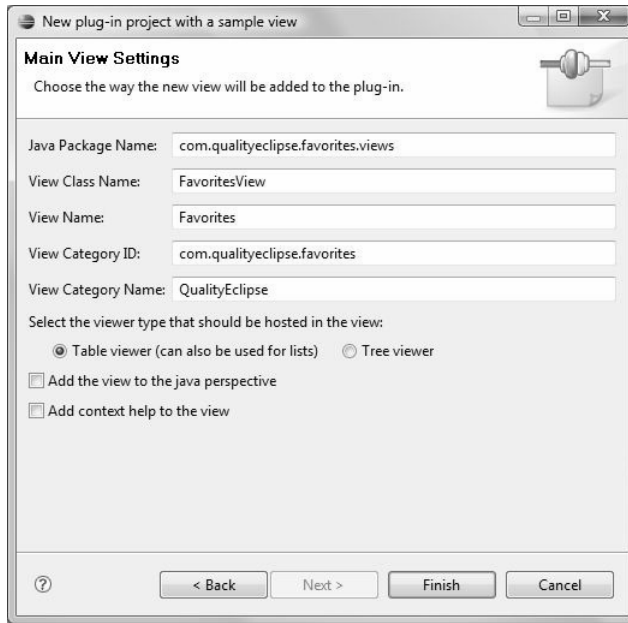


Figure 2–5 New Plug-in Project wizard page 5—defining the view.

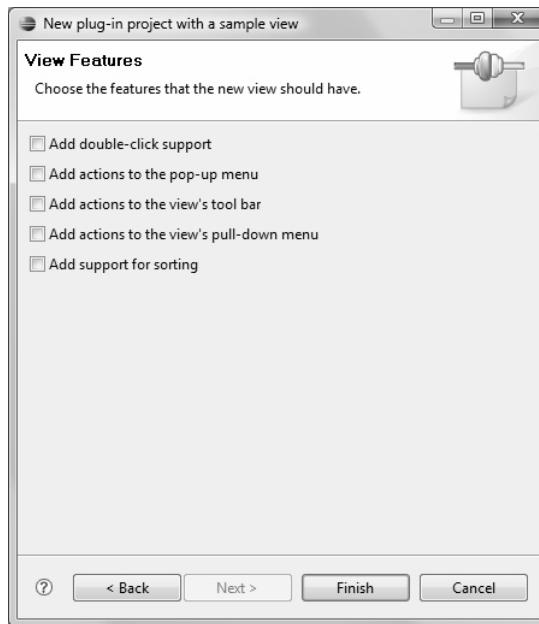


Figure 2–6 New Plug-in Project wizard page 6—code generation options for the view (Eclipse 3.3 only)

2.3 Reviewing the Generated Code

Reviewing the code generated by the **New Plug-in Project** wizard provides a brief look at the following major parts comprising the sample plug-in.

- The plug-in manifests
- The plug-in class
- The Favorites view

2.3.1 The Plug-in manifests

The plug-in manifest editor shows the contents of the two plug-in manifest files, `META-INF/MANIFEST.MF` and `plugin.xml`, which define how this plug-in relates to all the others in the system. This editor is automatically opened to its first page (see Figure 2–9) as a result of creating a new plug-in project. If the plug-in manifest editor is closed, double-clicking on either the `META-INF/MANIFEST.MF` or the `plugin.xml` file reopens the editor. The following is an overview of the manifest editor, while more detail on the plug-in manifest itself can be found in Chapter 3.

Although the editor is a convenient way to modify the plug-in’s description, it’s still useful to peek at the source behind the scenes to see how the editor’s different parts relate to the underlying code. Click the **MANIFEST.MF** tab to display the source of the `META-INF/MANIFEST.MF` file that defines the runtime aspects of this plug-in (see Figure 2–7). The first two lines define it as an OSGi manifest file (see Section 3.3, Plug-in Manifest, on page 113). Subsequent lines specify plug-in name, version, identifier, classpath, and plug-ins on which this plug-in depends. All these aspects are editable using other pages in the plug-in manifest editor.

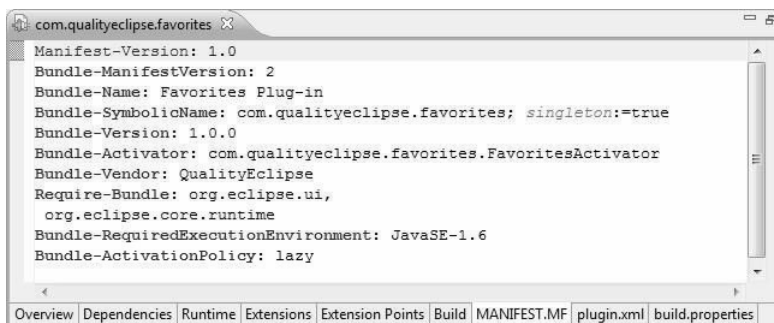


Figure 2–7 Plug-in manifest editor MANIFEST.MF page.

New in Eclipse 3.4 The `Eclipse-LazyStart: true` directive in the `MANIFEST.MF` file has been replaced with `Bundle-ActivationPolicy: lazy`. Both directives have the same semantics; only the name has changed.

Clicking on the **plugin.xml** tab of the editor displays the `plugin.xml` file that defines the extension aspects of this plug-in (see Figure 2–8). The first line declares this to be an XML file, while subsequent lines specify plug-in extensions.

```

<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.2"?>
<plugin>
  <extension
    point="org.eclipse.ui.views"
  >
    <category
      name="QualityEclipse"
      id="com.qualityeclipse.favorites"
    >
      </category>
    <view
      name="Favorites"
      icon="icons/sample.gif"
      category="com.qualityeclipse.favorites"
      class="com.qualityeclipse.favorites.views.FavoritesView"
      id="com.qualityeclipse.favorites.views.FavoritesView"
    >
      </view>
    </extension>
  </plugin>

```

Figure 2–8 Plug-in manifest editor `plugin.xml` page.

Overview

This section describes general information about this plug-in.

General Information

ID:

Version:

Name:

Provider:

Platform Filter:

Activator:

Activate this plug-in when one of its classes is loaded

This plug-in is a singleton

Plug-in Content

The content of the plug-in is made up of two sections:

- Dependencies:** lists all the plug-ins required on this plug-in's classpath to compile and run.
- Runtime:** lists the libraries that make up this plug-in's runtime.

Extension / Extension Point Content

This plug-in may define extensions and extension points:

- Extensions:** declares contributions this plug-in makes to the platform.
- Extension Points:** declares new function points this plug-in adds to the platform.

Testing

Test this plug-in by launching a separate Eclipse application:

- [Launch an Eclipse application](#)
- [Launch an Eclipse application in Debug mode](#)

Overview | Dependencies | Runtime | Extensions | Extension Points | Build | MANIFEST.MF | plugin.xml | build.properties

Figure 2–9 Plug-in manifest editor Overview page.
The **This plug-in is a singleton** option was added in Eclipse 3.4.

The **Overview** page of the manifest editor shows a summary of the plug-in manifest (see Figure 2–9). The section on this page describing general information, such as the plug-in identifier (ID), version, name, class, and provider, corresponds to the first chunk of source in the `META-INF/MANIFEST.MF` file:

```
Bundle-Name: Favorites Plug-in
Bundle-SymbolicName: com.qualityeclipse.favorites; singleton:=true
Bundle-Version: 1.0.0
Bundle-Activator: com.qualityeclipse.favorites.FavoritesActivator
Bundle-Vendor: QualityEclipse
Bundle-RequiredExecutionEnvironment: JavaSE-1.5
```

You can edit the information on the **Overview** page or switch to the **MANIFEST.MF** page and edit the source directly.

Tip: Making changes to any page other than the `plugin.xml` and `MANIFEST.MF` pages may cause the manifest editor to reformat the source. If you are particular about the formatting of either manifest file, then either use only the `plugin.xml` and `MANIFEST.MF` pages to perform editing or use another editor.

Caution: The formatting rules of `META-INF/MANIFEST.MF` include some quite nonintuitive rules related to line length and line wrapping. Edit `plugin.xml` with care, and `META-INF/MANIFEST.MF` with caution!

The reliance of this plug-in on other plug-ins in the system appears on the **Dependencies** page of the plug-in manifest editor (see Figure 2–10).

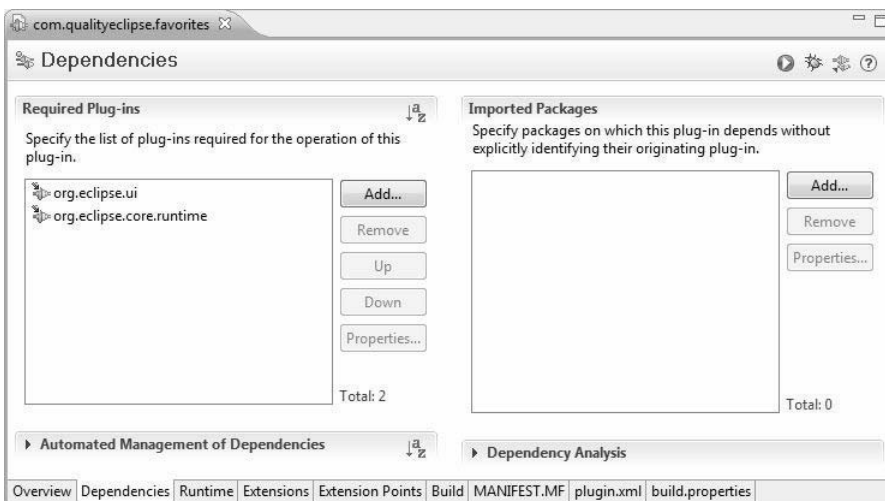


Figure 2–10 Plug-in manifest editor Dependencies page.

This corresponds to the `Require-Bundle` chunk of source in the `META-INF/MANIFEST.MF` file:

```
Require-Bundle: org.eclipse.ui,  
               org.eclipse.core.runtime
```

For the **Favorites** plug-in, this section indicates a dependency on the `org.eclipse.core.runtime` and `org.eclipse.ui` plug-ins. This dependency declaration differs from the **Favorites** project's Java build path (also known as the compile-time classpath) because the Java build path is a compile-time artifact, while the plug-in dependency declaration comes into play during plug-in execution. Because the project was created as a plug-in project and has the `org.eclipse.pde.PluginNature` nature (see Section 14.3, *Natures*, on page 561 for more on project natures), any changes to this dependency list will automatically be reflected in the Java build path, but not the reverse. If these two aspects of your plug-in get out of sync, then you can have a plug-in that compiles and builds but does not execute properly.

Tip: Edit this dependency list rather than the Java build path so that the two are automatically always in sync.

Alternatively, the dependencies could have been expressed as **Imported Packages** on the **Dependencies** page of the manifest editor (see Figure 2–10 and the end of Section 3.3.3, *Plug-in dependencies*, on page 116). This would correspond to an `Import-Package` chunk of source in the `META-INF/MANIFEST.MF` file looking something like this:

```
Import-Package: org.eclipse.ui.views,  
               org.eclipse.core.runtime.model
```

The **Runtime** page of the manifest editor (see Figure 2–11) corresponds to the `Bundle-ClassPath` chunk of source in the `META-INF/MANIFEST.MF` file, which defines what libraries are delivered with the plug-in and used by the plug-in during execution, what package prefixes are used within each library (used to speed up plug-in loading time), and whether other plug-ins can reference the code in the library (see Section 21.2.5, *Related plug-ins*, on page 783 for more on package visibility). For the **Favorites** plug-in, all the code is contained in the `com.qualityeclipse.favorites_1.0.0.jar` itself, so no `Bundle-ClassPath` declaration is necessary.

The **Favorites** plug-in does not export any packages for other plug-ins to use or extend.

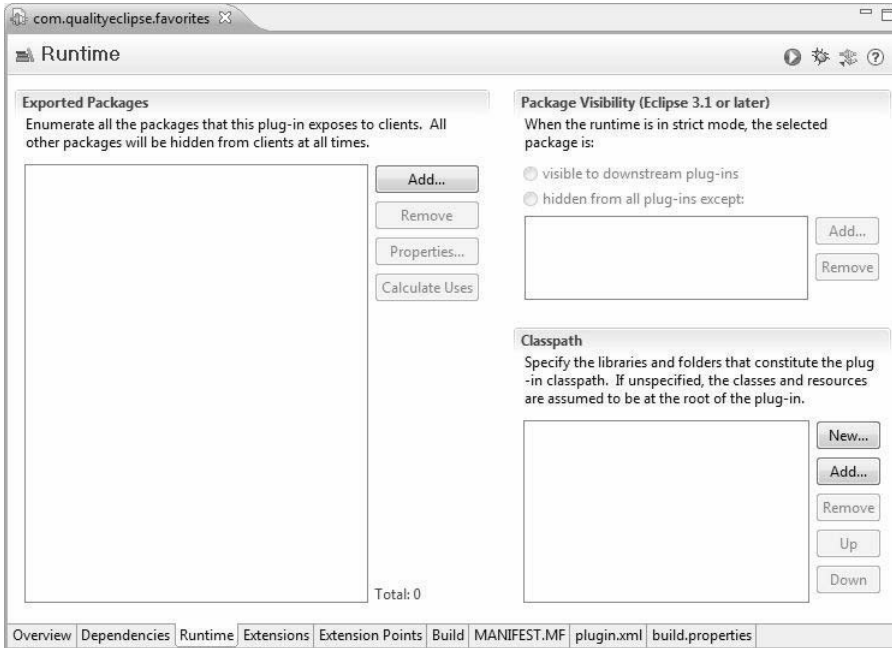


Figure 2-11 Plug-in manifest editor Runtime page.

The **Extensions** page (see Figure 2-12) displays how this plug-in augments the functionality already provided by other plug-ins in the system, and corresponds to the `<extension point="org.eclipse.ui.views">` chunk of XML in the `plugin.xml` file:

```
<extension
  point="org.eclipse.ui.views">
  <category
    name="QualityEclipse"
    id="com.qualityeclipse.favorites">
  </category>
  <view
    name="Favorites"
    icon="icons/sample.gif"
    category="com.qualityeclipse.favorites"
    class="com.qualityeclipse.favorites.views.FavoritesView"
    id="com.qualityeclipse.favorites.views.FavoritesView">
  </view>
</extension>
```

The **Favorites** plug-in declares an extension to the `org.eclipse.ui` plug-in using the `org.eclipse.ui.views` extension point by providing an additional category of views named **QualityEclipse** and a new view in that cat-

egory named **Favorites**. Selecting an item in the tree on the left in the **Extensions** page causes the properties for that item to appear on the right. In this case, selecting **Favorites (view)** on the **Extensions** page displays the name, identifier, class, and more information about the **Favorites** view that is being declared. This corresponds to the XML attributes defined in the `<view>` chunk of XML shown previously.

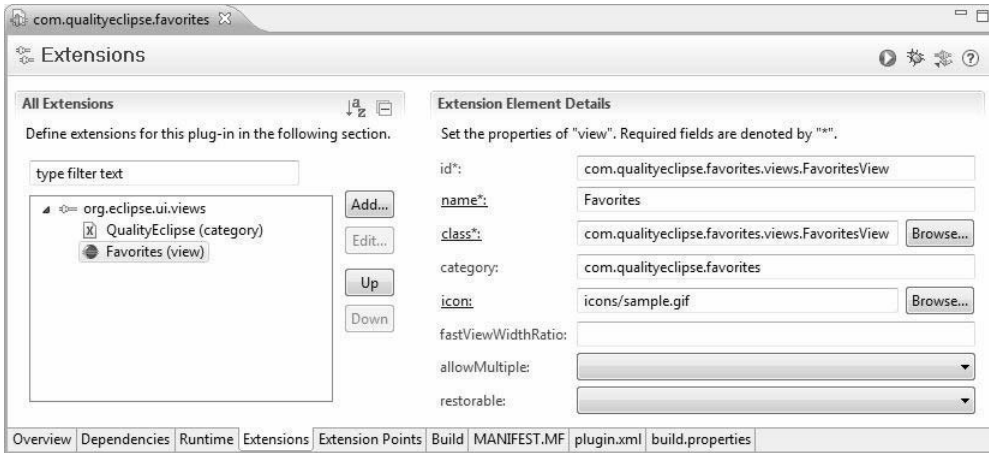


Figure 2–12 Plug-in manifest editor Extensions page.

Finally, the **Extension Points** page of the manifest editor (see Figure 2–13) facilitates the definition of new extension points so that other plug-ins can augment the functionality provided by this plug-in. At this time, the **Favorites** plug-in doesn't define any extension points and therefore cannot be augmented by other plug-ins (see Section 17.2, *Defining an Extension Point*, on page 639).

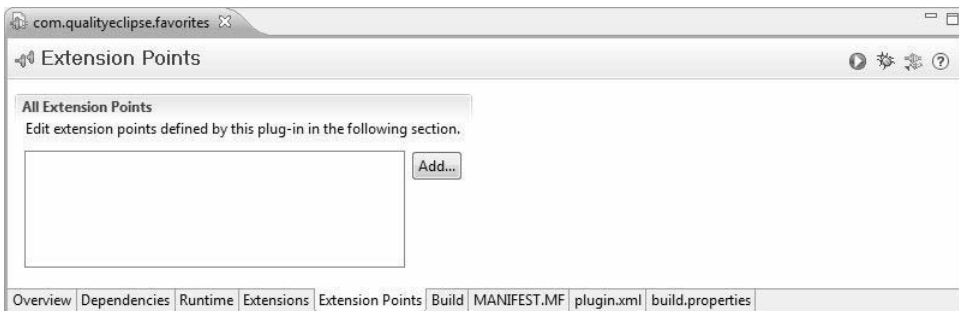


Figure 2–13 Plug-in manifest editor Extension Points page.

2.3.2 The Activator or Plug-in class

Every plug-in optionally can declare a class that represents the plug-in from a programmatic standpoint as displayed on the manifest editor's **Overview** page (see Figure 2–9). This class is referred to as an **Activator** (or, in earlier versions of Eclipse, a **Plug-in class**). In the **Favorites** plug-in, this class is named `com.qualityeclipse.favorites.FavoritesActivator`.

```
package com.qualityeclipse.favorites;

import org.eclipse.jface.resource.ImageDescriptor;
import org.eclipse.ui.plugin.AbstractUIPlugin;
import org.osgi.framework.BundleContext;

/**
 * The activator class controls the plug-in life cycle
 */
public class FavoritesActivator extends AbstractUIPlugin {

    // The plug-in ID
    public static final String PLUGIN_ID
        = "com.qualityeclipse.favorites";

    // The shared instance
    private static FavoritesActivator plugin;

    /**
     * The constructor
     */
    public FavoritesActivator() {
    }

    /**
     * This method is called upon plug-in activation.
     */
    public void start(BundleContext context) throws Exception {
        super.start(context);
        plugin = this;
    }

    /**
     * This method is called when the plug-in is stopped.
     */
    public void stop(BundleContext context) throws Exception {
        plugin = null;
        super.stop(context);
    }

    /**
     * Returns the shared instance
     */
    public static FavoritesActivator getDefault() {
        return plugin;
    }
}
```

```
/**
 * Returns an image descriptor for the image file at the given
 * plug-in relative path
 *
 * @param path the path
 * @return the image descriptor
 */
public static ImageDescriptor getImageDescriptor(String path) {
    return imageDescriptorFromPlugin(PLUGIN_ID, path);
}
}
```

If the `Bundle-ActivationPolicy` in the `META-INF/MANIFEST.MF` file is `lazy`, then when the plug-in is activated, the Eclipse system instantiates the activator class before loading any other classes in it. This corresponds to the “Activate this plug-in when one of its classes is loaded.” checkbox on the **Overview** page of the manifest editor (see Figure 2–9). This single activator class instance is used by the Eclipse system throughout the life of the plug-in and no other instance is created.

Tip: For more background on `Bundle-ActivationPolicy`, see http://wiki.eclipse.org/Lazy_Start_Bundles.

Typically, activator classes declare a static field to reference this singleton so that it can be easily shared throughout the plug-in as needed. In this case, the **Favorites** plug-in defines a field named `plugin` that is assigned in the `start` method and accessed using the `getDefault` method.

Tip: The Eclipse system always instantiates exactly one instance of an active plug-in’s `Activator` class. Do not create instances of this class yourself.

2.3.3 The Favorites view

In addition to the plug-in manifest and plug-in class, the **New Plug-in Project** wizard generated code for a simple view (in the following sample) called **Favorites**. At this point, the view creates and displays information from a sample model; in subsequent chapters, however, this view will be hooked up to a favorites model and will display information from the favorites items contained within that model. Eclipse 3.4 generates additional code unnecessary for this exercise, so adjust the generated code to appear as shown below.

```
package com.qualityeclipse.favorites.views;

import org.eclipse.swt.widgets.Composite;
import org.eclipse.ui.part.*;
import org.eclipse.jface.viewers.*;
import org.eclipse.swt.graphics.Image;
import org.eclipse.jface.action.*;
import org.eclipse.jface.dialogs.MessageDialog;
import org.eclipse.ui.*;
import org.eclipse.swt.widgets.Menu;
import org.eclipse.swt.SWT;

/**
 * This sample class demonstrates how to plug-in a new workbench
 * view. The view shows data obtained from the model. The sample
 * creates a dummy model on the fly, but a real implementation
 * would connect to the model available either in this or another
 * plug-in (e.g., the workspace). The view is connected to the
 * model using a content provider.
 * <p>
 * The view uses a label provider to define how model objects
 * should be presented in the view. Each view can present the
 * same model objects using different labels and icons, if
 * needed. Alternatively, a single label provider can be shared
 * between views in order to ensure that objects of the same type
 * are presented in the same way everywhere.
 * <p>
 */
public class FavoritesView extends ViewPart {
    private TableViewer viewer;

    /**
     * The content provider class is responsible for providing
     * objects to the view. It can wrap existing objects in
     * adapters or simply return objects as-is. These objects may
     * be sensitive to the current input of the view, or ignore it
     * and always show the same content (Task List, for
     * example).
     */

    class ViewContentProvider
        implements IStructuredContentProvider
    {
        public void inputChanged(
            Viewer v, Object oldInput, Object newInput) {
        }

        public void dispose() {
        }

        public Object[] getElements(Object parent) {
            return new String[] { "One", "Two", "Three" };
        }
    }
}
```

```
/*
 * The label provider class is responsible for translating
 * objects into text and images that are displayed
 * in the various cells of the table.
 */

class ViewLabelProvider extends LabelProvider
    implements ITableLabelProvider
{
    public String getColumnText(Object obj, int index) {
        return getText(obj);
    }

    public Image getColumnImage(Object obj, int index) {
        return getImage(obj);
    }

    public Image getImage(Object obj) {
        return PlatformUI.getWorkbench().getSharedImages()
            .getImage(ISharedImages.IMG_OBJ_ELEMENT);
    }
}

/**
 * The constructor.
 */
public FavoritesView() {

}

/**
 * This is a callback that will allow us to create the viewer
 * and initialize it.
 */

public void createPartControl(Composite parent) {
    viewer = new TableViewer(
        parent, SWT.MULTI | SWT.H_SCROLL | SWT.V_SCROLL);
    viewer.setContentProvider(new ViewContentProvider());
    viewer.setLabelProvider(new ViewLabelProvider());
    viewer.setInput(getViewSite());
}

/**
 * Passing the focus request to the viewer's control.
 */
public void setFocus() {
    viewer.getControl().setFocus();
}
}
```

2.4 Building a Product

Building a product involves packaging up only those elements to be delivered in a form that the customer can install into his or her environment. You can

build the product in several different ways, including manually or by using a Windows batch script, a UNIX shell script, or an Apache Ant script. You can deliver the end product as a single compressed file or as a stand-alone executable. For our purposes, the **Favorites** plug-in will be delivered with source code as a single compressed zip file.

2.4.1 Building manually

Building a product manually involves launching an Eclipse **Export** wizard, filling out a few fields, and clicking the **Finish** button. Select the **File > Export** command to launch the desired export wizard. On the first wizard page (see Figure 2–14), select **Deployable plug-ins and fragments** and then click the **Next** button.

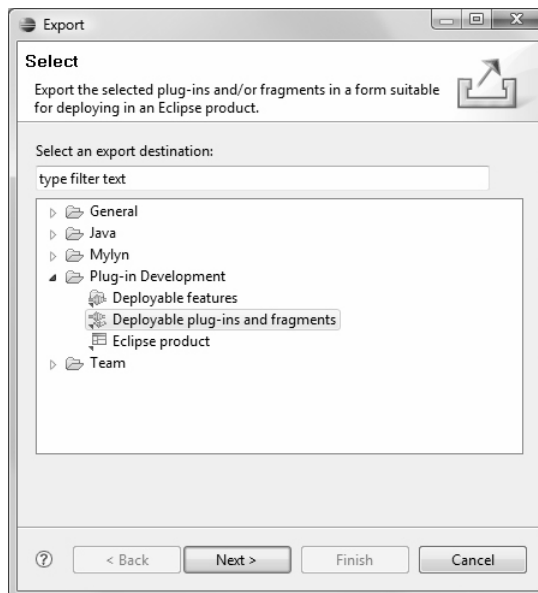


Figure 2–14 Export wizard page 1—choosing the type of export.

On the second page of the **Export** wizard (see Figure 2–15), select the plug-ins to be exported, enter the name of the zip file to contain the result, and select the options shown. In addition, specify that this export operation be saved as an Ant script in a file named `build-favorites.xml` in the `com.qualityeclipse.favorites` project and check the **Include source code** option, then click **Finish**.

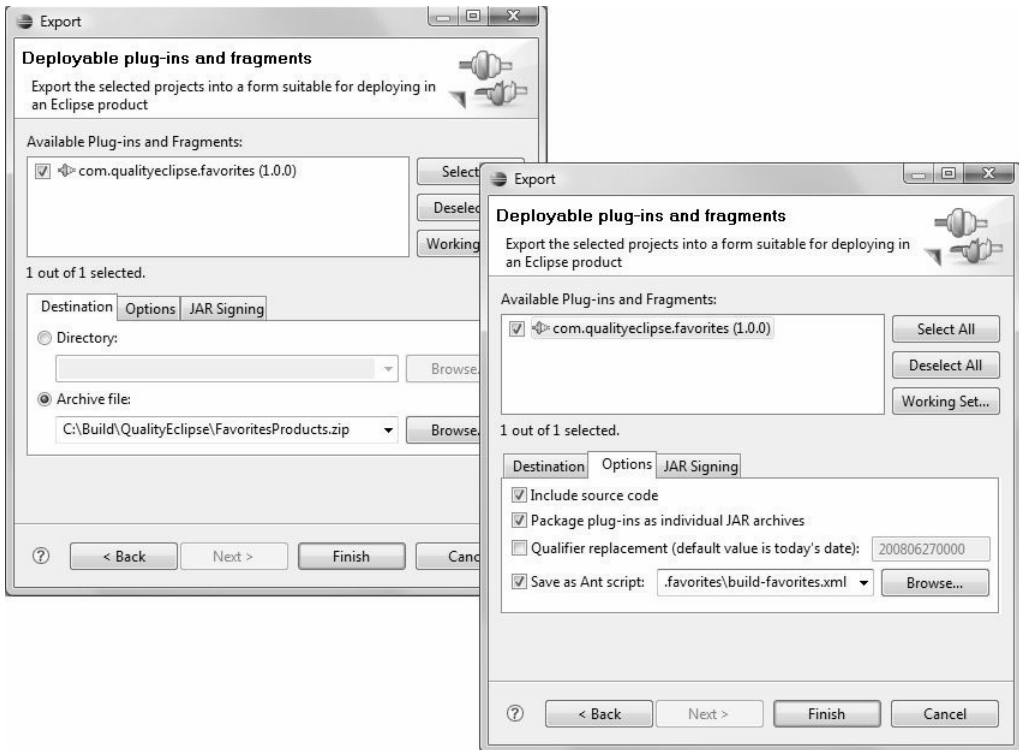


Figure 2–15 Export wizard page 2—specifying the zip file’s contents.

The created zip file contains a single plug-in JAR file (a plug-in can be deployed as a single JAR file as of Eclipse 3.1):

```
plugins/com.qualityeclipse.favorites_1.0.0.jar
```

And that plug-in JAR file contains the plug-in as specified in the Export wizard:

```
com.qualityeclipse.favorites classes
com.qualityeclipse.favorites source files
plugin.xml
icons/sample.gif
META-INF/MANIFEST.MF
```

Unfortunately, this process is manual, and therefore prone to errors. Manually building a product is fine once or twice, but what if a different person in the company needs to build the product? What happens as the product grows and encompasses more plug-ins? A product needs a repeatable and reliable method for building it.

2.4.2 Building with Apache Ant

An Apache Ant script provides a reliable, flexible, and repeatable process for building a plug-in project. There is a little more up-front work to set up an Ant script, but it is much less error-prone over time than building a product manually. For more information about Ant and constructing more complex build scripts, see Chapter 19.

Eclipse can generate simple Ant scripts. The prior section specified that the **Export** wizard generates an Ant script file named `build-favorites.xml` in the `com.qualityeclipse.favorites` project:

```
<?xml version="1.0" encoding="UTF-8"?>
<project default="plugin_export" name="build">
  <target name="plugin_export">
    <pde.exportPlugins
      destination="C:\Build\QualityEclipse"
      exportSource="true"
      exportType="zip"
      filename="FavoritesProduct.zip"
      plugins="com.qualityeclipse.favorites"
      useJARFormat="true" />
  </target>
</project>
```

The preceding simple script works well from the Eclipse UI; however, unfortunately, the `pde.exportPlugins` and other `pde.export*` tasks are asynchronous and cannot be used in a headless environment (see Bugzilla entry 58413 at bugs.eclipse.org/bugs/show_bug.cgi?id=58413) making it difficult to build more than simple scripts.

If you want your build script to do more than just export plug-ins (see Section 3.2.1, Link files, on page 111), then you'll need a more complex Ant script similar to the following. For more on Ant and build scripts, see Chapter 19, Building a Product.

```

<?xml version="1.0" encoding="UTF-8"?>
<project default="plugin_export" name="build">
  <target name="plugin_export">

    <!-- Define build directories -->
    <property name="build.root"
      location="/Build/QualityEclipse" />
    <property name="build.temp"
      location="${build.root}/temp" />
    <property name="build.out"
      location="${build.root}/product" />

    <!-- Create build directories -->
    <delete dir="${build.temp}" />
    <mkdir dir="${build.temp}" />
    <mkdir dir="${build.out}" />

    <!-- Read the MANIFEST.MF -->
    <copy file="META-INF/MANIFEST.MF" todir="${build.temp}" />
    <replace file="${build.temp}/MANIFEST.MF">
      <replacefilter token=":" value="=" />
      <replacefilter token=":" value="=" />
      <replacetoken>;</replacetoken>
      <replacevalue>
        </replacevalue>
    </replace>
    <property file="${build.temp}/MANIFEST.MF"/>

    <!-- Plugin locations -->
    <property name="plugin.jarname" value=
      "com.qualityeclipse.favorites_${Bundle-Version}" />
    <property name="plugin.jar" location=
      "${build.temp}/jars/plugins/${plugin.jarname}.jar" />
    <property name="product.zip" value=
      "${build.out}/Favorites_v${Bundle-Version}.zip" />

    <!-- Assemble plug-in JAR -->
    <mkdir dir="${build.temp}/jars/plugins" />
    <zip destfile="${plugin.jar}">
      <zipfileset dir="bin" />
      <zipfileset dir="." includes="META-INF/MANIFEST.MF" />
      <zipfileset dir="." includes="plugin.xml" />
      <zipfileset dir="." includes="icons/*.gif" />
      <zipfileset dir="." includes="src/**/*" />
    </zip>

    <!-- Assemble the product zip -->
    <zip destfile="${product.zip}">
      <fileset dir="${build.temp}/jars" />
    </zip>

  </target>
</project>

```

To execute this Ant script, right-click on the `build-favorites.xml` file and select **Run Ant...** (see Figure 2–16). When the Ant wizard appears, click on the **JRE** tab and select the **Run in the same JRE as the workspace** option (see Figure 2–17). Click the **Run** button to build the product.

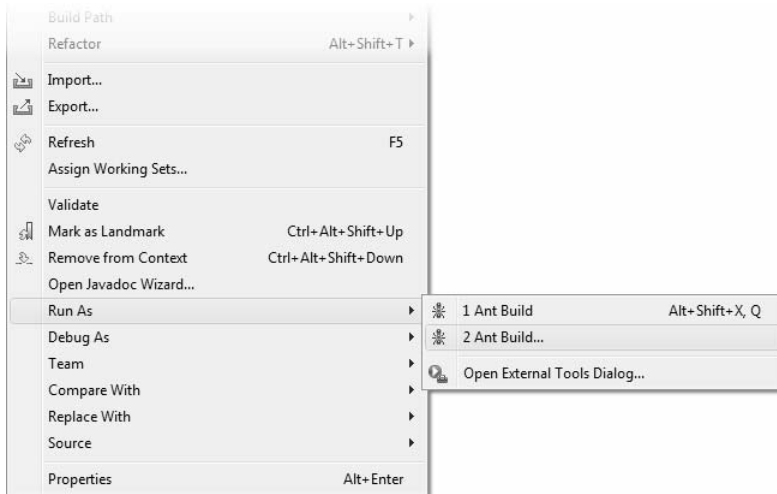


Figure 2–16 The build.xml popup context menu.

Tip: If your Ant script uses Eclipse-specific Ant tasks, such as `pde.exportPlugins`, then you must select the **Run in the same JRE as the workspace** option for your Ant script to execute properly.

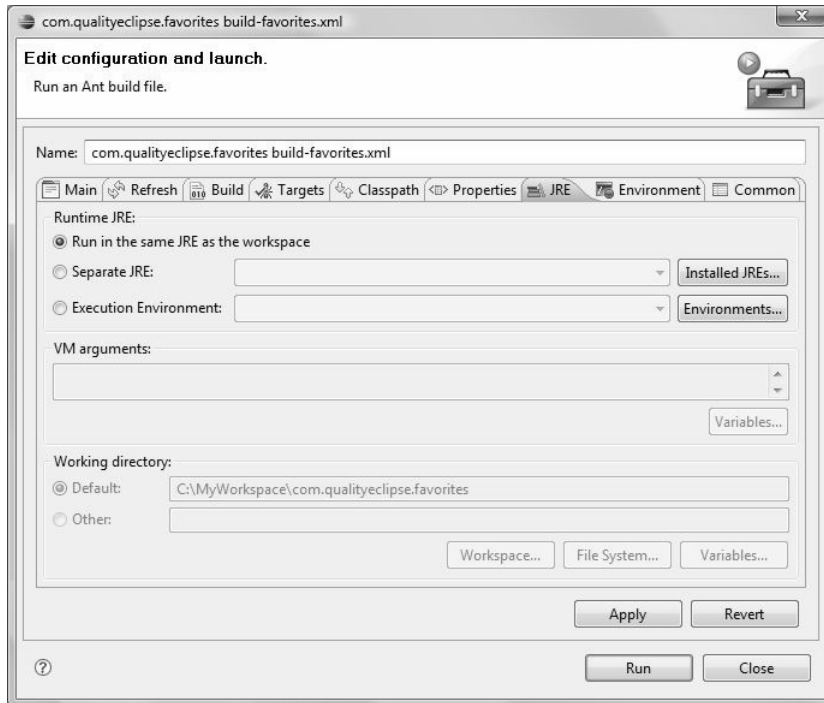


Figure 2-17 The Ant wizard.

2.5 Installing and Running the Product

To install the Favorites plug-in, do the following.

- Shut down Eclipse
- Unzip the `Favorites_v1.0.0.zip` file into your Eclipse directory (e.g., `C:/eclipse`)
- Verify that the favorites plug-in is in the `/plugins` directory (e.g., `C:/eclipse/plugins/com.qualityeclipse.favorites_1.0.0.jar`)
- Restart Eclipse

Tip: Eclipse caches plug-in information in a configuration directory (see Section 3.4.5, Plug-in configuration files, on page 123). If you are installing a new version of your plug-in over an already installed one without incrementing the version number, then use the `-clean` command-line option when launching Eclipse so that it will rebuild its cached plug-in information.

After Eclipse has restarted, from the **Window** menu, select **Show View > Other...** (see Figure 2–18) to open the **Show View** dialog (see Figure 2–19). In the dialog, expand the **Quality Eclipse** category, select **Favorites**, and then click the **OK** button. This causes the **Favorites** view to open (see Figure 2–20).

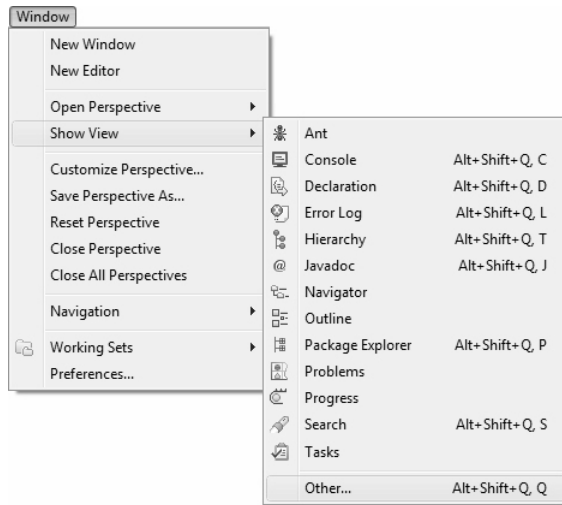


Figure 2–18 Show View > Other... from the Window menu.

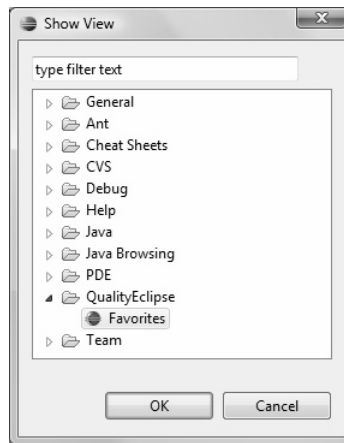


Figure 2–19 Show View dialog.



Figure 2–20 The Favorites view in its initial and simplest form.

2.6 Debugging the Product

Inevitably, during the course of producing a product, you'll need to debug a problem or you'll simply have to gain a better understanding of the code through a means more enlightening than just reviewing the source code. You can use the **Runtime Workbench** to determine exactly what happens during product execution so that you can solve problems.

2.6.1 Creating a configuration

The first step in this process is to create a configuration in which the product can be debugged. Start by selecting **Debug Configurations...** in the **Debug** toolbar menu (see Figure 2–21).

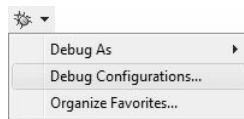


Figure 2–21 Debug menu.

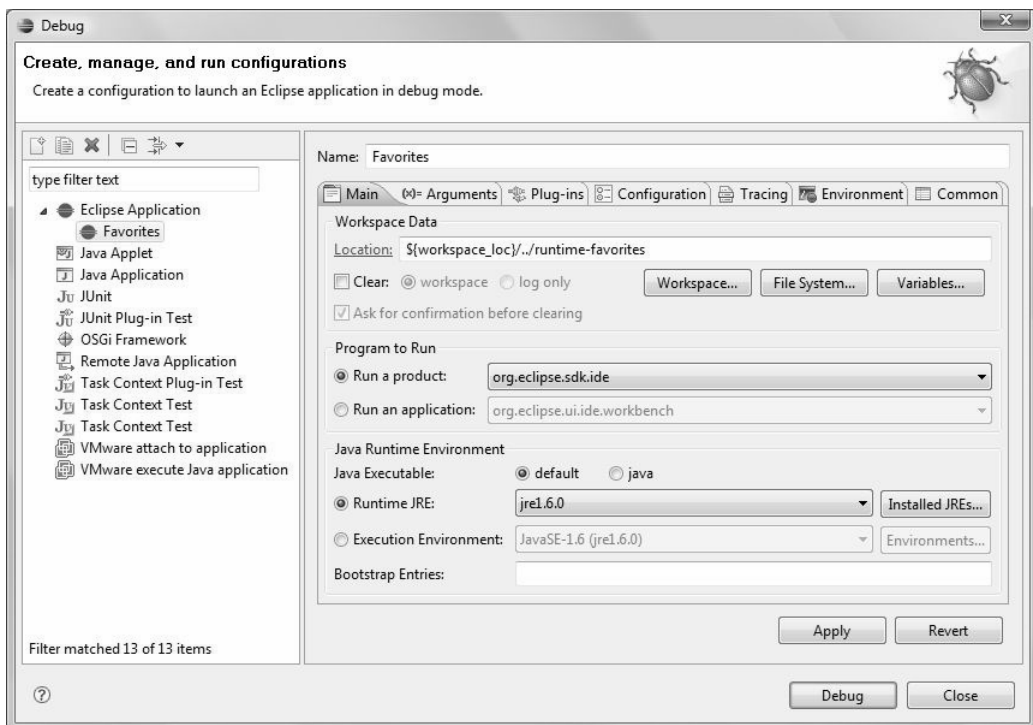



Figure 2–22 Defining a new configuration.

In the dialog that appears (see Figure 2–22), select **Eclipse Application** and then click the **New**  button. Next, enter “Favorites” as the name of the configuration.

2.6.2 Selecting plug-ins and fragments

After the preceding, select the **Plug-ins** tab and **plug-ins selected below only** in the **Launch with** combo box (see Figure 2–23). In the list of plug-ins, make sure that the **Favorites** plug-in is selected in the **Workspace Plug-ins** category but not in the **External Plug-ins** category.

Tip: Plug-in projects specified in the configuration take precedence over plug-ins installed in Eclipse itself. If you have a plug-in project with the same identifier as a plug-in installed in Eclipse and want to use the installed plug-in in the **Runtime Workbench** rather than the plug-in project, uncheck the plug-in project in the **Workspace Plug-ins** category and check the installed plug-in in the **External Plug-ins** category.

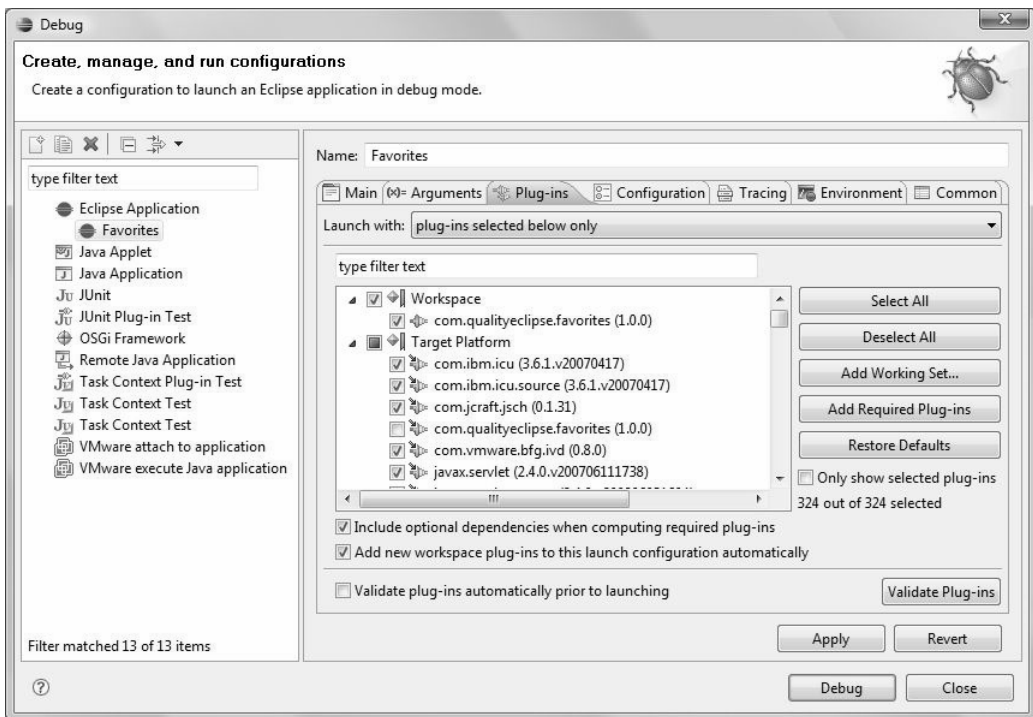


Figure 2–23 Selecting plug-ins in the configuration.

2.6.3 Launching the Runtime Workbench

Click the **Debug** button to launch the **Eclipse Application** in the **Runtime Workbench** to debug the product. Now that you've defined the configuration and used it once, it appears in the **Debug** toolbar menu (see Figure 2–21). Selecting it from that menu launches the **Runtime Workbench** without opening the **Configuration** wizard.

After clicking the **Debug** button in the **Configuration** wizard or selecting **Favorites** from the **Debug** toolbar menu, Eclipse opens a second workbench window (the **Runtime Workbench**, as opposed to the **Development Workbench**). This **Runtime Workbench** window executes the code in the projects contained in the **Development Workbench**. Making changes and setting breakpoints in the **Development Workbench** affects the execution of the **Runtime Workbench** (see Section 1.10, Introduction to Debugging, on page 59 for more about this).

2.7 PDE Views

The Plug-in Development Environment (PDE) provides several views for inspecting various aspects of plug-ins. To open the various PDE views, select **Window > Show View > Other...**; in the **Show View** dialog, expand both the **PDE** and **PDE Runtime** categories.

2.7.1 The Plug-in Registry view

The **Plug-in Registry** view displays a tree view of all plug-ins discovered in the current workspace (see Figure 2–24). Expanding the plug-in in the tree shows its components such as extension points, extensions, prerequisites, and runtime libraries.

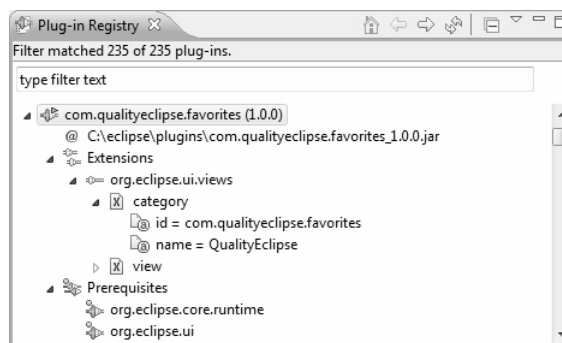


Figure 2–24 The Plug-in Registry view.

2.7.2 The Plug-ins view

The **Plug-ins** view shows a tree list of external plug-ins and plug-in projects in the current workspace and provides a quick way to review plug-ins that already exist (see Figure 2–25). In the tree, you can expand each external plug-in to browse the files located in the plug-in directory. Unfortunately, if that plug-in is contained in a JAR file rather than a directory (new in Eclipse 3.1), the files are not displayed in this view (see Bugzilla entry 89143 at bugs.eclipse.org/bugs/show_bug.cgi?id=89143). Double-clicking on a file element opens that file in an editor for viewing, and there are several useful actions in the context menu such as **Add to Java Search** (see Section 1.6.2.1, Java Plug-in Search, on page 32), **Find References** and **Open Dependencies**.

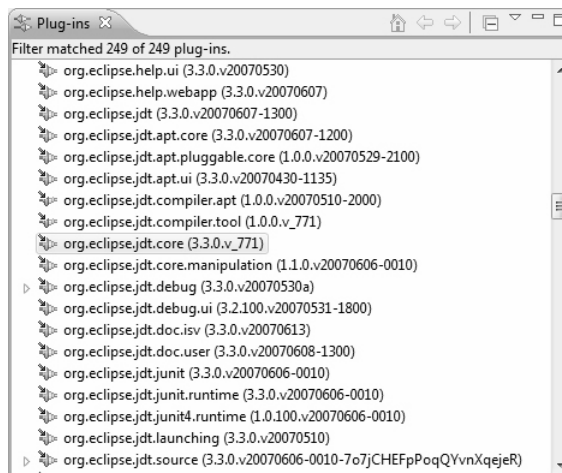


Figure 2–25 The Plug-ins view.

2.7.3 The Plug-in Dependencies view

The **Plug-in Dependencies** view shows a hierarchy of which plug-ins are dependent on which other plug-ins, which in turn are dependent on other plug-ins, and so on (see Figure 2–26). When the view opens, first right-click on the `com.qualityeclipse.favorites` plug-in and select **Focus On**. Double-clicking on an element in the tree opens the plug-in manifest editor for the corresponding plug-in.

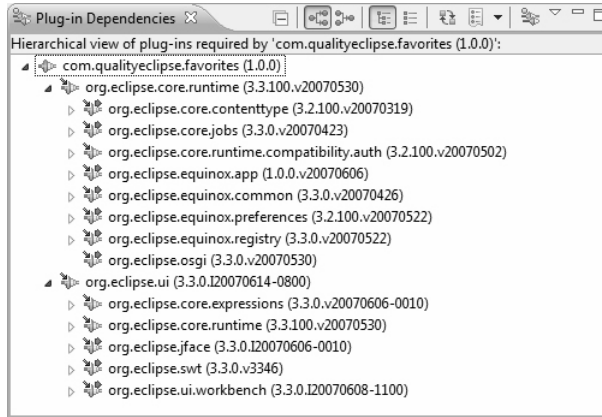


Figure 2–26 The Plug-in Dependencies view.

2.7.4 Plug-in Artifact Search

In addition to the views described above, PDE provides the ability to search for extension references, extension point declarations, and plug-ins all in one place. Type `Ctrl+Shift+A` to open the PDE search dialog (see Figure 2–27), then enter the plug-in ID to filter the list. The dialog also includes filters for extensions and extension points to help you quickly and easily find what you are looking for.

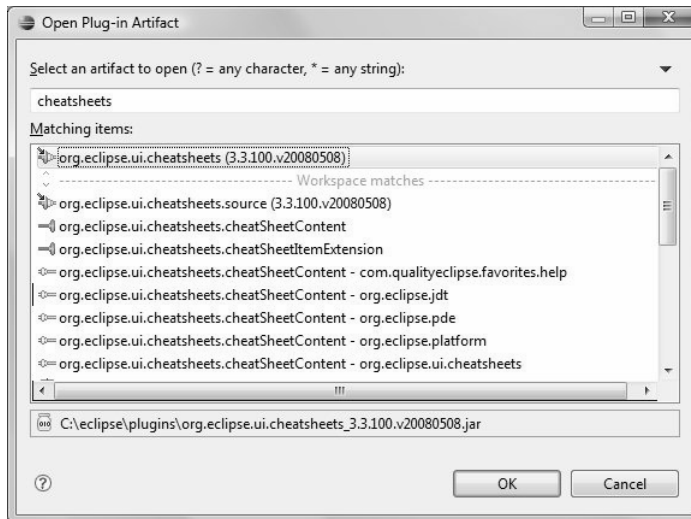


Figure 2–27 PDE Plug-in Artifact dialog.

2.7.5 Plug-in Spy

To find more information about the currently selected user interface element, open the Plug-in Spy (see Figure 2–28) by pressing Alt+Shift+F1. The Plug-in Spy (also known as the PDE Spy) currently provides information about selections, editors, views, dialogs, preference pages, and wizards. When reviewing the information provided by the Plug-in Spy, clicking on the various hyperlinks opens the Plug-in Manifest editor on that plug-in.

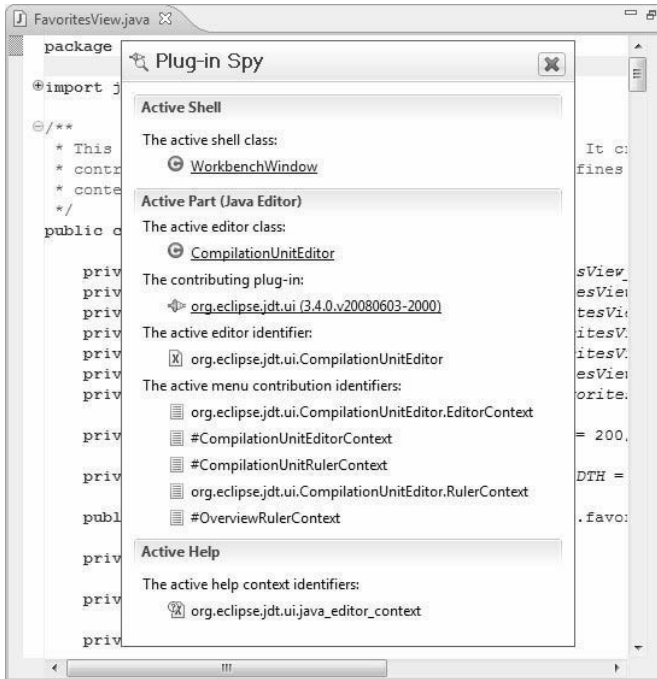


Figure 2–28 Plug-in Spy popup.

2.8 Writing Plug-in Tests

Eclipse is a continually moving target, and when building plug-ins, tests are necessary to ensure that the product continues to function properly over multiple releases. If the goal was to develop and release a plug-in once, then manual testing would suffice; however, a combination of automated and manual tests are better at preventing regressions from creeping into the product over time.

2.8.1 Test preparation

Before a test for the **Favorites** view can be created, you must modify the Favorites plug-in manifest so that the appropriate classes are visible to the test plug-in. Open the plug-in manifest editor by double-clicking on the `plugin.xml` file, then switch to the **Runtime** page (see Figure 2–11). In the **Exported Packages** section, click **Add...**, select the `com.qualityeclipse.favorites.views` package and save the changes by selecting **File > Save**.

Tip: You can limit the visibility of your exported packages by specifying which plug-ins can access a package in the **Package Visibility** section of the **Runtime** page in the plug-in manifest editor (see Section 21.2.5, *Related plug-ins*, on page 783). Alternatively, you can place tests into a fragment so that no packages need to be exported (for more on fragments, see Section 16.3, *Using Fragments*, on page 629).

Next, add the appropriate accessor so that the test can validate the view content. In the `FavoritesView` class, add the following method:

```
/**
 * For testing purposes only.
 * @return the table viewer in the Favorites view
 */
public TableView getFavoritesViewer() {
    return viewer;
}
```

2.8.2 Creating a Plug-in test project

Use the same procedure as outlined in Section 2.2, *Creating a Plug-in Project*, on page 72, to create a new plug-in project with the following exceptions:

- Name the project `com.qualityeclipse.favorites.test`
- Uncheck the **Create a plug-in using one of these templates** checkbox

After the project has been created, use the **Dependencies** page of the plug-in manifest editor (see Figure 2–10 on page 79) to add the following required plug-ins and then save the changes:

- `com.qualityeclipse.favorites`
- `org.junit4`

2.8.3 Creating a Plug-in test

When a project has been created and the plug-in manifest modified, it's time to create a simple test for the **Favorites** plug-in (see the following code example). The goal of the test is to show the **Favorites** view, validate its content, and then hide the view.

```
package com.qualityeclipse.favorites.test;

import static org.junit.Assert.assertArrayEquals;
import static org.junit.Assert.assertEquals;
import org.eclipse.core.runtime.Platform;
import org.eclipse.jface.viewers.IStructuredContentProvider;
import org.eclipse.jface.viewers.ITableLabelProvider;
import org.eclipse.jface.viewers.TableViewer;
import org.eclipse.swt.widgets.Display;
import org.eclipse.ui.PlatformUI;
import org.junit.After;
import org.junit.Before;
import org.junit.Test;

import com.qualityeclipse.favorites.views.FavoritesView;
/**
 * The class <code>FavoritesViewTest</code> contains tests
 * for the class {@link
 *     com.qualityeclipse.favorites.views.FavoritesView}.
 * @pattern JUnit Test Case
 * @generatedBy CodePro Studio
 */

public class FavoritesViewTest
{
    private static final String VIEW_ID =
        "com.qualityeclipse.favorites.views.FavoritesView";

    /**
     * The object that is being tested.
     *
     * @see com.qualityeclipse.favorites.views.FavoritesView
     */
    private FavoritesView testView;

    /**
     * Perform pre-test initialization.
     */
    @Before
    public void setUp() throws Exception {
        // Initialize the test fixture for each test
        // that is run.
        waitForJobs();
        testView = (FavoritesView)
            PlatformUI
                .getWorkbench()
                .getActiveWorkbenchWindow()
                .getActivePage()
                .showView(VIEW_ID);

        // Delay for 3 seconds so that
        // the Favorites view can be seen.
        waitForJobs();
        delay(3000);
        // Add additional setup code here.
    }
}
```

```
/**
 * Run the view test.
 */
@Test
public void testView() {
    TableView viewer = testView.getFavoritesViewer();
    Object[] expectedContent =
        new Object[] { "One", "Two", "Three" };
    Object[] expectedLabels =
        new String[] { "One", "Two", "Three" };

    // Assert valid content.
    IStructuredContentProvider contentProvider =
        (IStructuredContentProvider)
            viewer.getContentProvider();
    assertEquals(expectedContent,
        contentProvider.getElements(viewer.getInput()));

    // Assert valid labels.
    ITableLabelProvider labelProvider =
        (ITableLabelProvider) viewer.getLabelProvider();
    for (int i = 0; i < expectedLabels.length; i++)
        assertEquals(expectedLabels[i],
            labelProvider.getColumnText(expectedContent[i], 1));
}

/**
 * Perform post-test cleanup.
 */
@After
public void tearDown() throws Exception {
    // Dispose of test fixture.

    waitForJobs();
    PlatformUI
        .getWorkbench()
        .getActiveWorkbenchWindow()
        .getActivePage()
        .hideView(testView);

    // Add additional teardown code here.
}

/**
 * Process UI input but do not return for the
 * specified time interval.
 *
 * @param waitTimeMillis the number of milliseconds
 */
private void delay(long waitTimeMillis) {
    Display display = Display.getCurrent();

    // If this is the UI thread,
    // then process input.
}
```

```
        if (display != null) {
            long endTimeMillis =
                System.currentTimeMillis() + waitTimeMillis;
            while (System.currentTimeMillis() < endTimeMillis)
            {
                if (!display.readAndDispatch())
                    display.sleep();
            }
            display.update();
        }
        // Otherwise, perform a simple sleep.

    else {
        try {
            Thread.sleep(waitTimeMillis);
        }
        catch (InterruptedException e) {
            // Ignored.
        }
    }
}

/**
 * Wait until all background tasks are complete.
 */
public void waitForJobs() {
    while (!Job.getJobManager().isIdle())
        delay(1000);
}
}
```

2.8.4 Running a Plug-in test

The next step after creating a test class is to configure and execute the test. Similar to creating a runtime configuration (see Section 2.6.1, Creating a configuration, on page 94), creating a test configuration involves right-clicking on the `FavoritesViewTest` in the **Package Explorer** and selecting the **Run As > JUnit Plug-in Test** command. This automatically builds a test configuration and executes the test. You should then see the **Runtime Workbench** appear, the **Favorites** view open, and the **Runtime Workbench** close. The **JUnit** view indicates that your test executed successfully and the **Favorites** view content has been validated (see Figure 2–29).

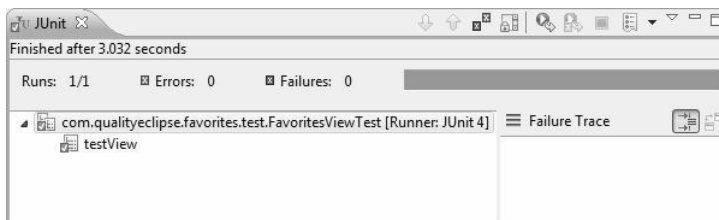


Figure 2–29 The JUnit view.

Right clicking on the `FavoritesViewTest` once again and selecting **Run As > Run Configurations...** opens the **Configuration** wizard (see Figure 2–30). Here you can specify whether a single test should be executed by itself or whether all tests in a project should be executed simultaneously. Eclipse defaults to launching a product, which opens the Welcome view (see Figure 1–2 on page 4) rather than an application. To change this, click on the **Main** tab and select the **Run an application** radio button.

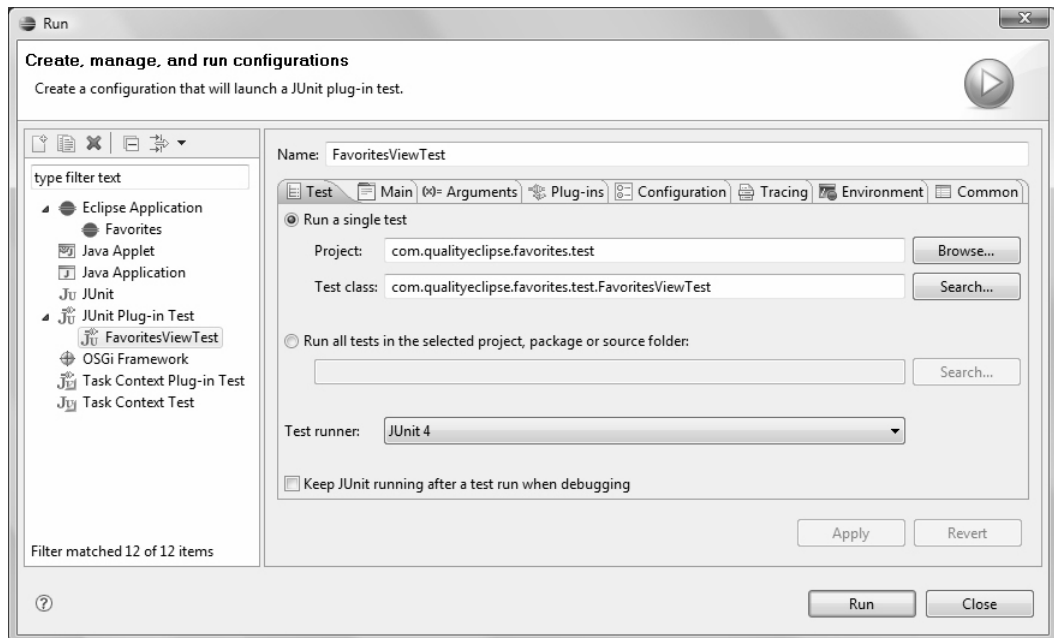


Figure 2–30 The test Configuration wizard.

2.8.5 Uninstalling the Favorites plug-in

Use the following steps to delete the **Favorites** plug-in from the **Development Workspace**:

1. Close the **Favorites** view.
2. Shut down Eclipse.
3. Delete the `com.qualityeclipse.favorites_1.0.0.jar` file in the Eclipse plug-ins directory.
4. Restart Eclipse. If you get an error message (see Figure 2–31) when restarting, at least one of the **Favorites** views was not closed when Eclipse was shut down in Step 2.

5. Verify that the **Favorites** view is no longer available by opening the **Show View** dialog (see Figure 2–18) and verifying that the **Quality Eclipse** category is no longer present (see Figure 2–19).

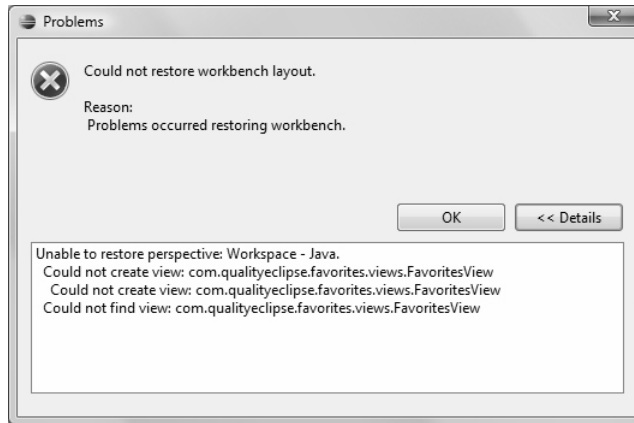


Figure 2–31 Problems dialog when restarting Eclipse.

2.9 Book Samples

The sample code for each chapter in this book can be downloaded and installed into Eclipse for you to review. Download and install the book samples from <http://www.qualityeclipse.com> or using the update manager (see Section 18.3.5, Accessing the update site, on page 685) by entering “<http://www.qualityeclipse.com/update>” (see Figure 18–29 on page 687). Once installed, open the view by selecting **Window > QualityEclipse Book Samples** (see Figure 2–32).

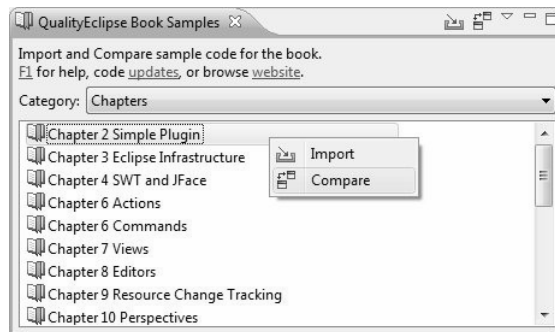


Figure 2–32 QualityEclipse Book Samples View

Using the book samples view, you can compare your work to the sample code and load the code for particular chapters for review.

2.10 Summary

This chapter covered the process of creating, running, debugging, inspecting, and testing a simple plug-in from start to finish. Subsequent chapters will cover every aspect of this process, plus more in much greater detail.

References

Gamma, Eric, and Kent Beck, *Contributing to Eclipse*. Addison-Wesley, Boston, 2003.

McAffer, Jeff, and Jean-Michel Lemieux, *Eclipse Rich Client Platform: Designing, Coding, and Packaging Java Applications*. Addison-Wesley, Boston, 2005.

FAQ How do I find a particular class from an Eclipse plug-in? (http://wiki.eclipse.org/FAQ_How_do_I_find_a_particular_class_from_an_Eclipse_plug-in%3F)

Helpful tools in PDE Incubator (<http://www.eclipse.org/pdel/incubator/>)

Symbols

`{ant.file}` 700
`{ant.java.version}` 700
`{ant.project.name}` 700
`{ant.version}` 700
`{basedir}` 700
`{eclipse.home}` 700
`{eclipse.pdebuild.home}` 700
`{eclipse.pdebuild.scripts}` 700
`{eclipse.pdebuild.templates}` 700
`{eclipse.running}` 694
`{eclipse.running}` 695, 700
`{os.name}` 700
`{variable}` 699
`nl` 633
`$NON-NLS-1$` 623, 627
 % (percent sign)
 in `plugin.xml` file 618
 in String formatting 624
 & (ampersand) 285
`&`; 285
`**` 696
 .* resources filter 22
`<and>` 261
`<ant>` 695–696
`<antcall>` 695, 706, 708
`<code>` 642
`<fileset>` 696, 698
`<not>` 261
`<or>` 261
`<pre>` 642
`<samp>` 642, 646
`<view>` 82
`<visibility>` 261
`<zipfileset>` 698

A

About dialog 115, 617, 674–676, 678, 689

About Eclipse SDK Features dialog 689

About Features dialog 675–676

`about.html` file 674–675, 690

`aboutImage` 678

`about.ini`

`aboutImage` 674

`aboutText` 675

 branding information 675

`featureImage` 674–676

 file 664, 674

 localization 676

 relation to **About** dialog 617

`tipsAndTricksHref` 675–676

`windowImages` 674

`about.mappings` file 674, 676

`about.properties` file 664, 674, 676–677

`aboutText` 675, 678

abstract base class vs. interface, extension points 647

`AbstractConnectionEditPart` 734

`AbstractDetailsDialog`

 class 459

 extends 456

`AbstractEditPart` 734–735

`AbstractElementListSelectionDialog` 443

`AbstractFavoritesGraphicalEditPart` 750

`AbstractFavoritesTest`

 class 253

 extends 253, 268

`AbstractGraphicalEditPart` 734

`AbstractHandler`

 class 315, 322, 334, 473, 799

 extends 325

`AbstractOperation` 393

`AbstractPreferenceInitializer` 506

`AbstractUIPlugin`

 class 83, 123, 125, 505, 653

`imageDescriptorFromPlugin` 84

`Abstract Windowing Toolkit`, *see* AWT

Access restriction 254

Action

 class 319, 414

 enablement 251

 extends 319

 menu groups 245

 menu item 245

action

`allowLabelUpdate` 245

 cheatsheet 610

 class 246, 259, 264, 271, 279, 281, 790, 793

`definitionId` 246, 281

`disabledIcon` 246

`enablesFor` 246, 259, 264, 272, 281

`helpContextId` 246, 259, 272, 282, 593

`hoverIcon` 246, 282

`icon` 246, 259, 272, 282, 790, 793

`id` 245, 259, 264, 272, 279, 281, 790, 793

`label` 245, 259, 264, 272, 274, 279, 281, 790, 793

`menubarPath` 245, 259, 264, 272, 275, 279, 281, 790, 793

`overrideActionId` 259, 272

`retarget` 246

`state` 246, 259, 272, 282

`style` 247, 260, 282, 790

`toolbarPath` 245, 282, 793

`tooltip` 245, 259, 264, 272, 282, 790, 793

`url` 790

`ActionDelegate`, run 252

- ActionFactory 399
 - COPY 386
 - CUT 386
 - DELETE 386, 388
 - FIND 386
 - PASTE 386
 - REDO 386
 - SELECT_ALL 386
 - UNDO 386
- ActionHandler 237
- Actions
 - & 285
 - <and> 261
 - <not> 261
 - <or> 261
 - adapt 261
 - adaptable attribute 257
 - and 262
 - appear in reverse order 259
 - associating commands 284
 - clipboard 322
 - context menus 12
 - convert to handler 237
 - customizing 14
 - delegate 249
 - editorContribution 280
 - Editors 277
 - enablement 264
 - enablesFor 251, 260
 - filter 263
 - filtering 317
 - filtering and enablement 260
 - filtering based on file content 265
 - filters 519
 - global 321, 349
 - global labels 286
 - groupMarker 267, 274
 - helpContextId 246
 - hover icon 246
 - IActionFilter 263
 - IActionSetDescriptor 256
 - IAction vs. IActionDelegate 240
 - IEditorActionDelegate 279
 - images 247
 - insertion points 248
 - instanceof 262
 - Invalid Menu Extension 249
 - IObjectActionDelegate 266
 - IResourceActionFilter 263
 - IViewActionDelegate 271, 273
 - IWorkbenchWindowActionDelegate 246
 - IWorkbenchWindowPullDownDelegate 246
 - key bindings 238
 - keyboard 320
 - keyboard accessibility 285
 - lazy initialization 241, 260
 - lifecycle events 242
 - manual testing 252
 - menubarPath 249
 - nested actionSet problem 249
 - not 262
 - Object 257
 - objectClass 262
 - objectState 262
 - or 262
 - overrideActionId 259
 - overview 11
 - persistentProperty 263
 - pluginState 262
 - projectNature 263
 - projectPersistentProperty 264
 - projectSessionProperty 264
 - retarget 246
 - retargetable 387
 - RFRS 286
 - Run 56
 - Run History 56
 - run method 252
 - selection 264
 - selectionChanged 251
 - separator 274
 - sessionProperty 264
 - setEnabled 251
 - submenu 267, 270, 274
 - systemProperty 262
 - systemTest 262
 - test 262
 - testing 253, 268, 275, 283
 - toolbar action 273
 - toolbar button 245
 - toolbarPath 249
 - toolbars 13
 - tooltip 259
 - top-level editor menu 280
 - top-level menus 11
 - underscore 285
 - viewerContribution 271
 - views 270
 - visibility 261
 - workbench window 242
- ActionSet 223
 - actionSet
 - extension 436–437
 - id 244
 - label 244
 - visible 244
- ActionSets 222, 256
 - action sets 14
 - activateHandler 322
 - activeEditorId 229
 - activePartId 324
 - activeWhen 236, 324

- adapt 261
- adaptable
 - decorator 803
 - objectContribution 257
 - page 517
- Adaptable, to resource 518
- Adapters
 - example 400
 - factory 786
 - for viewers 193
 - framework 784
 - getAdapter 294
 - IAdaptable 337, 785
 - IPropertySource 524
 - IWorkbenchAdapter 788
 - registering at startup 786
 - unregistering at shutdown 786
 - using 785
- add 156, 159, 200, 203, 207
- addActionSet 428–429
- addArmListener 175
- addBuilderToProject 545, 547, 565, 570
- addCategory 396
- addCheckStateListener 204, 209
- addControlListener 145, 163
- Add CVS Repository** dialog 50, 797
- addDisposeListener 145
- addDoubleClickListener 198, 202
- addDragListener 328
- addDragSupport 198
- addDropListener 329
- addDropSupport 199
- addEditorActivationListener 336, 378
- addEntry 396
- addFastView 429
- addFavorites 326
- addFavoritesListener 512, 524
- addFavoritesManagerListener 805
- Add Feature** command 671
- addField 489
- addFilter 199
- addFocusListener 145
- addHelpListener 146, 175–176, 199
- additions identifier 248
- addKeyListener 146, 388
- addListener 145, 149, 377
- addMenuListener 175, 316, 382
- addModifyListener 154, 159, 161, 476
- addMouseListener 146, 339
- addMouseMoveListener 146
- addMouseTrackListener 146
- addNature 570–571
- addNewSection 450, 475
- addNewWizardShortcut 429
- Add-ons for Eclipse 819
- addOpenListener 199
- addOverlay 804
- addPage 359–360, 474
- addPageChangeListener 469
- addPages 465
- addPaintListener 146
- addPartListener 388
- addPerspectiveShortcut 429
- addPlaceholder 428, 430
- Add Plug-in** command 671
- addPostSelectionChangedListener 199
- addPostSelectionListener 338
- addPropertyChangeListener 508, 810
- addPropertyFileListener 375
- addRegistryChangeListener 128
- addResourceChangeListener 408
- addResourcesTo 805
- addSaveParticipant 420
- addSelectionChangedListener 199, 202, 314, 316
- addSelectionListener 152, 154, 156, 158–159, 161–163, 165–166, 168, 170, 172, 176, 179, 310, 314, 318, 476
- addShellListener 150
- addShowInPart 430
- addShowViewShortcut 430
- Add Site** button 687
- addStandaloneView 430
- addStyleRange 213
- addSuffix 804
- addTextListener 211
- AddToFavoritesHandler 238, 313
- AddToFavoritesTest 268
- Add to Java Search** command 32, 97
- addTraverseListener 146
- addTreeListener 166, 207
- addVerifyListener 155–156
- addView 428, 430
- adjustForNumColumns 492
- afterEditorActivated 336
- afterEditorDeactivated 336
- Alphaworks on Eclipse 830
- ALT 336
- Alt+Shift+F1 99
- Alt-click 377
- AltClickCellEditListener 378
- and 262
- Ant
 - also known as* Apache Ant
 - `{eclipse.home}` 700
 - `{eclipse.pdebuild.home}` 700
 - `{eclipse.pdebuild.scripts}` 700
 - `{eclipse.pdebuild.templates}` 700
 - `{eclipse.running}` 700
 - `{variable}` 699
 - ** 696
 - <ant> 696
 - <antcall> 696, 706

Ant (*continued*)`<antcall>` vs. depends 695`<fileset>` 696**Arguments** field 704

auto generated build script 727

basedir 694

brief introduction 693

buildDir 699

build-favorites.xml 91

building the product 89

build.update.jar target 727

build.xml 727

Classpath variables 725

command-line properties 696, 704

compiling Java source 698

Convert file system path 697

Convert resource path 697

copy 696

Create Ant Build File 727

creating directories 698

debugging 697, 716

default target 694

delete 697

determining Eclipse installation directory 700

determining if launched in Eclipse 700

echo 697, 702

echoproperties 716

echoproperties 697

ECLIPSE_HOME 725

ECLIPSE30_HOME 726

eclipse.buildScript 697, 710, 718

eclipse.convertPath 697, 710

eclipse.fetch 697, 710

eclipse.generateFeature 697, 710

eclipse.incrementalBuild 697, 710

eclipse.jarProcessor 697

eclipse.pdebuild.scripts property 715

eclipse.refreshLocal 698, 710

extensions 710

Generate a build script 697

import 715

includeEmptyDirs 697

inheritAll 696

init target 727

introduction 693

javac 698

macrodef 708

mkdir 698

multiple assignments ignored 703

pde.exportPlugins 89, 91

predefined properties 700

project 693

project basedir 694

project default 694

project name 694

properties 699

properties outside targets 703

properties target 727

property 715

property 698, 702

property mutability 702

property scoping 701

record 716

refresh resources 698

replace 90

Retrieve from CVS 697

Run in the same JRE as the workspace 91, 710, 718

running 91

script generation 89

sequential 708

sub 708

system properties 700

target 693–694, 701

target depends 694

target description 694

target execution order 701

target if 694

target name 694

target unless 695

tasks 695

undefined property 699

Use global properties checkbox 705

user properties 696

zip 698

ZIP files 698

zip.plugin target 727

antBuildFile 265–266

AntBuildfileContentDescriber 266

Ant script 89

Apache Ant, *see* Ant**Appearance** preferences 15, 17

appendBody 792

appendText 380

appendVerifyKeyListener 211

applyDialogFont 448

applyEditorValue 377

appName 678

Arm event 143

ArmListener 175

ArrayContentProvider

class 202–203, 206

extends 211

articles 828

artifacts.xml file 683

asLocalURL 122

asResources 323

Assert 346

Associated editors list 47

asText 323, 328, 380

asyncExec 148, 371, 413

auditPluginManifest 541, 551–552

AUTO_BUILD 539

Auto-generated build script 727

AWT 136

B

Background Tasks 808

baseLocation property 715

base property 715

BasicFavoriteItem 512–513, 515

BasicFavoriteItemEditPart 750

BasicTextEditorActionContributor 385

Batching resource change events 414

beep 149

beforeEditorActivated 336

beforeEditorDeactivated 336

beginTask 415, 474, 809, 811

Binary Projects 780

BinarySignatureDescriber 357

bin directory 20–21

Book Samples Online 105

BooleanFieldEditor 490, 495–496, 809

branches, CVS 50

Branding

about.html 674

about.ini 675

about.mappings 676

about.properties 676–677

Banner Image 667

splash screen 679

using features 673

browseForSourceFile 479

Browsers

LaunchURL 789

OpenBrowserAction 790

opening 788

BrowserViewer 789

Bucknell xliv

BufferedOutputStream 333

Bugzilla 65–69

Eclipse bug tracking system 782

entry #105949 249

entry #36389 249

entry #39455 256

entry #58413 89

web page 801

Bugzilla Query wizard 67

build

forget last state 541

build 539, 548, 552

Build automatically preference 572

Build Configuration editor 720

buildDirectory property 715

builder

add 570

remove 570

builder

callOnEmptyDelta 537

hasNature 537–538

isConfigurable 537

builder property 715

Builders

associating with a project 545

creating 538

declaring 535

defined 535

derived resources 533, 535, 545

early startup 535

global 535

hadNatures 537

hasNature 564

invoking 548

RFRS 572

build-favorites.xml 715

build-favorites.xml file 87–91, 111–112

Building the product

auto-generated script 711

link files 111

manually 87

using Ant 89

buildMailSpec 791

Build Order preferences 16–17

build.properties

bin.includes 722

editor 720

extra.<library> 722

jars.extra.classpath 722

output.<library> 722

source.<library> 722

build.properties file 714

build.properties file 720

build.properties page 721

Build script 693

buildSpec 546

build.xml file 89, 265, 727

Bundle 122

class 456

Bundle-ActivationPolicy 77, 84

Bundle-Activator 79, 114, 116, 120, 634

Bundle-ClassPath 80, 110, 116, 634

BundleContext 83

Bundle-ManifestVersion 583, 634

Bundle-Name 114, 459, 583, 634

Bundle-RequiredExecutionEnvironment 79

Bundles, *see* Plug-ins

Bundle-SymbolicName 79, 114, 128, 583, 634

Bundle-Vendor 79, 114, 459, 583, 634

Bundle-Version 79, 114, 583, 634

bundle-version 634–635

Button

class 143, 152

example 154

- buttonPressed 446, 461
- ByteArrayInputStream 332
- ByteArrayOutputStream 332–333
- ByteArrayTransfer
 - class 330–331
 - extends 331
- C**
- CANCEL_STATUS 394
- cancelEditing 204
- cancelEditor 377
- cancelPressed 446
- canDoOperation 211
- canEdit 335, 373
- canFinish 465
- Capabilities preferences** 16
- Caret 145
- category
 - cheatsheet 605–606
 - id 292, 470, 605, 618
 - method 197
 - name 292, 470, 605, 618
 - page 487, 498–499, 517, 530
 - parentCategory 470
 - view 292, 618, 755
- categoryId
 - command 219, 763
- CellEditor 204, 335, 373, 376–377
- Change Method Signature** command 43
- changeTextPresentation 212–213
- Character 156
- Cheat Sheet
 - category 605
 - Description 605
 - description 606
 - Editor 604
 - Register 604
 - Wizard 603
- cheatsheet
 - action 610
 - category 605–606
 - contentFile 606
 - id 605–606
 - item 610
 - name 605–606
- Cheat sheets
 - adding commands 607
 - category 605
 - Click to Begin** command 602
 - Click to Perform** command 602, 610
 - Click to Skip** command 602
 - contentFile 606
 - creating 603
 - Open Related Help** command 610
 - using 601
- Cheat Sheet Selection** dialog 602
- checkboxes 152
- CheckboxTableViewer 203–204, 481
- CheckboxTreeViewer 207, 209
- CheckedTreeSelectionDialog 445
- Checking out a project from CVS 51
- Check Out As...** command 52
- Check Out** command 51
- checkState 489, 497
- class
 - command handler 237
 - decorator 803
 - editor 356, 761
 - itemType 642
 - page 487, 498–499, 517, 530
 - perspective 426
 - view 292, 618, 755
 - viewContribution 273
 - wizard 470, 472
- class field** 237, 249
- Class, forName 655
- ClassLoader
 - extends 814
 - Plug-in 811
 - ProjectClassLoader 813
 - restrictions 783
- classpath
 - access restriction 254
 - Java Build Path 23, 80
 - launch configuration 57
 - variables 725
- .classpath file 22–23
- Classpath** page 57
- Classpath Variables** preferences 17
- class visibility between plug-ins 100
- clean 539
- CLEAN_BUILD 539, 574
- clearSelection 155, 159
- Clear workspace data before launching 252
- Clipboard 322–323, 326
 - setContent 324
- Clipboard
 - actions 400
 - commands 322
 - serializing state 296
- ClipboardHandler 326
- close 149, 151, 446
- code formatting 36
- Code Generation** preferences 17
- CodePro xxxviii, xliv, 26, 64, 628
- CodePro Profiler 821
- collapseAll 207
- collapseToLevel 208
- Color 523
- Color 164, 167, 212, 523
- ColorDialog 442
- ColorFieldEditor 490
- ColorPropertyDescriptor 527
- Colors 189
- Colors and Fonts** preferences 55
- ColorSelector 521

- Color syntax highlighting 36
- ColumnLabelProvider 335
- ColumnPixelData 348
- ColumnViewerEditorActivationEvent 336
- ColumnViewerEditorActivationListener 336, 377
- ColumnWeightData 348, 362
- Combo
 - class 143, 159, 161
 - example 161
- Command 216, 219
- command
 - activePartId 324
 - categoryId 219, 593, 763
 - commandId 221–222, 224, 229, 763
 - defaultHandler 763
 - description 219, 593, 763
 - editor 357
 - enableWhen 334
 - helpContextId 593
 - icon 221–222, 224, 229
 - id 219, 221–222, 593, 763
 - label 221
 - mnemonic 221
 - name 219, 593, 763
 - selection 324–325, 334
 - tooltip 222
- Command Category 216, 219
- command category
 - description 219
 - id 219
 - name 219
- command groups 14
- command handler
 - activeWhen 236
 - class 237
 - commandId 236
 - enabledWhen 236
- Command Handlers 236
- commandId
 - command 221–222, 224, 229, 763
 - command handler 236
- command line
 - clean 92
 - debug 127
 - obtaining arguments 126
- Commands
 - associating actions 284
 - convert Action to Handler 237
 - Copy 322
 - Cut 325
 - global 321
 - HandlerUtil 237
 - keyboard 320
 - org.eclipse.ui.file.properties 522
 - Paste 326
 - views 313
- COMMENT_PREFKEY 515
- commercializing 661
- Commit...** command 53
- Common** page 57
- Common Widget 135
- Comparable, implements 297
- Comparator 309–311
- compare 197, 203
- Compare/Patch** preferences 16
- Compare** dialog 46
- compareToIgnoreCase 202
- Compare With > Another Branch or Version...** command 54
- Compare With > History...** command 53
- Compare With > Latest from Head** command 54
- Compare With > Local History...** command 46
- comparing versions 671
- Compiler**
 - nonexternalized strings option 628
 - preferences 17
- compile-time classpath 23
- Composite
 - class 169–170, 362, 373, 451, 461
 - example 170
- Compute** command 671
- com.qualityeclipse.favorites_1.0.0 directory 110
- com.qualityeclipse.favorites_1.0.0.jar 80, 88, 110–111
- com.qualityeclipse.favorites.editors 360
- com.qualityeclipse.favorites.link file 112
- com.qualityeclipse.favorites.popupMenu 267
- com.quality.favorites_1.0.0.jar file 104
- Concurrent Versions System, *see* CVS
- Conferences, EclipseCon 829–830
- config.inifile 678
- configuration directory 679
- ConfigurationScope** 503
- Configuration** wizard 96, 104
- configure 565
- configureShell 453, 460
- CONNECTION_LAYER** 756
- ConnectionLayer 756
- Connection Type 797
- Console** view 56, 412, 553, 697, 801, 812
- Constants
 - Copy 321
 - Cut 321
 - Delete 321
 - IWorkbenchActionDefinitionIds 321, 324
 - keyboard 321, 324
 - Paste 321
 - Redo 321
 - Undo 321
- ContainerCheckedTreeViewer 445
- ContainerSelectionDialog 444
- Containing text** field 28–29
- contains 505

- Content Assist 38, 346
- Content Assist** preferences 39
- contentFile, cheatsheet 606
- ContentOutlinePage 401
- Content providers
 - defined 195
 - editor 369
 - hierarchy 196
 - responsibility 85
 - view 306
- contentTypeBinding, editor 357
- Content Types** preferences 16
- content.xml file 683
- context 228
- Context menus 325
 - additions 382
 - createContextMenu 382
 - creating 316
 - dynamically building 382
 - editor 381
 - invoking 12
 - view 314
- contexts
 - file 597
 - plugin 597
- contexts.xml file 595, 598
- contributeToToolBar 388
- ContributionItem 314, 319
- Contributions
 - views 314
- contributorClass, editor 356, 761
- Control 143–145
- Control event 143
- ControlListener
 - class 145, 163
- Convert Anonymous Class to Nested** command 43
- convertHeightInCharsToPixels 449
- convertHorizontalDLUsToPixels 449, 458, 460
- Convert Local Variable to Field** command 44
- Convert Member Type to Top Level** command 43
- convertVerticalDLUsToPixels 449
- convertWidthInCharsToPixels 449
- CoolItem 143
- Copy 322
- copy 155, 159
- CopyFavoritesHandler 322–325, 328
- copyright, feature 665
- CoreException 457, 560
- CoreExceptions, status attribute 130
- create 269
- Create a Java project** option 581
- Create Ant Build File** command 727
- createBrowser 788–789
- createButton 446
- createButtonBar 447
- createButtonsForButtonBar 446–447, 460
- createChild 341
- createContents 447, 489, 494–495, 519
- createContextMenu 316–317, 382
- createContextSnapshot 315
- createContributions 316
- createControl 467, 475, 755
- createDetailsArea 458
- createDetailsViewer 458–459
- createDialogArea 447, 451–452, 460
- createExecutable 655, 793
- createExecutableExtension 652, 654
- createFieldEditors 489, 496
- createFigure 750–751
- createFolder 428, 430
- createImageRegistry 125
- createInitialLayout 427–428
- createInlineEditor 335, 372, 523
- createMarker 552
- createPageControls 465
- createPages 359
- createPartControl 86, 251, 293, 295, 320, 326, 358, 507
- Create Patch...** command 801
- createPlaceholderFolder 430
- createProductInfoArea 458
- createPropertiesPage 360, 362
- CreatePropertyKeyResolution 559
- createSourcePage 360
- createStatus 129
- createTableSorter 310, 342
- createTableView 524
- createToolBarButtons 318
- createToolTipLabel 750
- createTypeCheckboxes 452
- createViewPulldownMenu 320, 342
- createWriteRoot 344
- Creating a plug-in project 72
- Creating a plug-in test 100
- Creating a plug-in test project 100
- Creating a project 19
- Ctrl+F11** key 56
- Ctrl+Shift+A 98
- Ctrl-Space** key 38, 40
- currentTimeMillis 103
- Customize Perspective** dialog 14, 244, 252
- Customizing actions 14
- Cut 325
- cut 155, 159
- CutFavoritesHandler 325
- CVS
 - also known as* Concurrent Versions System
 - branches 50
 - Checking out a project 51
 - Check Out As** command 52
 - Check Out** command 51
 - Checkout Projects from CVS** wizard 52

- comparing and replacing 53
 - comparison and merge tools 49
 - comparison editor 53
 - CVS Repository Location** wizard 50
 - Eclipse repository 797
 - getting started 50
 - HEAD 50
 - label decorators 16, 54
 - patch 801
 - repository 49
 - synchronizing 52
 - team development using 49
 - Use an existing module** option 52
 - Use existing repository location** option 52
- CVS Repositories** view 50, 797
- CVS Repository Exploring** perspective 50
- D**
- DataInputStream 332
- DataOutputStream 332–333
- Days to keep files** field 47
- deactivateHandler 322
- Debugger 812
- Debugging
 - configuration 94
 - enabled and disabled breakpoints 60
 - introduction 59
 - Runtime Workbench 94, 96
 - selecting plug-ins 95
 - setting breakpoints 59
- Debug** perspective 10–11, 59
- Debug** view 61
- deconfigure 565
- decorate 804
- decorateImage 808
- decorateText 808
- DecoratingLabelProvider 807
- decorator
 - adaptable 803
 - class 803
 - description 804, 806
 - enablement 804, 806
 - icon 803, 806
 - id 803, 806
 - label 803, 806
 - lightweight 803, 806
 - location 803, 806
 - objectClass 803
 - state 803, 806
- default Ant target 694
- DefaultBrowserSupport 789
- defaultColor 512
- DefaultEditDomain 761
- default, editor 357
- defaultHandler
 - command 763
- Default output folder** field 21
- Default Plug-in Project** wizard 581
- defaultPropertyNames 505
- defineClass 814
- Definition** page 645
- delay 101, 254
- Delayed resource changed events 420
- delete 269
- deleteAuditMarkers 551–552, 570
- Delete key 320, 388
- deleteMarkers 552
- DeletePropertiesHandler 396
- DeletePropertiesOperation 393, 397
- DeleteResourcesAction 414
- Dependencies** page 79, 100, 298, 671
- Dependencies** tab 117
- Dependency Analyzer 820
- depends, Ant target 694
- Deployable plug-ins and fragments** option 87
- Deploying Plug-ins 111
- Derived resources 533, 535, 545
- described, objectContribution 270
- description
 - command 219, 763
 - command category 219
 - decorator 804, 806
 - feature 665
- description, Ant target 694
- deselect 157, 159, 162
- deselectAll 157, 159, 162, 166
- Detached** command 6
- Details** button 455
- Dialog
 - class 446
 - extends 450, 459
 - Show View 96
- dialog_settings.xml file 126
- Dialogs
 - About** 115, 617, 674
 - About Features** 675–676
 - AbstractDetailsDialog 459
 - AbstractElementListSelectionDialog 443
 - Add CVS Repository** 50, 797
 - API 446
 - buttonPressed 461
 - Compare** 46
 - createDialogArea 451–452
 - creating 446
 - Customize Perspective** 14, 244, 252
 - defined 441
 - DetailsDialog 454
 - dialog_settings.xml 126
 - dialog units 448
 - Editor Selection 49
 - ErrorDialog 454
 - example 450

Dialogs (*continued*)

- ExceptionDetailsDialog 455
- filter 450
- getDialogSettings 125
- IconAndMessageDialog 443
- IDialogSettings 450, 475
- initial location and size 449
- JFace 443
- JFace vs. SWT 442
- launch 710
- Launch Configuration** 65, 141
- loadDialogSettings 126
- modal 443
- modal vs. modeless 441
- modeless 443
- New File Type** 48
- New Java Class 237
- New Template** 42
- New Working Set** 34
- null layout 448
- Open Type** 26
- parent 462
- Plug-in Details** 674
- Plug-ins 115
- Preferences** 14, 18
- Product Configuration** 685–686
- Properties** 23, 138, 511, 515, 522, 531, 573
- Refactoring Preview** 45
- Replace** 29, 54
- resizable 450
- Resource Selection** 247
- Select Working Sets** 34
- setReturnCode 448
- setShellStyle 447
- settings 450, 475
- shell style 447
- Show View** 93, 105, 131, 290, 780
- SWT 442
- SWT ColorDialog 442
- SWT DirectoryDialog 442
- Version Synchronization** 668
- wizards 464
- Workbench Preferences** 14
- Direct edit, views 333
- DirectoryDialog 442
- DirectoryFieldEditor 490
- Display 513
- Display
 - asyncExec 148, 371
 - class 139–140, 148, 457, 464
 - getActiveShell 464
 - getCurrent 148, 457
 - getDefault 148, 413, 464
 - readAndDispatch 103
 - reducing flicker 383
 - syncExec 148
 - timerExec 149

- Display** command 62
- displayContext 594
- displayDynamicHelp 594
- displayHelp 594, 599
- displayHelpResource 594, 599
- displaySearch 594
- Display** view 63
- dispose 139, 145, 151, 154, 294, 385, 465, 467, 489, 492, 655
- disposeColors 513
- Dispose event 143
- disposeExec 149
- DisposeListener 145
- disposeTypes 655
- DND
 - DROP_NONE 330
- DND 328
 - DROP_COPY 327, 329–330
 - DROP_MOVE 327, 329
- Document 213
- doc.zip file 590
- doFillIntoGrid 492
- doGetPreferenceStore 494
- doLoad 492
- doLoadDefault 492
- done 415, 474, 809
- doneSaving 420
- doOperation 212
- doSave 359, 361, 381
- doSaveAs 359, 361, 381
- doStore 492
- doubleClick 202–203
- Drag and Drop
 - custom 330
 - nativeToJava 332
 - views 326
- dragEnter 329
- dragFinish 328
- dragFinished 328
- dragSetData 328
- DragSource 145, 327–328
- drag source 326
- DragSourceEvent 328
- DragSourceListener 328
 - class 198, 328
- dragStart 328
- Drawing, setRedraw 307, 383
- drop 329–330
- DROP_COPY 328–330
- DROP_DEFAULT 330
- DROP_MOVE 329–330
- DROP_NONE 330
- Drop-down list 159
- dropins 673, 717
- DropTarget 145, 329
- drop target 326
- DropTargetAdapter 329

DropTargetEvent 330
DropTargetEvent 330
DropTargetListener 199, 329

E

EAF 107–108
earlyStartup 121, 421
Eclipse
 and the JRE 3
 Bugzilla 782, 801
 command-line switches 3
 configuration 3
 CVS Repository 797
 developing in one version for another 725
 distributions 2
 downloading 2
 early use of Swing 137
 Easter Eggs 830
 Getting Started 1
 Installing 3
 integration builds 2
 memory allocation 3
 memory usage 3
 modifying the base 797
 modifying to find part identifiers 797
 newsgroup 781
 nightly builds 2
 projects 828
 release notes 3
 Research 830
 RFRS 831
 setup 14
 shortcut 3
 structural overview 107
 submitting changes 801
 supported platforms 3
 Web site 1
 Wiki 829
 Workbench Overview 3
 workspace location 3
ECLIPSE_HOME 725–726
ECLIPSE21_HOME 726
ECLIPSE30_HOME 726
ECLIPSE31_HOME 726
Eclipse Application configuration 57, 95
Eclipse Application Framework 107–108
EclipseCon 830
eclipse.ini 3
Eclipse-LazyStart 77
Eclipse.org 828
Eclipse Plug-in Central 829
Eclipse products 673
Eclipse Rich Client Platform book 817
EclipseUML 822
Edit > Content Assist command 38
editElement 204, 336

EditingSupport 335, 373, 376–377
Edit menu 11, 321, 393, 402
editor
 class 356, 761
 command 357
 contentTypeBinding 357
 contributorClass 356, 761
 default 357
 extensions 356
 filenames 357
 icon 356
 id 356, 761
 launcher 357
 matchingStrategy 357
 menu item 229
 name 356, 761
 toolbar item 229
EditorActionBarContributor
 class 385
 extends 386
EditorActivationListener 336, 377
editorContribution
 id 280
 targetID 280
EditorInputTransfer 330
EditorPart 358
 class 353
 methods 358
Editor preferences 17
Editors
 accelerators 388, 391
 Actions 277
 active 386
 active editor 6
 Ant 283
 area 354
 associating file types 47
 cell editors 372
 cell validator 376
 change listeners 374
 Class File 283
 clipboard 400
 clipboard actions 386, 389
 Commands 381
 Compilation Unit 283
 content provider 369
 context menu 278, 381
 contributor 384
 CVS comparison 53
 declaration 354
 default editor 357
 Default Text 283
 dirty 378
 dispose 358
 editing a Java file 26
 editing vs. selecting 377

Editors (*continued*)

- editorContribution 280
- example 360
- external 357
- feature manifest 666
- flicker 307
- global actions 385, 388
- Go to Line** command 27
- id 356
- IEditorActionDelegate 279
- image 356
- init 359, 378
- introduction 5
- isDirty 359
- Java 35
- Java Attribute 426, 538, 557, 565, 658
- Java Editor 35
- label provider 370
- launcher 357
- lifecycle 378
- linking 400
- methods 358
- model 363
- obtaining progress monitor 419
- opening from a selection 339
- part 358
- plug-in manifest editor 77
- Quick Fix 40
- references 353
- RFRS 401
- save 359
- saving content 381
- schema 641
- Show Source of Selected Element Only 35
- Snippet 283
- tabbed 379
- Templates 40
- toolbar 388, 391
- top-level menu 280, 387, 389
- undo 392
- Editor Selection** dialog 48–49
- Editors** preferences 16
- EditorUtil 339
- editorValueChanged 377
- EditPart 734
- EditPartListener 734
- EditParts 734
- ElementListSelectionDialog 445
- ElementTreeSelectionDialog 445
- Email
 - creating 788
 - generating 601
- enabled breakpoint 60
- enabledWhen 236, 324–325, 334, 517
- enablement, decorator 804, 806
- enableOperation 212

- enablesFor 251, 260
- Encapsulate Field** command 44
- Enclosing projects scope 29–30
- Entries per file** field 47
- Environment** page 57
- ENVY 46
- .epf file 18
- EPiC 829
- Equinox project 128
- ErrorDialog 444, 454
- Error Log** view 131, 660
- Errors
 - Handling 131
- Errors/Warnings** preferences 628
- EvaluationResult.NOT_LOADED 263
- event.doit 328
- ExceptionDetailsDialog 455–456, 459
- Exception handling, extension points 651
- exec 791
- Executable extensions 653
- execute 323, 326, 334, 394, 397, 414–415, 418, 473, 599–600, 789–790
- Execute an external program 600
- executeMethod 813
- ExecuteMethodHandler 812, 814
- Execution Environments** preferences 20
- ExecutionEvent 315, 322–323, 326, 334, 397, 418, 473, 599–600, 789
- ExecutionException 315, 322, 326, 334, 397, 418, 473, 599–600, 789
- expandAll 208
- expandToLevel 208
- Export > Preferences** wizard 18
- Exporting Packages 100, 254
- Export-Package 116
- Export** wizard 87–89, 673
- Expressions** view 62
- .exsd file 639
- extending the functionality of another plug-in 629
- Extensible Markup Language 340
- extension 263
- extension attribute properties
 - Deprecated 643
 - Description 644
 - Extends 643
 - Implements 644
 - Name 643
 - Restrictions 644
 - Translatable 644
 - Type 643
 - Use 643
 - Value 644
- extension-point
 - id 640
 - name 640
 - schema 640

Extension points

- <code> 642
- <pre> 642
- <samp> 642, 646
- abstract base class vs. interface 647
- Class.forName 655
- cleanup 655
- coding 649
- com.qualityeclipse.favorites.popupMenu 267
- createExecutable 655
- createExecutableExtension 652
- defined 118
- defining 639
- dispose 655
- documentation 656, 659
- documentation formatting 642
- editing extensions 81
- element grammar 647
- elements and attributes 643
- exception handling 651
- executable extensions 653
- extension example 119
- fast fail 652
- getConfigurationElementsFor 128
- getExtensionPoint 128
- introduction 119
- lazy initialization 645, 652
- log errors 660
- manifest editor page 82
- mechanism 637
- org.Eclipse.actionSetPartAssociations 256
- org.eclipse.core.expressions.propertyTesters 225
- org.eclipse.core.resources.builders 535, 538
- org.eclipse.core.resources.markers 550
- org.eclipse.core.resources.natures 562
- org.eclipse.help.contexts 596–597
- org.eclipse.help.toc 586
- org.eclipse.ui.actionSets 242–243
- org.eclipse.ui.command 218
- org.eclipse.ui.decorators 802–803
- org.eclipse.ui.editorActions 280
- org.eclipse.ui.editors 356
- org.eclipse.ui.exportWizards 469
- org.eclipse.ui.handlers 236
- org.eclipse.ui.ide.markerHelp 598
- org.eclipse.ui.ide.markerResolution 556
- org.eclipse.ui.ide.projectNatureImages 567
- org.eclipse.ui.importWizards 470
- org.eclipse.ui.menus 220
- org.eclipse.ui.newWizards 470–471, 485
- org.eclipse.ui.perspectiveExtensions 430, 434, 437
- org.eclipse.ui.perspectives 423, 425–426
- org.eclipse.ui.popupMenus 257
- org.eclipse.ui.preferencePages 485
- org.eclipse.ui.propertyPages 516
- org.eclipse.ui.startup 121, 421, 816
- org.eclipse.ui.viewActions 273
- org.eclipse.ui.views 81, 291
- overview 637
- page 639
- parsing 649
- perspectives 424
- Platform 126
- proving enhancements 783
- proxies 651
- Registry 127
- RFRS 659
- schema 639, 641
- search 98
- Show only Extension points from required** 596
- unique identifier 639
- usage example 109
- using 657
- wizards 469
- Extension Points** page 640
- extensions 126
 - search 98
- extensions, editor 356
- Extensions** page 81–82, 217, 220, 225, 236, 243, 280, 291, 356, 425, 431, 434, 437, 471, 515, 535, 550, 562, 595, 597, 605, 656, 658
- extension, viewerContribution 270–271
- Extension Wizard** page 584
- Externalize Strings strings to externalize list 621
- Externalize Strings** wizard
 - class name field 624
 - common prefix 622
 - Configure** button 624
 - described 621
 - error messages 625
 - Ignore** button 624
 - Internalize** button 624
 - package field 624
 - property file name field 624
 - proposed changes 626
 - string substitution pattern 625
- External Plug-ins 95
- External Programs** radio button 49
- External Tools** command 12
- Extract Class** command 44
- Extract Constant** command 43
- Extract Constant** refactoring 507
- extracting strings from Java source 620
- Extract Interface** command 44
- Extract Local Variable** command 43
- Extract Method** command 43
- ExtractStringsWizard 473, 475
- Extract Strings** wizard 475
- Extract Superclass** command 44

F

- F1 key 143, 591, 599
- F3 key 27, 61
- F6 key 61
- F8 key 61
- Failures** page 64
- fast fail, extension points 652
- fast view bar 6, 8
- Fast View** command 6
- FavoriteAdapterFactory 786–787
- FavoriteConnectionEditPart 751
- FavoriteDefaultsPreferencePage 529
- FavoriteItemFactory
 - class 651, 655
 - extends 659
- FavoriteItemPropertyPage
 - class 518, 521–522
 - extends 529
- FavoriteItemType 297, 649–651, 657, 659
- FavoriteJavaElement 296, 304, 337
- FavoriteResource 296, 302, 337, 659
- FavoriteResourcePropertyPage 515, 517–520
- FavoritesActivator 83, 343, 345, 407, 514–515, 786–787, 817
- FavoritesActivator.PLUGIN_.ID 650
- FavoritesActivator.PLUGIN_ID 514
- FavoritesActivatorPLUGIN_.ID 83
- favoritesChanged 413
- FavoritesCheatSheet.xml file 603, 610
- FavoritesDragSource 327–328
- FavoritesDropTarget 327, 329
- FavoritesEditPartFactory 755
- favorites.exsd file 641
- FavoritesFilterDialog 450
- FavoritesGEFView 755–756
- favoritesItemChanged 512, 524
- FavoritesLightweightDecorator 804
- FavoritesLog 344, 418
 - class 129, 339, 813
 - logError 251
- FavoritesManager 296, 299, 301–302, 306, 313, 343, 407–408, 411–412, 512, 524, 755–756, 786–787
- FavoritesManagerEditPart 756
- FavoritesManagerEvent 302, 805
- FavoritesManagerListener
 - implements 306, 804
 - interface 302
- FavoritesPerspectiveFactory 427
- Favorites plug-in 71
- FavoritesPreferencePage 496
- FavoritesResourcePropertyPage 517
- FavoritesTester 312
- FavoritesTestSuite 255
- FavoritesTransfer 331
- FavoritesView 85, 315, 341, 508, 524, 787
- Favorites** view 84, 105
- FavoritesViewContentProvider 313
- FavoritesViewFilterAction 319, 341, 453
- FavoritesView.ID 253
- Favorites** view, identifier 238, 250
- FavoritesViewLabelProvider 308
- FavoritesViewLocationFilter 454
- FavoritesViewNameFilter 311, 341
- FavoritesViewSorter 309–310, 340
- FavoritesViewTest 101, 103, 253
- FavoritesViewTypeFilter 454
- feature** 684
 - building 673
 - exporting 673
 - testing 673
- feature**
 - copyright 665
 - description 665
 - id 665
 - label 665
 - license 665
 - plugin 666
 - provider-name 665
 - url 665
 - version 665
- Feature Details** command 675
- Feature License** page 688
- Feature** manifest 665
- Features**
 - about.html 674
 - about.ini 675
 - about.mappings 676
 - about.properties 676–677
 - branding 673
 - Branding Plug-in** field 667
 - Build Configuration** button 668
 - component diagram 662
 - Copyright Notice** tab 669
 - Copy versions** 668
 - creating a new project 663
 - exclusive installation option 672
 - Exporting** section 668
 - Export Wizard** button 668
 - Feature Description** tab 669
 - feature image 675–676
 - Feature Name** field 664, 671
 - Feature Provider** field 664
 - Feature Version** field 664
 - feature.xml file 665
 - Force feature version** 668
 - License Agreement** tab 669
 - License** page 688
 - manifest 665
 - manifest editor 666–667
 - Operating Systems** section 668
 - Optional URL** field 669

- plug-in_customization.ini 678
- plugin_customization.properties 678
- primary feature files 674
- project 662
- Provider Name** field 667
- referenced plug-ins and fragments 665
- RFRS 689
- Sites to Visit** tab 669
- splash screen 679
- subfeatures 670
- supported environments 667
- Supported Environments** section 667
- supported operating systems 668
- Synchronize** button 668
- synchronize versions on build 668
- Synchronize versions on build** option 668
- Text** field 669
- update and discovery URLs 685
- Update Site Name** field 667
- update sites 679
- Update Site URL** field 667, 685
- Version** field 667
- versions of included plug-ins 666, 668
- features directory 680, 683
- Feature Selection** dialog 681
- feature.xml
 - file 727
- feature.xml file 665–666
- FieldEditor
 - class 490
 - IS_VALID 497
 - VALUE 497
- FieldEditorPreferencePage
 - assumptions 494
 - class 487, 489–490, 492, 500
 - extends 496
- Field Editors
 - article 509
 - Boolean 490
 - Color 490
 - create and forget 492
 - described 490
 - Directory 490
 - File 491
 - Font 491
 - Integer 491
 - Path 491
 - Radio Group 491
 - Scale 491
 - String 491
- File > Export...** command 87, 469
- File > Import...** command 470
- File > New > Class** command 24
- File > New > Folder** command 603
- File > New > Other...** command 470, 473
- File > New > Package** command 24
- File > New > Project...** command 19
- File Associations** preferences 16, 47, 506
- FileDialog 442, 480
- FileEditorInput 560
- File Extension Associations 47
- FileFactory 659
- FileFieldEditor 491
- File IO, example 544
- FileLocator 121
 - find 122
 - openStream 122
 - resolve 122
 - toFileURL 122
- File** menu 11
- File name patterns** field 29
- filenames, editor 357
- FileNotFoundException 343
- FileReader 343
- Files
 - hidden 22
 - reading and writing 343
- File Search 28
- File Search** page 28
- FileTransfer 330
- File types** list 47
- FileWriter 344
- fillContextMenu 316–317, 325, 382, 398, 522
- FillLayout
 - class 178–179, 206
 - example 179
 - type 179
- filter
 - name 519
 - value 519
- FILTER_ID_EXPERT 528
- Filter** dialog 450
- Filtering actions based on file content 265
- Filtering files 22
- Filters
 - hierarchy 198
 - view 311
 - viewer 197
- Filters...** command 22
- find 121
- findMarkerDeltas 409
- findMember 303, 410
- findView 254
- findWidget 149
- fireFavoritesItemChanged 512
- fireStateChanged 492–493
- fireValueChanged 492–493
- Flags 813
- Focus event 143
- FocusListener 145
- FontData 189
- FontDialog 442

- FontFieldEditor 491
 - Fonts 189
 - Fonts preferences** 15
 - forcePluginActivation 228
 - forgetLastBuildState 541
 - forgetLastBuiltState 539
 - format 624
 - FormAttachment
 - alignment 186
 - class 185–187, 481
 - control 186
 - denominator 186
 - numerator 186
 - offset 186
 - Formatter preferences** 17, 36
 - FormData
 - bottom 186–187
 - class 185–187, 481
 - height 186
 - left 186–187
 - right 186–187
 - top 186–187
 - width 186
 - FormEditor 358
 - FormLayout
 - class 185, 481
 - example 187
 - marginHeight 185
 - marginWidth 185
 - Smalltalk origins 178
 - forName 655
 - Fragment Content page** 632
 - Fragment-Host 634–635
 - fragment, plugin 666
 - Fragments
 - accessing internal classes 629
 - accessing internal code 784
 - attribute 666
 - definition 784
 - dependencies 635
 - Fragment Project wizard** 630
 - internationalization 784
 - lifecycle 634
 - manifest 633
 - manifest editor 633
 - merged with base plug-in 629
 - new project 630
 - project content 635
 - referenced plug-ins and fragments 665
 - used to support different Eclipse versions 629
 - using 629
 - version match rules 671
 - fragment.xml
 - comparing versions 671
 - file 633
 - FULL_BUILD 539, 548, 566, 570
- G**
- GEF 132, 731
 - activate 734
 - active tool 761
 - Architecture 731
 - child figures 747
 - Command 734
 - command stack 761
 - connection 756
 - connection source 756
 - connection target 756
 - deactivate 734
 - DefaultEditDomain 761
 - edit domain 761
 - EditPart 734
 - edit part factory 755–756
 - EditParts 734
 - EditPolicies 734
 - figure not visible 756
 - focus 756
 - Graphical Editing Framework 731
 - Introduction 731
 - Label 750
 - layout 756
 - root edit part 755
 - Tooltips 750
 - view 755
 - Generalize Type command** 44
 - General preferences** 14–15
 - Generate a new Java class 804
 - getActionBars 318, 377, 385–386, 397, 399, 419
 - getActiveEditor 397
 - getActivePage 101, 251, 253, 269, 361, 381, 386, 569
 - getActivePart 334, 799
 - getActiveShell 149, 322
 - getActiveWorkbenchWindow 101, 253, 269, 418, 464, 473, 569, 789
 - getAdapter 294, 304, 306, 323, 337, 339, 394–395, 397, 400, 417, 785–786, 805
 - getAdapter, IPropertySource 524
 - getAdapterList 786
 - getAdapterManager 304, 306, 337, 787
 - getAffectedChildren 410
 - getAttribute 553–554, 560, 650–653
 - getAttributes 554
 - getBackground 523
 - getBoolean 503, 507, 810
 - getBooleanValue 497
 - getBounds 149
 - getBrowserSupport 789
 - getBuilderName 546–547
 - getBuildKind 409
 - getBuildSpec 546–547
 - getBundle 126–127
 - getBundleGroupProviders 126–127

- getBundleId 127
- getCellEditor 335, 373, 376–377
- getChecked 164, 167, 205, 209
- getCheckedElements 205, 209, 482
- getChild 341
- getChildren 169, 171, 196, 211, 734
- getClientArea 148–149, 462
- getCollator 197
- getColor 512, 523, 528
- getColorPropertyValue 529
- getColumn 162
- getColumnImage 86, 195, 207, 807
- getColumns 162
- getColumnText 86, 102, 195, 207, 373, 807
- getColumnViewerEditor 336, 378
- getCommand 540, 734
- getCommentPropertyValue 519–520, 529
- getConfigurableElements 650
- getConfigurableElementsFor 128
- getConfigurableLocation 123
- getContainer 359, 465, 467, 474
- getContentProvider 102
- getContextMenuIds 799
- getControl 173, 200, 317, 373, 376–377, 593, 755
- getCurrent 148, 513
- getCurrentPage 468
- getCurrentSelection 313, 315, 323, 397, 417, 473
- getCursorControl 149
- getCursorLocation 149
- getData 145, 149
- getDeclaringExtension 650, 653–654
- getDeclaringType 813
- getDecoratorManager 807
- getDefault 84, 129, 148, 413, 508
- getDefaultBoolean 503
- getDefaultColor 512, 528
- getDefaultDouble 503
- getDefaultFloat 503
- getDefaultInt 503
- getDefaultLong 503
- getDefaultPageImage 465
- getDefaultString 503
- getDelta 409, 412, 540
- getDialogBoundsSettings 450
- getDialogSettings 125, 450, 466–467
- getDisplay 146, 151, 322, 462
- getDisplay 146, 151, 322, 462
- getDocument 380, 560
- getDocumentProvider 380, 560
- getDouble 503
- getDoubleClickTime 150
- getEditableValue 528
- getEditorArea 428, 430
- getEditorInput 380
- getEditorSite 377, 397, 399, 419
- getElement 520
- getElementAt 200, 204
- getElements 102, 196, 199
- getEntry 127
- getErrorMessage 377
- getExpanded 167
- getExpandedElements 208
- getExpandedState 208
- getExtensionPoint 128, 650
- getExtensionRegistry 127–128, 650
- getExtensions 650
- getFactory 654
- getFieldEditorParent 489, 496
- getFirstElement 334
- getFlags 410, 813
- getFloat 504
- getFocusControl 150
- getForeground 523
- getFullPath 306, 410
- getFullyQualifiedName 813
- getGrayed 205, 209
- getGrayedElements 205, 209
- getHeaders 127
- getHelpSystem 593, 599
- getHorizontalBar 148
- getId 386, 388
- getImage 86, 195, 203, 448, 457, 653
- getImageDescriptor 84, 653
- getImageRegistry 126, 347
- getInfo 296
- getInitialLocation 449
- getInitialSize 449
- getInt 504
- getItem 157, 160, 162, 172, 175
- getItemCount 157, 160, 166–167, 172, 175
- getItems 157, 160, 167, 173, 175
- getJavaElementClipboardTransfer 326, 329–330
- getJobManager 103, 126
- getKind 410
- getLabelControl 492
- getLabelFor 393–394
- getLabelProvider 102
- getLabelText 492
- getLayer 756
- getList 200
- getLocation 478, 480, 808
- getLog 127, 129
- getLong 504
- getManager 326, 524, 755
- getMarkerDeltas 410
- getMenuIds 798
- getMessage 457
- getModel 734
- getModelChildren 734, 756
- getMovedFromPath 410
- getMovedToPath 411
- getNamespace - deprecated,
 see getNamespaceIdentifier

- getNamespaceIdentifier 650, 653
- getNextPage 466
- getNumberOfControls 493
- getOperationHistory 397–399
- getOperationSupport 398
- getPage 385
- getParent 146, 167, 176, 196, 211, 735
- getParentItem 167, 175
- getParentMenu 175
- getPersistentProperty 514
- getPlatform 791
- getPluginPreferences 123, 810
- getPreferenceName 493
- getPreferenceStore 122–123, 493–494, 496, 503, 505–508, 515, 817
- getPreviousPage 466
- getProduct 127
- getProgressMonitor 397, 419
- getProject 269, 540
- getProjectRelativePath 411
- getProperty 320, 323
- getPropertyDescriptors 525
- getPropertyValue 525
- getResolutions 558
- getResolvedClasspath 814
- getResource 306, 409, 411
- getResult 444–445
- getRGB 522, 528
- getRoot 269, 303, 478, 480
- getRuntime 791
- getSection 450, 475
- getSelectedPage 469
- getSelectedRange 212
- getSelection 152, 157–158, 162, 167, 173, 176, 199, 212, 322, 334, 569
- getSelectionCount 157, 162, 167
- getSelectionIndex 157, 160, 162, 173
- getSelectionIndices 157, 162
- getSelectionText 155
- getService 315, 321
- getSessionProperty 514
- getSeverity 457
- getSharedImages 297, 315
- getShell 146, 151, 464, 468, 473
- getShells 150–151
- getSite 337, 382, 419, 593
- getSource 409
- getStartingPage 466
- getState 127
- getStateLocation 122, 127, 345
- getStatusHandler 131
- getStatusLineManager 377, 397, 419
- getString 341, 504
- getSymbolicName 128, 459
- getSystemColor 150, 189
- getSystemFont 150
- getTable 204, 206, 307, 413, 481
- getTargetException 456–457
- getText 155, 160, 164, 195, 203
- getTextWidget 212
- Getting started 1
- Getting started with CVS 50
- getTitle 456
- getToolBarManager 318
- getType 409
- getUnderlyingResource 305–306, 477
- getUndoContext 397–398
- getURL 123
- getVerticalBar 148
- getViewLayout 430
- getViewSite 86, 318, 321, 419
- getVisibleExpandedElements 208
- getWizard 467
- getWorkbench 86, 101, 253, 269, 297, 315, 464, 569, 593, 599, 789, 807
- getWorkbenchWindow 419, 593
- getWorkspace 269, 303, 408, 415, 478, 480
- Go to Line** command 27
- gotoMarker 359, 361
- Graphical Editing Framework, *see* GEF
- GridData
 - BEGINNING 451
 - CENTER 451
 - class 183–184, 451
 - END 451
 - FILL 184, 451
 - FILL_BOTH 458–459, 519, 521
 - FILL_HORIZONTAL 460, 476
 - grabExcessHorizontalSpace 183–184
 - grabExcessVerticalSpace 183
 - heightHint 183
 - HORIZONTAL_ALIGN_CENTER 460
 - HORIZONTAL_ALIGN_END 476
 - HORIZONTAL_ALIGN_FILL 477
 - horizontalAlignment 183–184
 - horizontalIndent 183
 - horizontalSpan 183–184
 - VERTICAL_ALIGN_BEGINNING 460
 - verticalAlignment 183
 - verticalSpan 183
 - widthHint 183
- GridLayout
 - class 182, 184, 448, 451
 - example 184, 521
 - horizontalSpacing 182
 - makeColumnsEqualWidth 182
 - marginHeight 182, 519, 521
 - marginWidth 182, 519, 521
 - numColumns 182, 184, 521
 - verticalSpacing 183
- Group
 - class 171
 - example 172
- GroupMarker 317–318

groupMarker 267, 274

GUI Builder

Swing Designer 825, 827

SWT Designer 825, 827

H

handlePropertyChange 379

Handler

activeWhen 324

convert Action to handler 237

enabledWhen 324

HandlerUtil 237

handler 324–325, 334

Handlers 236

HandlerService 321

HandlerUtil 313, 315, 322–323, 334, 397, 417–418, 473, 789

hasBuilder 570

hasChildren 196, 211

hasHelpUI 594

hasNature 564, 569, 571

hasResolutions 558

HEAD, CVS 50, 798

Headless Eclipse, modifying resources 414

Help

active 580

anchor points 587

bottom-up composition 587–589

Cheat Sheets menu 601

cheat sheets 601

content 589, 595

context element 595

context identifiers 592

context-sensitive 591

contexts.xml 595, 598

description element 595

displayHelp 599

displayHelpResource 599

dynamic 580

Dynamic Help 591

Eclipse anchor points 588

extension points 595

F1 key 591

files 586

generating email 601

Getting started 585

Help Contents menu 577

helpContextId 246, 259

href attribute 588

HTML 580

implementing 580

infopop 591

internationalization 589

Label for table of contents field 583

link_to attribute 588

markers 598

nl/<language> 589

nl/<language>/<country> 589

PDF 580

plug-in manifest 585

plug-ins 580

Primary option 583

primary toc files 583, 588

programmatically open **Help** view window 599

programmatically opening help page 599

programmatically opening Web page 600

project 581

Reference 585

RFRS 611

searching 578

search scope 579

secondary toc files 588

Select Search Scope dialog 579

setHelp 592

setHelpContextIDs 592

stemmed search 578

table of contents 586

tocgettingstarted.xml 585, 587

tocreference.xml 585

toc.xml 585

top-down composition 588–589

top-down nesting 587

topic elements 595

translated files 589

using 577

using a cheat sheet 601

WorkbenchHelp 594, 599

XHTML support 580, 591

ZIP file 590

Help > About Eclipse SDK command 114

Help > Cheat Sheets... command 601

Help > Help Contents command 113, 119, 577, 599

Help > Software Updates... 685

Help > Tips and Tricks command 676

Help > Welcome command 4

helpContextId 259

helpContextId, action 246, 259, 272, 282, 593

Help event 143

HelpListener 146, 175–176, 199

Help menu 12

Help view 597

Help window 577

hidden files 22

hideView 102, 254

Hierarchy view 591

History view 46, 53

hookDragAndDrop 327, 329

hookGlobalHandlers 321

hookGlobalTreeActions 387

hookKeyboard 321

hookMouse 339

hookPageSelection 338

horizontalIndent 451

- Host field 50, 797
- hot code replacement 59
- hoverIcon, action 246, 282
- How Eclipse is different 783
- I**
- i18n 617
- i18n, *see* Internationalization
- IAction
 - interface 241, 251, 594
 - setEnabled 251
 - vs. IActionDelegate 240
- IActionBars 385–386, 388, 399
- IActionDelegate 241–242, 414
- IActionDelegate2 242
- IActionFilter 263, 519
- IActionSetDescriptor 256
- IAdaptable 296, 304, 306, 317, 323, 337, 339, 394, 397, 417, 518, 569, 785–786, 788
- IAdapterFactory 786
- IBaseLabelProvider 200, 204
- IBM 135–136, 191, 831
- IBM Alphaworks on Eclipse 830
- IBM Business Partners 831
- IBM Eclipse Research 830
- IBM middleware 831
- IBM Smalltalk 135
- IBM Software Development Platform xxxix, xli
- ICellEditorListener 377
- ICellEditorValidator 376, 527
- ICellModifier
 - interface 204
- ICheckStateListener 204
- IClasspathEntry 814
- IColorDecorator 802
- IColorProvider 308, 523
- ICommand 546–547
- ICompilationUnit 811
- icon
 - action 246, 259, 272, 282, 790, 793
 - command 221–222, 224, 229
 - decorator 803, 806
 - editor 356
 - image 567
 - page 517
 - perspective 426
 - view 119, 293, 618
 - viewContribution 273
 - wizard 470
- IconAndMessageDialog 443
- Icon creation 248
- IConfigurationElement 650–652, 655
 - class 540, 793
 - createExecutable 655
 - createExecutableExtension 652
 - getAttribute 652
- icons directory 110
- IContentDescriber
 - class 357
 - INDETERMINATE 358
 - INVALID 358
- IContentOutlinePage 400–401
- IContentProvider 199
- IContext 594
- IContextProvider 294, 597
- IContributedContentsView 294
- id
 - action 245, 259, 264, 272, 279, 281, 790, 793
 - actionSet 244
 - category 292, 470, 605, 618
 - cheatsheet 605–606
 - command 219, 221–222, 763
 - command category 219
 - decorator 803, 806
 - editor 356, 761
 - editorContribution 280
 - extension-point 640
 - feature 665
 - image 567
 - itemType 642
 - menu 220, 244, 267, 271, 274, 278, 280
 - objectContribution 257
 - page 486, 498–499, 516, 530
 - perspective 425–426
 - plugin 666
 - toolbar 222
 - view 293, 432, 434, 618, 755
 - viewContribution 273–274
 - viewerContribution 271
 - visibility 278
 - wizard 470, 472
- IDE 339, 560
- IDecoratorManager 807
- IDelayedLabelDecorator 802
- identifiers 248
- IDialogConstants
 - DETAILS_ID 461
 - HIDE_DETAILS_LABEL 461
 - HORIZONTAL_MARGIN 458
 - MINIMUM_MESSAGE_AREA_WIDTH 460
 - OK_ID 461
 - OK_LABEL 461
 - SHOW_DETAILS_LABEL 461
- IDialogSettings 450, 475
- IDocument 212, 560
- IDoubleClickListener 198, 202
- IEditorActionBarContributor 356, 385, 387
- IEditorActionDelegate 242, 279, 281
- IEditorInput 378
- IEditorLauncher 357
- IEditorPart 397, 800
 - hierarchy 354, 412
 - interface 353, 356, 358, 463, 800
 - PROP_DIRTY 378
- IEditorReference 353, 799–800

- IEditorSite 353, 378
- IElementComparer 199
- IEvaluationContext 315
- IExecutableExtension 292, 470–471, 487, 794–796
- IExportWizard 469
- IExtension 653
- if, Ant target 694
- IFavoriteItem 335
 - implements 304
 - interface 296, 521, 654–655, 659, 785–786
- IFavoritesListener 512, 524
- IFigure 750
- IFile 339, 478, 519, 659
- IFolder 519
- IFolderLayout 428
- IFontDecorator 802
- IFontProvider 308
- IGotoMarker 361
- IHandler 237, 316, 473
- IHandlerService 236, 315, 321
- IImportWizard 470
- IJavaElement 224–225, 304, 330, 786–787
- IJavaModelStatus 130
- IJavaProject 814
- ILabelDecorator 802
- ILabelProvider
 - implementors 195
 - interface 195, 200, 204, 445, 527
- ILabelProviderListener 807
- ILightweightLabelDecorator
 - implements 804
 - interface 803–804, 806
- ILlegalAccessException 783
- ILog 129
- Image
 - class 152, 164, 457
- image
 - icon 567
 - id 567
 - natureId 567
- ImageCache 347, 653
- ImageData 189
- ImageDescriptor 83–84, 190, 346, 466–467, 651, 804
- imageDescriptorFromPlugin 84, 653, 804
- ImageRegistry 347
- Images
 - caching 346
 - createImageRegistry 125
 - files 110
 - getImageRegistry 126
 - initializeImageRegistry 126
 - lazily loading 347
 - overlay 567
 - relation to ImageDescriptor 346
 - SWT resources 189
- images directory 110
- IMarker
 - BOOKMARK 549
 - CHAR_END 552, 555–556
 - CHAR_START 552, 555–556
 - DONE 555
 - gotoMarker 359
 - LINE_NUMBER 555
 - LOCATION 556
 - MARKER 550
 - MESSAGE 552, 556
 - PRIORITY 556
 - PROBLEM 550, 561
 - SEVERITY 556
 - SEVERITY_ERROR 552
 - SEVERITY_WARNING 552
 - TASK 550
 - TRANSIENT 556
 - USER_EDITABLE 556
- IMarkerResolution2, implements 559
- IMarkerResolutionGenerator 558
- IMarkerResolutionGenerator2
 - implements 558
 - interface 558
- IMemento 340–342, 344–345
- IMenuListener 316
- IMenuManager 316–317, 320, 382, 388, 398
- IMethod 811, 813
- Import > As Binary Project** command 780
- Import > Preferences** wizard 18
- Import As > Source Project** command 797
- Imported-Packages 80
- Importing and Exporting Preferences 18
- Import-Package 118
- Included Features** page 670
- includeEmptyDirs 697
- INCREMENTAL_BUILD 539
- IncrementalProjectBuilder
 - AUTO_BUILD 539
 - class 538
 - CLEAN_BUILD 539
 - example 540
 - extends 540
 - FULL_BUILD 539, 548, 566, 574
 - INCREMENTAL_BUILD 539
 - methods 539
- inDebugMode 127
- index.html file 680
- indexOf 157, 160, 162, 173, 175
- INewWizard 470, 472–473
- Infer Generic Type Arguments** command 44
- Infopop 591
- Information** page 669
- inheritAll 696
- init 340–342, 359, 378, 385, 388, 477, 489
- initContent 453

- initContents 477
- initialize 489
- initializeDefaultPluginPreferences 123, 505
- initializeDefaultPreferences 505–506
- initializeImageRegistry 126
- initializeMessages 627
- initTreeEditors 376, 378, 388
- initUndoRedo 398
- Inline** command 43
- Inline editing, views 333
- inputChanged 196, 199
- InputDialog
 - class 319–320, 445
 - OK 320
- insert 155, 204
- Insertion points 248
- Inspect** command 62
- Installation
 - hybrid approach 113
 - instructions 92
 - link files 111
 - location 126
 - ZIP files 113
- Installation Details** page 672
- Install** button 687
- Installed JREs** preferences 17, 57
- Installing Eclipse 3
- install-size, plugin 666
- Install wizard** 687
- instanceof 262
- InstanceScope 503
- Instantiations xxxviii, xlv
- IntegerFieldEditor 491
- Interfaces, naming convention 296
- interface vs. abstract base class, extension points 647
- Internal code
 - accessing 781
 - checking Bugzilla 782
 - checking newsgroup 781
 - definition 781
 - fragments 784
 - options for accessing 782
 - restrictions 783
- Internationalization
 - //NON-NLS-1\$ 623, 627
 - binding patterns 624
 - end-of-line comments 623
 - externalizing the plug-in manifest 618
 - extracting strings 620
 - factor out common values 628
 - formatting strings 624
 - French translation 620, 635–636
 - German translation 619, 635–636
 - include error numbers 628
 - ISO 3166, country codes 619
 - ISO 639, language codes 619
 - keep keys similar to text 628
 - log missing keys 629
 - plugin.properties 618, 635
 - plugin.xml 618
 - remove colons 628
 - remove punctuation characters 628
 - replacing strings with field references 623
 - sort .properties file entries 628
 - suggestions 628
 - using fragments 629, 784
- InterruptedException 418, 474
- Introduce Factory** command 44
- Introduce Indirection** command 44
- Introduce Parameter** command 44
- Introduce Parameter Object** command 44
- Invalid Menu Extension 249
- Invalid thread access 413
- InvocationTargetException 418, 456–457, 474
- IObjectActionDelegate 242, 266, 269, 463
- IOException 333, 344, 791
- IOpenListener 199
- IOperationApprover 400
- IOperationHistory 393
- IPageChangedListener 469
- IPageLayout
 - BOTTOM 428
 - ID_OUTLINE 428
 - ID_PROBLEM_VIEW 428
 - ID_TASK_LIST 428
 - interface 428–429
 - LEFT 428
 - methods 429
- IPath 121–122, 479–481, 814
- IPerspectiveFactory
 - implements 427
 - interface 423, 426
- IPreferenceConstants 817
- IPreferenceStore 493, 503
- IProgressMonitor 359, 394–395, 397, 414–416, 418, 474, 539, 541, 553, 809
- IProgressMonitorWithBlocking 416
- IProgressService 419, 811
- IProject 269, 478, 540, 548
- IProjectDescription 546–547, 571
- IProjectNature
 - implements 565
 - interface 565
- IPropertyChangeListener 493, 508, 810
- IPropertyDescriptor 526, 528
- IPropertySheetEntry 528
- IPropertySource
 - interface 524
 - methods 525
- IPropertySource2 526–527
- IRegistryChangeListener 128

- IResource 224–225, 323, 330, 417, 517
 - adaptable to 517
 - DEPTH_INFINITE 552, 561
 - interface 302, 518, 520, 574, 785–786, 803, 805
 - properties 513
 - setDerived 545
- IResourceActionFilter 263
- IResourceChangeEvent 408
 - interface 408, 412
 - POST_AUTO_BUILD 408
 - POST_CHANGE 408, 411
 - PRE_AUTO_BUILD 408
 - PRE_CLOSE 408
 - PRE_DELETE 408
 - PRE_REFRESH 408
- IResourceChangeListener
 - implements 408
 - interface 407
- IResourceDelta
 - ADDED 409
 - ADDED 411
 - ADDED_PHANTOM 409
 - ALL_WITH_PHANTOMS 409
 - CHANGED 409, 411
 - CONTENT 409
 - COPIED_FROM 409
 - DESCRIPTION 409
 - ENCODING 409
 - interface 409–410, 541
 - LOCAL_CHANGED 409
 - MARKERS 409
 - MOVED_FROM 409
 - MOVED_TO 410
 - NO_CHANGE 410
 - OPEN 410
 - REMOVED 410–412
 - REMOVED_PHANTOM 410
 - REPLACED 410
 - SYNC 410
 - TYPE 410
- IResourceDeltaVisitor 412
- IResourceProxy 574
- IResourceProxyVisitor 574
- IResourceStatus 130
- IResourceVisitor 574
- IRunnableContext 417
- IRunnableWithProgress 418
 - example 418
 - interface 417–418, 468, 474
- ISafeRunnable 127
- ISaveContext 420
- ISavedState 420
- ISaveParticipant
 - interface 420
 - snapshot 345
- isCanceled 415, 474
- isCellEditorActive 204
- isContextHelpDisplayed 594
- isDefault 504
- isDigit 156
- isDirty 359, 378–379
- isDisposed 139, 145–146, 154
- isEditable 212
- ISelection 200, 212, 315, 323, 339, 397, 473, 787
- ISelectionChangedListener 199, 202, 314
- ISelectionListener 338
- isEmpty 322, 334
- isEnabled 146, 151, 176
- isExpandable 208
- isFilterProperty 198
- ISharedImages 315
 - interface 86, 297
- IShellProvider 456, 460
- IShowInSource 294
- IShowInTarget 294
- IShowInTargetList 294
- isInstance 337
- isInterrupted 540
- ISO
 - 3166 619
 - 639 619
 - country codes 619
 - language codes 619
- isPropertyResetable 526, 528
- isPropertySet 526, 528
- isPublic 813
- isResizable 450
- isSaveAsAllowed 359, 361
- isSorterProperty 197
- isSupportedType 328, 330
- IStartup 121, 421
- IStatus
 - CANCEL 129
 - ERROR 129
 - Handling 131
 - INFO 129
 - interface 129–130, 444, 457
 - OK 129, 457
 - WARNING 129
- IStatusHandler 131
- IStatusLineManager, progress monitor 419
- IStructuredContentProvider
 - implementors 196
 - implements 306
 - interface 85, 102, 196, 306, 445
- IStructuredSelection 312–313, 315, 324, 334, 338, 397, 417, 787
 - EMPTY 254
 - interface 339
- isValid 376, 489, 493–494
- isVisible 147
- ITableLabelProvider

- implementors 195
 - implements 308, 370
 - interface 86, 102, 195, 204, 207
 - Item 145
 - item, cheatsheet 610
 - itemType
 - class 642
 - id 642
 - name 642
 - targetClass 642
 - iterate 225
 - iterator 339
 - ITextContentDescriber 357
 - ITextEditor 560
 - ITextHover 212
 - ITextListener 211
 - IToolBarManager 318, 388
 - ITreeContentProvider
 - implements 369
 - interface 196, 211
 - ITreeViewerListener 207
 - IType 811
 - IUndoContext 398
 - IViewActionDelegate 242, 271, 273
 - IViewerLabelProvider 308
 - IViewPart 119, 254, 289, 292–293, 463
 - IViewReference 289, 800
 - IViewSite 289, 342
 - IWebBrowser 789
 - IWizard 465
 - IWizardContainer 446
 - IWizardNode 469
 - IWizardPage 466–468
 - IWorkbenchActionConstants
 - ME_ADDITIONS 248, 317, 382, 388
 - IWorkbenchActionDefinitionIds 322
 - interface 321, 324
 - IWorkbenchAdapter 308, 788
 - IWorkbenchBrowserSupport 788–789
 - IWorkbenchHelpSystem 592–594, 599
 - IWorkbenchPage 251, 289, 339, 353, 385, 388, 799
 - IWorkbenchPart 334, 338
 - hierarchy 354, 412
 - interface 290, 463, 787, 800
 - IWorkbenchPartReference 800
 - IWorkbenchPreferencePage
 - implements 496, 529
 - interface 487, 529
 - IWorkbenchPropertyPage 517
 - IWorkbenchWindow 418, 473, 789
 - from editor or view 419
 - interface 250, 418–419
 - IWorkbenchWindowActionDelegate
 - interface 242, 246, 463
 - IWorkbenchWindowPulldownDelegate 246
 - IWorkspace
 - interface 548
 - run 414
 - IWorkspaceRunnable 415, 552–553
- J**
- Jalopy 36
 - jar file
 - pack 714
 - packing 697
 - sign 714
 - signing 697
 - JAR files 7, 116
 - Java
 - Foundation Classes (JFC) 136
 - garbage collector 347
 - use at IBM 136
 - Java Applet** configuration 58
 - Java Attribute Editor** dialog 538
 - Java Browsing** perspective 7–8
 - Java Build Path 23, 38, 63, 80, 780, 814
 - Java Class** wizard 24
 - Java Compiler 628
 - java compiler build settings 717, 720
 - javacSource 717, 720, 722
 - javacTarget 717, 720, 722
 - Java Editor
 - code assist 38
 - code formatting 36
 - Color syntax highlighting 36
 - Organizing Import Statements 37
 - using 35
 - JavaElementTransfer 330
 - Java History** view 26
 - JavaModelException 130, 305–306, 477, 813–814
 - Java Package** wizard 24
 - Java** perspective 7, 13, 19
 - Java** preferences 17
 - Java projects 19
 - Java Project** wizard 19
 - Java Runtime Environment 3
 - Java Search 30
 - javaToNative 332
 - JavaUI 326, 329–330
 - JDK 59
 - JFace Viewers 193
 - JFC 136
 - JIRA 65
 - Job 126
 - getJobManager 103
 - JobManager 126
 - Jobs API 808
 - JRE 3, 57, 59
 - JUnit 63, 99, 101, 106, 113, 253, 268, 275, 283, 345
 - Content Assist 346
 - JUnit Assert 346
 - JUnit** configuration 58

JUnit Plug-in test command 103
JUnit Plug-in Test configuration 59
JUnit Test Case wizard 63
JUnit view 64, 103

K

key

- commandId 239
- contextId 239
- locale 240
- platform 240
- schemeId 239
- sequence 239

KeyAdapter 321
Key bindings

- declaration 238

Keyboard

- commands 320

Keyboard accessibility 285
keyCode 336
KeyEvent 321, 388
Key event 143
KeyListener 146
keyPressed 388
keyReleased 389
Keys preferences 16, 238
Koi xlv

L

Label 151, 451, 459
label

- action 245, 259, 264, 272, 274, 279, 281, 790, 793
- actionSet 244
- command 221
- decorator 803, 806
- feature 665
- menu 220, 223, 244, 267, 271, 274, 278, 280

Label Decorations preferences 16, 54–55, 802–803
Label decorators

- declaring 803
- decorative 806
- defined 802
- I LightweightLabelDecorator 804

LabelProvider

- class 86, 207
- extends 203, 308, 370

LabelProviderChangedEvent 805
Label providers

- defined 194
- editor 370
- hierarchy 195
- responsibility 86
- view 307
- WorkbenchLabelProvider 308
- WorkbenchPartLabelProvider 308

LabelRetargetAction 388

Label, *see* GEF/Label
Launch

- external programs 600
- program 600

launch 600–601, 790–791
Launch Configuration dialog 65
Launch configurations 141, 801
launch dialog 710
launcher, editor 357
Launching

- clear workspace data before 252
- configuration 57, 94
- Java applications 56
- Run 56
- Run History 56
- running a Java application 55
- Runtime Workbench 94, 96
- selecting plug-ins 95
- Terminate 61

Launching preferences 61
LaunchURL 789–790
LayerConstants 756
Layout

- dialog units 448
- management 178
- null 448

layout 169, 171
Layouts, null layout 178
lazy initialization

- action filtering and enablement 241
- delayed change events 420
- extension points 119, 645, 652
- IActionDelegate 260

libraries 116
Libraries page 21
license, feature 665
lightweight, decorator 803, 806
LinearUndoEnforcer 400
LinearUndoViolationUserApprover 400
LineNumberReader 364, 366
line.separator 320, 323
Link files 111
List

- class 143, 156, 200
- example 158

ListDialog 445
List-oriented viewers 193
ListSelectionDialog 445
ListViewer 200, 202
List Viewers, example 202
load 493, 505
loadDefault 493
loadDialogSettings 126
loadFavorite 653–654, 658–659
loadFavorites 343–344
Loading Sample Code 105

Locale 240
 Local History 47
 Local history 46
Local History preferences 17, 47
 location, decorator 803, 806
 locationURI 220, 230, 319
 menuContribution 220, 222–223, 229, 763
 locationURL 228
 locatorURI 325, 334
 Log 127
 logError 129, 339, 344, 418, 650, 654, 789, 791, 813
 .log file 131
 Logging 128
 logInfo 129
 Long-running operations 808
 loosely coupled 637

M

macrodef 708
 mailing lists 828
 mailto, syntax 793
 main 139
 make 693
 MalformedURLException 789
 managing native resources 188
 MANIFEST.MF file 73, 77, 79–80, 108, 110, 583, 633, 722
 MANIFEST.MF page 77, 79, 583, 634
 Manifest-Version 583, 634
 Marker 361
 markerHelp
 helpContextId 598
 markerType 598
 markerResolutionGenerator
 class 557
 element 556
 markerType 557
 MarkerResolutionSelectionDialog 445
Markers
 attributes 553
 creating and deleting 551
 data structures 549
 defined 548
 finding 561
 help 598
 Java compiler usage 548
 quick fix 556
 resolution 556
 resolution example 559
 types 549
 markers
 delete 570
 MarkerTransfer 330
 matchingStrategy, editor 357

Match rules
 compatible 632, 672
 equivalent 632, 672
 greater or equal 632, 672
 perfect 632, 672
 required plug-ins 671
Maximum file size field 47
 MaxPermSize 3
 MB_ADDITIONS 248, 317
Members view 8, 270
 Memory leaks, preventing 346
 Menu
 class 143, 145, 177, 317
 example 177
 menu
 checked 569
 context 228
 id 220, 244, 267, 271, 274, 278, 280
 label 220, 223, 244, 267, 271, 274, 278, 280
 listener 569
 locationURL 228
 mnemonic 221
 org.eclipse.ui.main.menu 220
 path 244, 267, 271, 274, 278, 280
 popup 228
 pulldown 228
 selection listener 569
 visibleWhen 223–224, 227, 229
 menuAboutToShow 382
 menubarPath 249
 menubarPath, action 245, 259, 264, 272, 275, 279, 281, 790, 793
Menu Contribution 220
 menuContribution 220–221, 223, 229–230, 319, 325
 label (default) 219
 locationURI 220, 222–223, 229, 763
 locationURL 228
 org.eclipse.ui.main.menu 242
 visibleWhen 220
 Menu event 143
Menu groups 245
 MenuItem 143, 175–177, 569
 menu item 245
 editor 229
 view 228
 MenuListener 175, 569
 MenuManager 388
Menus
 & 285
 additions 317
 defined 174
 Edit 11
 File 11
 groupMarker 267, 274
 Help 12
 insertion points 248

- Navigate 12
 - Project 12
 - Run 12
 - Search 12
 - separator 274, 317
 - submenu 267, 270, 274
 - top-level menu discussion 255
 - underscore 285
 - view 319
 - Window 12
 - MessageBox 442
 - MessageDialog 397
 - class 394, 444
 - openQuestion 394
 - MessageDialogWithToggle 444
 - MessageFormat
 - class 457
 - format 457, 624
 - Messages 624, 627
 - messages.properties file 624, 627, 635
 - .metadata 342
 - metadata 342
 - .metadata directory 345, 501
 - Metrics 820
 - Microsoft 137
 - mnemonic
 - command 221
 - menu 221
 - Models
 - commands 313
 - display 369
 - editor 363
 - handlers 313
 - parsing 363
 - saving snapshot 345
 - saving state 343
 - singleton 299
 - view 295
 - Modified Type view 26
 - ModifyEvent 161
 - Modify event 143
 - Modifying the Eclipse Base 797
 - ModifyListener 154, 159, 161–162, 476
 - modifyText 161–162
 - More Info button 674
 - MOUSE_CLICK_SELECTION 336
 - MouseAdapter 339
 - mouseDoubleClick 339
 - MouseEvent 336, 339
 - Mouse event 143
 - MouseListener 146
 - MouseMove event 143
 - MouseMoveListener 146
 - MouseTrack event 143
 - MouseTrackListener 146
 - Move command 43
 - MultiPageEditorActionBarContributor 385
 - MultiPageEditorPart 358
 - class 379
 - example 360
 - extends 360
 - methods 359
 - MultiPageEditor, setPageImage 359
 - MultiStatus 130
 - MyEclipse Enterprise Workbench 823
 - Mylyn 65–69, 132
 - Searching Bugzilla 67
 - Task List view 67
 - Mylyn Query wizard 67
- ## N
- name
 - Ant target 694
 - attribute 119, 263
 - category 292, 470, 605, 618
 - cheatsheet 605–606
 - command 219, 763
 - command category 219
 - editor 356, 761
 - extension-point 640
 - itemType 642
 - page 486, 498–499, 516, 530
 - perspective 425–426
 - view 293, 618, 755
 - wizard 470, 472
 - nameFilter 260
 - nameFilter, objectContribution 257
 - nameFilter, page 517
 - Naming projects 73
 - National Language Support, *see* Internationalization
 - native resources 188
 - nativeToJava 332
 - natureId, image 567
 - Natures
 - add 570–571
 - associating projects 568
 - associating builders 564
 - conflicting natures 567
 - declaring 562
 - defined 561
 - existing in Eclipse 562
 - hadNatures 537
 - hasNature 564
 - id 571
 - image 567
 - IProjectNature 565
 - org.eclipse.pde.PluginNature 80
 - projectNature action attribute 263
 - remove 570–571
 - required natures 566
 - tag 24
 - nature tag 24

- Navigate > Go to Line... command 27
 - Navigate > Open Type... command 26
 - Navigate > Open Type Hierarchy... command 27
 - Navigate > Open Type in Hierarchy command 27
 - Navigate menu 12
 - Navigating the Workbench 26
 - Navigator view 12, 21, 33, 349, 403, 694, 802
 - needsSaving 505
 - Nested preference pages 498
 - Network Connections preferences 16
 - New > action command 245, 271, 273, 279, 281
 - New > actionSet command 243, 436
 - New > attribute command 550
 - New > builder command 537, 564
 - New > category command 219, 291
 - New > command command 219
 - New > contexts command 597
 - New > editorContribution command 280
 - New > Extension... command 217, 243
 - New > filter command 519
 - New > groupMarker command 245
 - New > itemType command 658
 - New > key command 238
 - New > menu command 244, 267, 271, 274, 278, 280
 - New > page command 516
 - New > persistent command 550
 - New > perspective command 425
 - New > perspectiveExtension command 431
 - New > perspectiveShortcut command 435
 - New > Project... command 50, 52, 72
 - New > Reference > itemType command 648
 - New > Repository Location... command 50, 797
 - New > run command 537, 565
 - New > runtime command 565
 - New > separator command 245
 - New > super command 550
 - New > view command 292, 432
 - New > viewerContribution command 271, 278
 - New > viewShortcut command 434
 - New > wizard command 471
 - newCommand 546
 - New Extension Point wizard 639, 641
 - New Extension wizard 217, 243, 424, 430, 485, 487, 583, 596, 656, 658
 - newFavorite 653, 658
 - newFavoriteFor 344, 786
 - New Feature wizard 663
 - New File Type dialog 48
 - NewFolderDialog 444
 - New Fragment Project wizard 630
 - New Java Class dialog 237
 - New Java Class wizard 24, 249, 426, 557, 565, 658
 - New Java Package wizard 24
 - New Plug-in Project wizard 72, 77, 84, 581
 - New Project wizard 19, 50, 52, 73, 581, 663, 679
 - New Search view 29, 31
 - Newsgroup 781
 - New Template dialog 42
 - New Update Site wizard 680
 - NewVersionCheckJob 808–810
 - New Working Set dialog 34
 - .nll 630
 - NLS class 627
 - NLS, *see* Internationalization
 - NoClassDefFoundError
 - ûjar (note funny hat over u) 718
 - NoClassDefFoundError 116
 - Non-externalized strings option 628
 - NonLocalUndoUserApprover 400
 - Normal identifier 249
 - not 262
 - notifyListeners 145
 - null layout 178
 - NullProgressMonitor 416
- O**
- ObjectAction 257
 - objectClass
 - decorator 803
 - filters 260
 - objectContribution 257
 - page 517
 - visibility 260
 - objectContribution
 - adaptable 257
 - described 270
 - id 257
 - nameFilter 257
 - objectClass 257
 - ObjectShare xliv
 - objectState 262
 - Object Technology International xxxvii, 135
 - ObjectUndoContext 398
 - OK_STATUS 395–396
 - okPressed 447
 - one-of-nature 567
 - open 139, 151, 154, 269, 447, 473
 - OpenBrowserAction 790
 - openEditor 339, 560
 - openError 397
 - Open Favorites View action 247
 - OpenFavoritesViewActionDelegate 253–254
 - OpenFavoritesViewHandler 237
 - OpenFavoritesViewTest 253
 - Opening help page programmatically 599, 603
 - Opening Perspectives 50
 - Opening Web page programmatically 600
 - Open Perspective menu 435

- openQuestion 394
- Open Schema** command 641
- Open Type** dialog 26–27
- openURL 789
- OpenWebPageHandler 600, 788
- OperationCanceledException 542
- OperationHistoryFactory 398
- OperationStatus 130
- Options for accessing internal code 782
 - or 262
- Order and Export** page 21
- Organize Imports** preferences 37
- Organizing Import Statements 37
- org.Eclipse.actionSetPartAssociations 256
- org.eclipse.core.commands.IHandler 237
- org.eclipse.core.commands.jar 201
- org.eclipse.core.expressions.propertyTesters 225
- org.eclipse.core.resources 268, 298, 407, 565, 710
 - org.eclipse.core.resources.bookmark 549
 - org.eclipse.core.resources.builders 535, 538
 - org.eclipse.core.resources.markers 550
 - org.eclipse.core.resources.natures 562
 - org.eclipse.core.resources.problemmarker 550
 - org.eclipse.core.resources.taskmarker 550
 - org.eclipse.core.resources.textmarker 550
 - org.eclipse.core.runtime 80, 101, 121–122, 129, 131, 415, 501, 794
 - org.eclipse.core.runtime.jar 201
 - org.eclipse.core.runtime.jobs 126
 - org.eclipse.core.runtime.preferences 506
 - org.eclipse.core.runtime.products 677
- org.eclipse.debug.core 131
 - statusHandlers extension point 131
- org.eclipse.equinox.common.jar 201
- org.eclipse.help.contexts 596–597
- org.eclipse.help.toc 586
- org.eclipse.help.ui 789
- org.eclipse.help.ui.browser 789
- org.eclipse.jdt.core 298
 - org.eclipse.jdt.core.IJavaElement 225
 - org.eclipse.jdt.core.javanature 562
 - org.eclipse.jdt.ui 304
- org.eclipse.jface.action 85
- org.eclipse.jface.dialogs 85, 442, 446
- org.eclipse.jface.fieldassist 468
- org.eclipse.jface.jar 201
- org.eclipse.jface.preference 489–490
- org.eclipse.jface.resource 83
- org.eclipse.jface.text 213, 371
- org.eclipse.jface.text.jar 201
- org.eclipse.jface.viewers 85, 101, 202, 206–207, 210–211
- org.eclipse.jface.wizard 464–465, 469
- org.eclipse.pde.build 710
- org.eclipse.pde.FeatureNature 562
- org.eclipse.pde.PluginNature 80, 562
- org.eclipse.pde.UpdateSiteNature 562
- org.eclipse.platform 677
- org.eclipse.swt 85, 154, 156, 158, 161, 165, 168, 170, 172, 174, 177, 179, 181, 184, 187, 202, 206, 210, 213, 442
 - org.eclipse.swt.custom 213
 - org.eclipse.swt.events 154, 156, 158, 161, 165, 168, 170, 172, 174, 177, 179, 181
 - org.eclipse.swt.graphics 85, 207, 213
 - org.eclipse.swt.layout 154, 156, 158, 161, 165, 168, 174, 179, 181, 184, 187, 202, 206, 210, 213
 - org.eclipse.swt.program 790
 - org.eclipse.swt.widgets 85, 101, 154, 156, 158, 161, 165, 168, 170, 172, 174, 177, 179, 181, 184, 187, 202, 206, 210, 213
- org.eclipse.text.jar 201
- org.eclipse.ui 80, 85, 101, 119, 353, 385, 427, 793
 - org.eclipse.ui.actions 387, 414
 - org.eclipse.ui.actionSets 242–243
 - org.eclipse.ui.cheatsheets 605
 - org.eclipse.ui.cheatsheets.cheatSheetContent 605
 - org.eclipse.ui.command 218
 - org.eclipse.ui.decorators 802–803
 - org.eclipse.ui.editorActions 280
 - org.eclipse.ui.editors 356
 - org.eclipse.ui.exportWizards 469–470
 - org.eclipse.ui.file.properties 522
 - org.eclipse.ui.handlers 236
 - org.eclipse.ui.ide 298
 - org.eclipse.ui.ide.markerHelp 598
 - org.eclipse.ui.ide.markerResolution 556
 - org.eclipse.ui.ide.projectNatureImages 567
 - org.eclipse.ui.ide.workbench 677
 - org.eclipse.ui.importWizards 470
 - org.eclipse.ui.main.menu 220
 - menuContribution 242
 - org.eclipse.ui.main.toolbar 222
 - org.eclipse.ui.menus 220
 - org.eclipse.ui.newWizards 470–471, 485
 - org.eclipse.ui.part 85, 353
 - org.eclipse.ui.perspectiveExtensions 430, 434, 437
 - org.eclipse.ui.perspectives 423, 425–426
 - org.eclipse.ui.plugin 83
 - org.eclipse.ui.popup.any 223
 - org.eclipse.ui.popupMenu 259
 - org.eclipse.ui.popupMenus 257

- org.eclipse.ui.preferencePages 485
- org.eclipse.ui.propertyPages 516
- org.eclipse.ui.resourcePerspective 435
- org.eclipse.ui.startup 121, 421, 816
- org.eclipse.ui.viewActions 273
- org.eclipse.ui.views 81
- org.eclipse.ui.views.extension 291
- org.eclipse.ui.views.properties 524
- org.eclipse.ui.workbench 798
- org.junit 101
- org.junit.Assert 346
- org.osgi.framework 83
- OSGi 107–108
 - Alliance 113
 - Bundle-ActivationPolicy 77, 84
 - Bundle-Activator 114, 120
 - Bundle-ClassPath 116
 - BundleContext 83
 - Bundle-Name 114
 - Bundles 127
 - Bundle-SymbolicName 114
 - Bundle-Vendor 114
 - Bundle-Version 114
 - defined 113
 - Eclipse-LazyStart 77
 - Equinox project 128
 - Export-Package 116
 - headers 127
 - id 127
 - lazy plug-in loading 109
 - manifest file, *see* MANIFEST.MF
 - osgi.splashPath 678
 - plug-in class 120
 - plug-in dependencies 116
 - plug-in runtime 116
 - registry changes 128
 - Require-Bundle 116
 - state 127
 - symbolic name 128
 - web site 113
- osgi.splashPath 678
- OTI xxxvii–xxxviii, 135–136, 191
- Outline** view 27–28, 349, 353, 400, 405, 427–428
- overrideActionId 259
- overrideActionId, action 259, 272
- Overview** page 79, 83–84
- P**
- p2 661, 683
- pack 147, 163, 165–166
- Package Explorer** view 6–7, 21, 33, 66, 562
- Packages** view 8
- page
 - adaptable 517
 - category 487, 498–499, 517, 530
 - class 487, 498–499, 517, 530
 - icon 517
 - id 486, 498–499, 516, 530
 - name 486, 498–499, 516, 530
 - nameFilter 517
 - objectClass 517
- pageChange 379, 399
- pageSelectionChanged 787
- Paint event 143
- PaintListener 146
- palette 761
- Parameterized Types 794
- Parameters** page 58
- ParcPlace 136
- parentCategory, category 470
- parseType 650
- Part Identifiers 797
- PartInitException 251, 339, 342, 360, 560, 789
- PartSite 799–800
- Password 797
- Paste 326
- paste 155, 160
- Path 480
- path 263
- PathEditor 491
- path, menu 244, 267, 271, 274, 278, 280
- Paths, example 478
- Patterns
 - singleton 299
 - templates 40
 - third-party plug-ins 42
 - type-safe enumeration 296
- PDE 132, 711
 - adding jars to classpath 722
 - baseLocation property 714
 - base property 716
 - buildDirectory property 714
 - builder property 714
 - build.xml 715
 - feature build templates 715
 - headless-build 715
 - headless build templates 715
 - java compilation level 717, 720, 722
 - packager build templates 715
 - plugin build templates 715
 - Scripts 714
 - sign and pack jar files 697, 714
 - Templates 714
 - topLevelElementId property 716
- PDE Build process 711, 714–716, 727

- pde.exportPlugins 89
- PDE Spy 99
- PDE Tools > Create Ant Build File** command 727
- PDE Tools > Externalize Strings...** command 619
- PDE views 96
- performCancel 466
- performDefaults 490, 494–495
- performFinish 466, 474
- performOk 490, 494–495, 520
- performOperation 474
- PermSize 3
- persistentProperty 263
- Person
 - class 200, 202
 - example 201
- PersonListLabelProvider 202–203, 210
- PersonTableLabelProvider 206–207
- PersonTreeContentProvider 210–211
- perspective
 - class 426
 - icon 426
 - id 425–426
 - name 425–426
- perspectiveExtension 431
- Perspectives
 - adding action sets 436
 - adding placeholders 432
 - adding shortcuts 434
 - adding views 432
 - createFolder 428
 - createInitialLayout 427–428
 - creating 423
 - CVS Repository Exploring** 50
 - Debug** 10, 59
 - declaring 424
 - enhancing existing 430
 - factories 426
 - getEditorArea 428
 - IFolderLayout 428
 - IPageLayout 428–429
 - IPerspectiveFactory 423, 426–427
 - Java** 7, 13, 19
 - Java Browsing** 7
 - layout 423
 - New 425
 - opening 50
 - org.eclipse.ui.perspectiveExtensions 430
 - overview 5
 - page layout 429
 - Resource** 9, 13, 432, 434–436
 - RFRS 438
- perspectiveShortcut 435–436
- Perspectives preferences 16
- PlanetEclipse 829
- Platform
 - class 126, 306, 337, 650
 - class 787
 - getConfiguratiOnLocation 123
 - getExtensionRegistry 128
 - getJobManager 103
 - resolve 122
 - resources 188
- platform 240
- PlatformUI 315, 599, 799, 807, 817
 - class 86, 101, 253, 269, 297, 398, 599
 - getActiveWorkbenchWindow 464
 - getShell 464
 - getWorkbench 464
- Plugin 121
 - AbstractUIPlugin 125
 - class 83, 120, 505
 - detecting disabled early startup 817
 - disabling early startup 817
 - download-size 666
 - early startup 121
 - example 120
 - feature 666
 - find 121
 - fragment 666
 - getStateLocation 127
 - id 666
 - install-size 666
 - openStream 121
 - preferences 122
 - resources 121
 - shutdown 120
 - startup 120
 - unpack 666
 - version 666
- plugin_customization.ini file 674, 677–678
- plugin_customization.properties file 674, 678
- PLUGIN_ID 514
- Plug-in Dependencies** view 97
- Plug-in Details** button 115
- Plug-in Details** dialog 674
- Plug-in Development Environment, *see* PDE
- Plug-in development projects 19
- .plugin directory 345
- Plug-in Manifest editor 77
- Plug-in Manifest editor, **Extensions** page 485
- plug-in manifest, in relation to runtime classpath 23
- Plug-in Project Content** page 582
- Plug-in Project wizard** 72
- plugin.properties file 114, 473, 533, 541, 618–619, 635
- Plug-in Registry** view 96

- Plug-ins 342
 - also known as* Bundles
 - access 254
 - Activator 120
 - cached plug-in information 92
 - class 120
 - class loader 811
 - class visibility 100
 - configuration directory 92
 - configuration files 123
 - converting URLs 122
 - Core 132
 - currently installed 126–127
 - dialog 115
 - Directory 110
 - directory name 111
 - extension registry 128
 - External 95
 - favorites plug-in 71
 - finding 829
 - fragment 629
 - from id 126–127
 - Help 132
 - Import Source Project 797
 - JAR file 88
 - JDT core 132
 - JDT UI 132
 - JFace 132
 - lazy initialization 119
 - Loader 107
 - loading 107, 420
 - Manifest 113
 - manifest editor page 670
 - org.eclipse.jdt.ui 304
 - PDE View 780
 - precedence 95
 - preferences 122, 501, 506–507
 - resources 127
 - runtime 116
 - search 98
 - searching 32
 - signing and security 115
 - start 83, 120
 - state 127
 - State information, *see* Workspace metadata
 - stop 83, 120, 125, 513
 - SWT 132
 - team 132
 - testing 100
 - tests 99
 - version 114
 - with the same identifier 114
 - Workbench core 132
 - Workbench UI 132
 - workspace 95
- .plugins 342
- plugins directory 92, 680, 683
- Plug-ins** page 670
- Plug-in Spy 99
- pluginState 262
- Plug-ins** view 97, 780, 797
- PluginTransfer 330
- Plug-in** view 97
- Plug-in with a view option** 72
- plugin.xml
 - actions in reverse order 259
 - class declaration 116
 - class visibility 100
 - declaration 114
 - dependencies 116
 - extensions and extension points 118
 - file 73, 77–78, 81, 108, 110, 533, 541, 575, 585, 633, 727
 - identifier 114
 - internationalization 618
 - name and provider 114
 - version 114
- plugin.xml** page 78, 426, 434, 436–437, 585, 597, 640, 790
- Pollinate xliv
- PolylineConnection 751
- popup 228
 - org.eclipse.ui.popup.any 223
- Popup menu 12, 278, 382
- PopupMenuExtender 798
- PreferenceConstants 488, 495
- preferenceCustomization 678
- PreferenceInitializer 488, 505
- PreferencePage
 - class 487, 492, 494, 500, 529
 - validating 494
- Preference Pages
 - dabel decorations 802
 - errors/warnings 628
 - target platform 798
- Preferences
 - *.prefs 501
 - accessing 503
 - adding property change listeners 508
 - API 501
 - ConfigurationScope 503
 - createFieldEditors 496
 - defaults 502
 - defaults in code 505
 - defaults in file 506
 - default values
 - Boolean 502
 - double 502
 - float 502

- int 502
- long 502
- String 502
- editing manually 501, 506
- example page 495
- FieldEditorPreferencePage 487, 500
- field editors 490
- getPluginPreferences 123
- getPreferenceStore 123
- getPreferenceStore 122, 503, 505
- getStateLocation 122
- import and export 18
- initializeDefaultPluginPreferences 123, 505
- InstanceScope 503
- IPreferenceStore 503
- IWorkbenchPreferencePage 487
- Java 17
- listening for changes 507
- nested pages 498
- page 487
- page declaration 485
- pages for a product 499
- PreferencePage 487, 494, 500, 529
- Preferences 503
- preferences.ini 506
- preferences.properties 122
- ProjectScope 503
- removing property change listeners 508
- reusing property pages as preference pages 529
- RFRS 508
- ScopedPreferenceStore 503
- sharing between multiple workspaces 123
- sharing resources among multiple workspaces 123
- storage 501
- tabbed pages 500
- validating 494
- validation 497
- values
 - current 502
 - default 502
 - default-default 502
- view integration 507
- Workbench 15
- Workbench Preferences dialog** 14
- XML storage 502
- Preferences dialog** 14–15, 18, 487, 506, 508, 705
- preferences.ini file 122, 506–507, 722
- preferences.properties file 122
- prefs file 501
- prepareToSave 420
- presentsDefaultValue 493
- PrintWriter 380, 457
- Problems view** 428, 533, 550, 553, 556, 798
- processing resource change events 411
- product
 - aboutImage 677
 - aboutText 677
 - application 677
 - appName 677
 - description 677
 - name 677
 - preferenceCustomization 677
 - windowImages 677
- Product branding** 677
- productizing 661
- Profiling 821
- Program 600–601, 788, 790–791
- PROGRAMMATIC 336
- Progress
 - displaying 416
 - monitor 415
- Progress Monitor
 - example 414–415, 541
 - example subprogress 415
 - example subtask 418
 - obtaining 419
 - setCanceled 359
 - status bar 419
 - used for long-running operations 415
 - wizard 466
- ProgressMonitorDialog 417, 419, 445
- ProgressMonitorPart 416
- ProgressMonitorWrapper 416
- Progress view** 811
- project
 - description 571
- Project > Rebuild Project** command 572
- ProjectClassLoader 813–814
- projectDescription 23
- Project Explorer view** 9
- .project file 22–23, 546, 562, 575
- ProjectLocationMoveDialog 444
- ProjectLocationSelectionDialog 444
- Project menu** 12
- Project naming 73
- projectNature 263
- Project natures 533
- projectPersistentProperty 264
- Projects
 - Checking out from CVS 51
 - creation 19
 - description 546
 - feature 662
 - Java Build Path 23
 - natures 533
 - types of 19
- Projects > Clean...** command 574
- ProjectScope 503

- projectSessionProperty 264
 - Projects** page 21
 - PROP_DIRTY 359
 - Properties**
 - Ant 699
 - creating 511
 - displaying in the **Properties** dialog 515
 - displaying in the **Properties** view 524
 - enabledWhen 517
 - example 512, 527
 - filters 519
 - getPersistentProperty 514
 - getSessionProperty 514
 - in resources 513
 - IPropertyChangeListener 508
 - IResource properties API 513
 - IWorkbenchPropertyPage 517
 - opening properties dialogs 522
 - persistent 513
 - persistentProperty action attribute 263
 - projectPersistentProperty action attribute 264
 - projectSessionProperty action attribute 264
 - Properties** view API 525
 - PropertyPage 529
 - property page creation 519
 - property page declaration 515
 - resource 513
 - reusing property pages as preference pages 529
 - RFRS 530
 - session 513
 - sessionProperty action attribute 264
 - setPersistentProperty 514
 - setSessionProperty 514
 - Show Advanced** command 527
 - PropertiesAuditorNature 565
 - Properties** dialog 23, 138, 511, 515, 522, 573, 669
 - PropertiesEditor 360
 - PropertiesEditorContentProvider 369
 - PropertiesEditorContributor 386, 388
 - PropertiesEditorLabelProvider 370
 - PropertiesFileAuditor 540, 548, 551
 - PropertiesOutlinePage 400–401
 - Properties** view
 - API 525
 - cell validation 527
 - displaying properties 511, 524
 - save immediately 349
 - simple modification tasks 401
 - ungrouped properties 527
 - PropertyCategory 364, 374, 396
 - propertyChange 497, 508
 - PropertyChangeEvent 508, 810
 - PropertyDescriptor, methods 526
 - PropertyElement
 - class 363, 394
 - extends 364, 366
 - PropertyEntry 363, 374
 - PropertyFile 366, 396
 - PropertyFileListener 368, 374
 - propertyNames 505
 - PropertyPage 519–520, 529
 - PropertySheet 294
 - propertyTester 225–226, 228
 - forcePluginActivation 228
 - not-loaded 228
 - provider-name, feature 665
 - pullDown 228
 - Pull Up** command 44
 - Push Down** command 44
 - putString 345
- Q**
- QNX Momentics 830
 - QualifiedName 514
 - QualityEclipse
 - tools 781
 - Web site 482
 - Quick Fix 40
 - Quick Fix** command 40
- R**
- RadioGroupFieldEditor 491
 - ratio 433
 - Rational Application Developer 113, 725
 - Rational Software family 831
 - RCP 107–108
 - defined 817
 - Eclipse Rich Client Platform* book 817
 - generic application framework 817
 - SWT usage 137
 - RCP Developer xlvii
 - readAndDispatch 103, 139–140, 150, 154, 202, 206
 - readFile 544
 - readOnly 263
 - readUTF 333
 - Ready for Rational Software, *see* RFRS
 - REDO 399
 - redo 394
 - RedoActionHandler 398
 - RedoActionHandler 398
 - redraw 147
 - Reexport this dependency** checkbox 118
 - Refactor > Extract Constant...** command 620
 - Refactoring
 - Preview** dialog 45
 - rename 333
 - usage 42
 - Refactor** menu 11–12, 42

- Reference Projects 30, 780
- Referencing a class in a different plug-in 796
- refresh 199, 212, 734
- refreshValidState 493
- registerAdapters 787
- Register Cheat Sheet 604
- registerContextMenu 317, 382
- relationship, view 432, 434
- relative, view 432, 434
- Remove 320
- remove 157, 160, 163, 200, 204, 208
- removeAll 157, 160, 163, 167
- RemoveAllWhenShown 316
- removeBuilderFromProject 547, 570
- RemoveFavoritesContributionItem 314–316, 319–320
- RemoveFavoritesHandler 315–316, 325
- removeFavoritesListener 512, 524
- removeFavoritesManagerListener 805
- removeNature 570–571
- removePageChangedListener 469
- removePostSelectionListener 338
- removePropertyChangeListener 508, 810
- removePropertyFileListener 375
- removeResourceChangeListener 408
- Remove terminated launches when a new launch is created** checkbox 61
- Rename
 - handler 333
 - refactoring 333
- Rename** command 43
- RenameFavoriteHandler 333
- RenameFavoritesHandler 334, 336
- Replace** dialog 29, 54
- Replace With > History** command 54
- Replace With > Local History...** command 46
- reportProblem 544, 552, 556
- Repository Location 797
- Repository Path 797
- Repository path** field 50
- Request 734
- Require-Bundle 80, 116–118
- requires-nature 566
- Rerun Last Test** button 65
- resetFilters 199
- resetPropertyValue 526, 528
- Resource Attribute Value** dialog 247
- resourceChanged 411–412
- ResourceEditPart 750
- ResourceExtender 263
- ResourceListSelectionDialog** 445
- Resource Management 188
- ResourceManager 347
- Resource** perspective 9, 13, 432–436, 438–439
- Resources
 - addResourceChangeListener 408
 - batching change events 414
 - change tracking 407
 - delayed changed events 420
 - derived 533, 535, 545
 - IResourceChangeEvent 408
 - IResourceChangeListener 407
 - processing change events 411
 - properties 513
 - remove listener on shutdown 407
 - removeResourceChangeListener 408
 - setDerived 545
- ResourceSelectionDialog 445
- ResourcesPlugin
 - addResourceChangeListener 408
 - addSaveParticipant 420
 - class 269, 303, 415, 478, 480, 553, 814
 - getRoot 478
 - getWorkspace 408, 478
 - removeResourceChangeListener 408
- ResourceStatus 130
- ResourceTransfer 324, 326, 328–331
- RetargetAction 387
- retarget, action 246
- reveal 200, 204, 208
- reverting changes 46
- RFRS
 - Actions, global action labels 286
 - Builders 572
 - do not misuse the term “build” 573
 - do not replace existing builders 573
 - mark created resources as “derived” 573
 - must be added by natures 575
 - respond to clean build requests 574
 - use IResourceProxy objects when possible 574
 - use to convert resources 572
 - certification program xl, 831
 - Cheat sheets
 - illustrate only one task 614
 - provide a help link with each step 614
 - provide an overview 613
 - Eclipse attribution 675
 - Editors 401
 - accessing global actions 402
 - action filters 404
 - closing when the object is deleted 403
 - lifecycle 401
 - Outline** view 405
 - prefix dirty resources 404
 - registering menus 403
 - synchronize with external changes 403
 - synchronize with the Outline view 405
 - unsaved modifications 404
 - used to edit or browse 401

RFRS (*continued*)

- Extension Points 659
 - document 659
 - log errors 660
 - Features 689
 - about .html file contents 690
 - branded feature visibility 689
 - do not override product branding 689
 - include attribution information 689
 - splash screen restrictions 690
 - Help 611
 - context help activated using F1 612
 - implement active help 612
 - provide all help via Workbench 612
 - provide through the Workbench 611
 - use of additional documentation 613
 - use of stand-alone help 613
 - Link files 111
 - logging 128
 - Perspectives 438
 - adding actions to window menu 439
 - create for long-lived tasks 438
 - extend existing 438
 - plugin.properties 473
 - Preferences dialog use 508
 - Properties, views use for quick access 530
 - Views
 - action filters 351
 - for navigation 348
 - global actions from menubar 349
 - initialization 349
 - persist state 350
 - register context menus 350
 - save immediately 349
 - Web site 832
 - Wizards 482
 - new project switches perspective 483
 - one-page wizards 484
 - open new file in editor 483
 - show new object 483
 - wizard look and feel 482
- RGB 512
- RGB 513, 522
- Rich Client Platform 107–108
- Rich Client Platform, *see* RCP
- rollback 420
- rootEditPart 755
- RowData
- class 181
 - height 181
 - width 181

- RowLayout
- class 180
 - example 181
 - justify 180
 - marginBottom 180–181
 - marginLeft 180–181
 - marginRight 180–181
 - marginTop 180–181
 - pack 180
 - spacing 180–181
 - type 180
 - wrap 180
- RTFTransfer 330
- Run 56
- run 127, 252, 414, 418, 463, 468, 553
 - Run > Debug As > Java Application** command 59
 - Run > Debug History** command 59
 - Run > Run As > JUnit Test** command 64
 - Run > Run...** command 57, 65
 - Run > Run History** menu 56
 - Run Ant** command 91
 - Run As > Ant Build...** command 694, 704
 - Run As > Java Application** command 56–57, 141
 - Run As > JUnit Plug-in test** command 103
 - Run As > Run...** command 104
 - Run dialog** 141
 - Run history 56
 - Run in the same JRE as the workspace option** 91, 710
 - Run menu** 12
 - Running a plug-in test 103
 - Runtime 791
 - runtime classpath 23
 - Runtime** page 80, 100, 633
 - Runtime preferences** 705
 - Runtime Workbench
 - launch configuration 268
 - launching 96
 - using 94, 96, 412, 801
 - vs. Development Workbench 96

S

- SameShellProvider 456, 459
- Sample Code Online 105
- Sash 143
- SaveAsDialog 444
- saveFavorites 343–344
- saveState 294, 340–342
- saveState, view 342
- Saving local view information 340
- ScalableFreeformRootEditPart 755
- Scale 143
- ScaleFieldEditor 491
- scanPlugin 543
- scanProperties 543

- schema editor
 - accessing 641
 - Attribute Details** 643
 - Documentation** 641
 - Element Details** 645
 - Element Reference Details** 648
 - Extension Element Details** 641
 - Extension Point Elements** 641, 647–648
 - General Information** 641
 - New Attribute** button 645
 - New Element** button 645
- schema, extension point 640–641
- schema subdirectory 639
- ScopedPreferenceStore 503
- scripts for PDE 714
- Scrollable 148
- ScrollBar 143, 145
- scrollDown 208
- ScrollingGraphicalViewer 755
- scrollUp 208
- SDP xxxix
- search 594
- Search > Declarations** command 33
- Search > Implementors** command 33
- Search > References** command 33
- Search > Search...** command 28
- Search > Write Access** command 33
- Search dialog 28
- Searching 28
 - declarations 30
 - file search 28
 - implementors 30
 - Java search 30
 - match locations 30
 - options 32
 - plug-ins 32
 - references 30
 - regular expressions 28
 - scope 29–30
 - working set 30, 33
- Search menu 12, 28
- Search preferences 16
- Search, reference projects 780
- Search string field 30–31
- Search view 16, 31
- select 157, 160, 163, 198
- selectAll 155, 157, 163, 167
- selectAndReveal 561
- Selected resources scope 29–30
- selection 324–325, 334
 - class 264
 - name 264
- SelectionAdapter 154, 158–159, 161–162, 165–166, 168–170, 172, 177, 179, 310, 314, 318, 476
- selectionChanged 203, 251–252, 269, 314, 322
- SelectionChangedEvent 202, 314, 322
- SelectionDialog 443
- SelectionEvent 154, 158, 161, 168, 177, 179, 310, 314, 318, 476
- Selection event 143, 152
- SelectionListener 142, 152, 154, 156, 159, 162–163, 166, 172, 176
- selection listener 337
- selection provider 337
- Selections
 - provider 306, 317
 - publish 317
 - structured 306
 - view 312
- SelectionStatusDialog 444
- selectReveal 269
- Select Search Scope** dialog 579
- SelectStringsWizardPage 481
- Select Working Sets** dialog 33–34
- Separator 317, 325, 382, 388
- separator 274
- sessionProperty 264
- setAccelerator 176–177
- setActive 151
- setActiveEditor 385–386
- setActivePage 385–386
- setActivePart 269, 463
- setAlignment 152, 163
- setAllChecked 205
- setAllGrayed 205
- setAlwaysIncompatible 526–528
- setAttribute 554
- setAttributes 554–555
- setAutoExpandLevel 208, 371
- setBackground 147, 164, 167, 189, 755
- setBounds 139–140, 147, 154, 178, 206
- setBuilderName 546
- setBuildSpec 546–547
- setCanceled 359, 415
- setCategory 527–528
- setCellEditors 204
- setCellModifier 204
- setChecked 164, 167, 205, 209
- setCheckedElements 205, 209
- setColor 512, 528
- setColorPropertyValue 529
- setColorValue 521
- setColumnData 348, 362
- setColumnProperties 204
- setCommentPropertyValue 520, 529
- setComparer 199
- setConnectionRouter 756

- setContentProvider 86, 196, 199, 202–203, 206–208, 210, 295, 307, 371, 481
- setContents 755–756
- setControl 173–174
- setCursorLocation 150
- setData 145, 150, 199
- setDefault 504
- setDefaultColor 512
- setDefaultPageImageDescriptor 466
- setDerived 545
- setDescription 467, 481, 527, 546–547
- setDocument 212–213
- setEchoChar 155
- setEditable 155, 212
- setEditingSupport 335, 373, 376–377
- setEditorAreaVisible 430
- setEditPartFactory 755
- setEnabled 147, 151, 175–176, 251, 319
- setErrorMessage 377, 467, 478, 494, 497
- setExpanded 167
- setExpandedElements 208
- setExpandedState 208
- setFilter 197
- setFilterFlags 527–528
- setFixed 430
- setFocus 86, 147, 294, 359, 493, 756
- setFont 147, 189, 460
- setForeground 139–140, 147, 164, 168, 189
- setGlobalActionHandler 388, 399
- setGrayChecked 209
- setGrayed 164, 168, 205, 209
- setGrayedElements 205, 209
- setHeaderVisible 163, 165–166, 206, 295, 481
- setHelp 592–594
- setHelpAvailable 466
- setHelpContextIDs 592
- setHelpContextIds 527
- setImage 152–153, 164, 168, 173, 176, 319
- setImageDescriptor 467
- setImageIndent 164
- setInitializationData 540, 794–795
- setInput 86, 196, 199, 202–203, 206, 210, 212, 295, 307
- setItems 157–161
- setLabelProvider 86, 195, 200, 202–204, 206–207, 209–210, 295, 335, 371, 481, 527, 807
- setLabelText 493
- setLayout 139–140, 154, 169, 171, 182, 206, 348
- setLayoutData 147, 183, 451
- setLinesVisible 163, 165–166, 206, 295
- setLocation 147
- setMenu 176–177, 317
- setMenuBar 177
- setMessage 467, 478–479, 494
- setNatureIds 572
- setNeedsProgressMonitor 466
- setPageComplete 467–468, 478
- setPageImage 359
- setPageText 359
- setParentsGrayed 209
- setPartName 361
- setPattern 320
- setPersistentProperty 514
- setPreferenceName 493
- setPreferenceStore 493
- setPresentsDefaultValue 493
- setProject 565
- setPropertyChangeListener 493
- setPropertyValue 526, 528
- setRedraw 147, 212, 307, 383, 395–396, 413
- setRemoveAllWhenShown 316, 382
- setResizable 164
- setReturnCode 448
- setRootEditPart 755
- setSelectedNode 469
- setSelectedRange 212
- setSelection 153, 155, 157, 163, 167, 173, 176, 200, 212, 338, 787
- setSelectionProvider 251, 337
- setSessionProperty 514
- setShellStyle 447
- setSize 147, 750
- setSorter 197, 200, 202
- setSubtreeChecked 209
- setTabList 169, 171
- setTaskName 416
- setText 139–140, 152–155, 160–161, 164, 166, 168–169, 171, 173, 176–177, 206, 362
- setTextColor 212
- setTextEditorUndoRedo 399
- setTextHover 212
- setTextLimit 155, 160
- Setting breakpoints 59
- Setting up your environment 14
- setTitle 467, 481
- setTitleBarColor 466
- setTitleToolTip 361
- setToDefault 504
- setToolTip 750
- setToolTipText 147, 173
- setTopIndex 155, 157, 163
- setTopItem 167
- setTransfer 329
- setTreeUndoRedo 399
- setUp 253, 269
- setUseHashlookup 200
- setValid 490, 494, 497
- setValidator 376, 527

- setValue 376, 504
- setVisible 147, 151, 175, 467, 481, 490
- setWidth 164, 206
- setWindowTitle 466, 474
- shared images 86
- SheetEntry, FILTER_ID_EXPERT 527
- Shell
 - class 139–140, 143, 150, 394, 456, 459
 - finding a parent 150
 - parent 150
 - setText 460
- Shell event 143
- ShellListener 150
- Shortcuts page 14
- shortenText 448
- ShortestPathConnectionRouter 756
- shouldAudit 541
- Show Advanced Properties command 527
- Show Advanced Properties option 528
- Show as List command 31
- showErrorMessage 494
- Show extension point description command 657
- showInDialog() 419, 811
- showMessage 494
- Show Next Match command 31
- Show only extension points from required 218, 220, 225, 236, 243, 596
- showPage 468
- ShowPartInfoHandler 799
- Show Previous Match command 31
- Show Source of Selected Element Only command 35
- showView 101, 269
- Show View command 6
- Show View dialog 93, 96, 105, 131, 290, 292, 780
- Show View menu 93, 119, 435
- shutdown 120, 407
- Simple projects 19
- Singleton 299
- site
 - category-def 684
 - description 684
 - feature 684
- site.css file 680
- Site Manifest Editor
 - Add Feature command 681
 - Archive Mapping section 683
 - Archives page 681, 683
 - Build All button 683
 - Description field 681, 683
 - Feature Environments 682
 - Feature Properties 682
 - Label field 681
 - Managing the Site list 681
 - Name field 681
 - New Category button 681
 - Site Map page 681
 - site.xml page 681, 683
 - This feature is a patch for another feature 682
 - URL field 682–683
- site.xls file 680
- site.xml file 562, 679–681, 683
- sleep 103, 139, 150, 154, 202, 206, 418, 474
- Slider 143
- Smalltalk
 - language xxxvii, 8, 135–136, 185
 - layout manager origins 178
- Software Updates and Add-ons dialog
 - Available Software 686
 - Installed Software 685
 - Properties task 686
 - Uninstall task 686
- sort 197
- Sorters 197
- Sorters, hierarchy 197
- Sorting
 - table 309
 - view 308
- Source > Add Import command 37
- Source > Clean Up... command 45
- Source > Externalize Strings... command 621
- Source > Format command 36
- Source > Organize Imports command 37, 139
- Source menu 11
- Source page 20
- Spellchecker 820
- splash.bmp file 674, 679
- splash screen
 - restrictions 690
 - splash.bmp file 679
- src directory 20
- startup 120
- Startup and Shutdown preferences 16, 817
- state
 - action 246, 259, 272, 282
 - decorator 803, 806
- stateMask 321, 336
- Status
 - CANCEL_STATUS 394
 - class 130
 - OK_STATUS 395–396
- StatusDialog 444
- StatusInfo 444
- Status objects 130
- stop 125, 343, 655
- store 494, 505
- StringBuffer 323
- StringFieldEditor 491
- StringMatcher 311–312

- StringReader 366
- Strings
 - formatting 624
 - matching 311
- Structured parameters 795
- StructuredSelection 338, 787
 - class 269
 - EMPTY 254
- StructuredViewer 312
 - class 198, 311
 - subclasses 198
- Struts support 824
- style, action 247, 260, 282, 790
- StyledText 143–144, 211
- StyleRange 214
- subfeatures 670
- submenu 267, 270, 274
- Submitting Changes to Eclipse 801
- SubProgressMonitor
 - class 415–416
 - example 414
- subTask 809, 811
- Swing
 - early use in Eclipse 137
 - Java Foundation Classes 136
 - Swing Designer 825, 827
- Swing Designer 825, 827
- SWT 336
 - ABORT 442
 - ALT 336, 388
 - APPLICATION_MODAL 443, 447, 460
 - ARROW 153
 - BAR 175, 177
 - BEGINNING 183
 - BOLD 189, 213
 - BORDER 169, 171, 447, 451, 459, 468, 477, 519
 - BOTTOM 186
 - CANCEL 442
 - CASCADE 176–177
 - CENTER 152–153, 155, 183, 186, 206
 - CHECK 153, 163, 167, 176, 569
 - classpath 138
 - CLOSE 447
 - DEFAULT 186, 462
 - DEL 321, 388
 - DIALOG_TRIM 447, 460
 - DOWN 153
 - DROP_DOWN 160, 175
 - early risk 137
 - Eclipse user interface foundation 137
 - END 183
 - event loop example 139
 - FILL 183
 - FULL_SELECTION 163, 206, 295, 360, 362
 - getPlatform 791
 - H_SCROLL 86, 169, 295, 459
 - HelloWorld example 138
 - history and goals 135
 - HORIZONTAL 152, 179–180
 - ICON_ERROR 443, 457
 - ICON_INFORMATION 443, 457
 - ICON_QUESTION 443
 - ICON_WARNING 443, 457
 - ICON_WORKING 443
 - IGNORE 442
 - interoperability with Swing 137
 - ITALIC 189
 - LEFT 152–153, 155, 186, 206, 295
 - MAX 447
 - MIN 447
 - MODELESS 443, 447
 - MULTI 86, 155, 158, 163, 167, 213, 295, 360, 362, 459, 519
 - native platform 138
 - NO 442
 - NO_RADIO_GROUP 169, 171
 - NONE 451
 - NORMAL 189
 - OK 442
 - OPEN 442, 480
 - origins 136
 - parent widget requirement 142
 - POP_UP 175
 - PRIMARY_MODAL 443, 447
 - PUSH 153, 176
 - RADIO 153, 176
 - RCP usage 137
 - READ_ONLY 155, 160, 459
 - RESIZE 447, 460
 - Resource Management 188
 - RETRY 442
 - RIGHT 152–153, 155, 186
 - SAVE 442, 480
 - SEPARATOR 152, 176–177
 - SHADOW_IN 152
 - SHADOW_NONE 152
 - SHADOW_OUT 152
 - SHELL_TRIM 447
 - SIMPLE 160
 - SINGLE 155, 158, 163, 167, 202, 206
 - stand alone example 138
 - style settings 142
 - SWT Designer 825, 827
 - SYSTEM_MODAL 443, 447
 - TITLE 447
 - TOGGLE 153
 - TOP 186
 - UI thread 148
 - UP 153
 - use in RCP 137

- v_SCROLL 86, 169, 295, 459
 - versus Swing 137
 - VERTICAL 152, 179–180
 - VSCROLL 213
 - widgets 138
 - widget state 141
 - WRAP 152, 155, 519
 - YES 442
 - SWTError 324
 - SWTException 148, 413
 - syncExec 148
 - Synchronize** view 52–53
 - Syntax Coloring** preferences 36
 - System, currentTimeMillis 103
 - System Properties
 - line.separator 323
 - system properties 700
 - System Property
 - line.separator 320
 - systemProperty 262
 - systemTest 262
- T**
- Tabbed pages, example 379
 - Tabbed Preference Pages 500
 - TabFolder
 - class 143, 172, 174
 - example 174
 - TabItem 172–174
 - Table
 - cell editors 372
 - cell validation 527
 - change listeners 374
 - class 143, 162, 165, 203
 - example 165
 - TableColumn 143, 162–163, 165, 206, 295, 309, 311, 481
 - TableColumnLayout 348
 - TableItem 162, 164, 295
 - Table Label Providers, hierarchy 195
 - Tables
 - auto-sizing 348
 - cell validator 376
 - editing vs. selecting 377
 - sorting 309
 - used in views 294
 - TableTree 143–144
 - tableTreeModified 378
 - TableViewer 328–329, 334
 - class 86, 203, 206, 294, 306–307, 309
 - example 205
 - TableViewerColumn 335, 523
 - TableViewerExample 206
 - TableViewers 203
 - Target, Ant 694
 - targetClass, itemType 642
 - targeted, viewContribution 273
 - targetID
 - editorContribution 280
 - perspectiveExtension 431
 - viewerContribution 271
 - visibility 278
 - Target Platform** preferences 798
 - Task List** view 66–68
 - TaskPropertiesDialog 446
 - Tasks, Ant 695
 - task scheduler 821
 - Tasks** view 428, 433, 533
 - Team > Create Patch...** command 801
 - Team > Synchronize with Repository** command 52
 - Team Development 49
 - TeamException 131
 - TeamStatus 131
 - tearDown 269
 - Templates 40, 42
 - templates for PDE 714
 - Templates** preferences 17, 41
 - test 262
 - TestCase, extends 101
 - Test Case** wizard 64
 - Testing
 - actions 253, 268, 275, 283
 - creating a plug-in test 100
 - creating a plug-in test project 100
 - creating JUnit test case 63
 - creating tests 63
 - introduction 63
 - plug-ins 99, 106
 - running a plug-in test 103
 - running tests 64
 - suite 255
 - using JUnit 58
 - views 345
 - testView 346
 - Text 143–144, 154, 211, 335, 451
 - TextCellEditor 335, 373, 376–377
 - TextEditor 360
 - TextEditorActionContributor 385
 - Text Editors** preferences 16
 - Text, multi line text example 156
 - TextPresentation 212–214
 - TextPropertyDescriptor 527
 - TextTransfer 324, 328, 330
 - TextViewer
 - class 211, 213
 - example 213
 - subclasses 211
 - TextViewerExample 213
 - Text viewers 211

- Thread 474
 - class 413
 - sleep 103, 418
- Throwable 456
- timerExec 149
- TimeTriggeredProgressMonitorDialog 417, 419
- Tips and Tricks** command 676
- TitleAreaDialog 444
- toc
 - anchor 586
 - bottom-up composition 587
 - extradir 586
 - file 586
 - href 588
 - primary 586
 - top-down nesting 587
 - topic 586
- tocgettingstarted.xml file 585–587, 590
- tocreference.xml file 585–586
- toc.xml file 585–586, 588
- Toggle Breakpoint** command 59
- toggleDetailsArea 461
- ToolBar 318
- ToolBar
 - action 273
 - views 318
- toolbar 228
 - org.eclipse.ui.main.toolbar 222
- toolbar
 - id 222
- toolbar button 245
- ToolBar Contribution 220
- toolbar item
 - editor 229
 - view 228
- toolbarPath 249
 - action 245, 282, 793
 - viewContribution 274
- Toolbars, workbench 13
- ToolItem 318
- ToolItem** 143
- tooltip 259
- tooltip
 - action 245, 259, 264, 272, 282, 790, 793
 - command 222
 - viewContribution 274
- tool tips 750
- top-level editor menu 280
- topLevelElementId property 715
- top-level menu discussion 255
- top-level menus 11
- toString 145
- Trac 65
- Tracker 143, 145
- Transfer 198–199, 324, 326, 329, 331
- TransferData 332
- translated files 619
- Traverse event 144
- TraverseListener 146
- TrayDialog 444
- Tree
 - class 143–144, 166, 168, 207, 382
 - example 168
- TreeColumn 362
- TreeColumnLayout 348, 362
- Tree event 144
- TreeItem 167–168
- TreeListener 166
- treeModified 374
- Trees
 - auto-sizing 348
- TreeViewer
 - ALL_LEVELS 371
 - class 207, 210, 360, 362
 - example 210
- TreeViewerColumn 372
- TreeViewerExample 210
- TreeViewers 207
- TwoPaneElementSelector 445
- TypeFilteringDialog 445
- Type Hierarchy** view 27
- Types
 - parameterized 794
 - referencing a class in a different plug-in 796
 - specified in an extension point 655, 793
 - structured parameters 795
 - unstructured parameters 794
- Types view 8
- U**
- UIJob 419, 811
- UI Thread
 - invalid thread access 413
 - SWT 148–149
 - SWTException 148
 - WorkspaceModifyOperation 414
- UML, EclipseUML 822
- undefined property 699
- underscore in Actions 285
- UNDO 399
- undo 395
- UndoActionHandler 398
- UndoActionHandler 398
- Undo, editor 392
- UndoHistoryView 400
- Uninstalling 104
- unless, Ant target 695

- unpack, plugin 666
- unregisterAdapters 787
- Unstructured parameters 794
- update 524
- update 147, 150, 200
- updateActionBars 399
- updateButtons 468
- updateColumnWidth 507
- Update command** 53
- updateEnablement 318
- updateLabel 308
- Update Manager 661, 667, 669, 671, 673, 679–680, 682, 685–686
- updateMessage 468
- updatePageComplete 467–468, 478
- Update Site Project 679
- Update sites
 - index.html 680
 - project 679
 - site.xml file 681
 - update and discovery URLs 685
 - using 679, 685
 - Web site 684
- updateSize 469
- updateTextEditorFromTableTree 380
- updateTextEditorFromTree 379, 381, 399
- updateTitleBar 468
- updateTreeFromTextEditor 371, 379, 399
- updateWindowTitle 468
- URL 122, 789
- url
 - action 790
 - feature 665
- Use Eclipse's string externalization mechanism** 622
- Use global properties checkbox** 705
- useJARFormat 89
- User 797
- Use Supertype Where Possible command** 44

V

- VA Assist xxxviii, xlv
- VAME 137
- Variables view 62–63
- VerifyEvent** 156
- Verify event 144
- VerifyKeyListener** 211
- VerifyListener** 155–156
- verifyText 156
- Version 128
- version
 - feature 665
 - plugin 666
- Version Synchronization dialog** 668

- View
 - createPartControl 251
 - identifier 238, 250
 - menu item 228
 - selection provider 251
 - site 251
 - toolbar item 228
- view
 - category 292, 618, 755
 - class 292, 618, 755
 - icon 293, 618
 - id 293, 432, 434, 618, 755
 - name 293, 618, 755
 - relationship 432, 434
 - relative 432, 434
 - visible 432, 434
- VIEW_ID 253
- ViewContentProvider 85
- viewContribution
 - class 273
 - icon 273
 - id 273–274
 - targeted 273
 - toolbarPath 274
 - tooltip 274
- Viewer 196–198
- viewerContribution
 - extension 270–271
 - id 271
 - targetID 271
- ViewerFilter
 - class 197, 199, 311
 - extends 311
 - subclasses 198
- Viewers
 - CheckboxTableViewer 203
 - CheckboxTreeViewer 207
 - content providers 195
 - filters 197
 - label providers 194
 - list example 202
 - list-oriented 193
 - ListViewers 200
 - overview 193
 - sorters 197
 - StructuredViewer 198
 - TableViewer example 205
 - TableViewers 203
 - text 211
 - TextViewer example 213
 - TextViewer subclasses 211
 - TreeViewer example 210
 - TreeViewers 207
 - update 524

- ViewerSorter
 - class 197, 200, 202, 308
 - extends 309
 - subclasses 197
- ViewFilter 198
- ViewLabelProvider 86
- ViewPart 293
 - class 85, 289
 - classes 290
- Views
 - actions 270
 - active view 6
 - Ant** 275
 - attributes 292
 - Bookmarks** 275
 - Breakpoints** 275
 - category 291
 - command 313
 - Console** 56, 275, 412, 553, 697, 801
 - content provider 306
 - context menu 314
 - contributions 314
 - createPartControl 293
 - CVS Repositories** 50, 297
 - Debug** 61, 276
 - declaration 291–292
 - detached 6
 - direct edit 333
 - Display** 63, 276
 - dispose 294
 - editors vs. views 290
 - Error Log** 131, 660
 - Expressions** 62, 276
 - fast views 6
 - Favorites** 84
 - filter 311
 - flicker 307
 - global actions 321
 - Hierarchy** 591
 - History** 46, 53
 - init 342
 - inline edit 333
 - introduction 5
 - IViewActionDelegate 271, 273
 - Java Package** 329
 - JUnit** 64
 - keyboard 320
 - label provider 307
 - linking 336
 - Members** 270, 276, 337
 - Memory** 276
 - menu 319
 - Navigator** 6, 9, 12, 21, 33, 276, 329, 788, 802
 - New Search** 29, 31
 - obtaining progress monitor 419
 - opening 6
 - opening an editor 339
 - Outline** 27–28, 400, 405, 427–428
 - Package Explorer** 7, 21, 33, 276
 - Packages** 276, 337
 - part 293
 - PDE views 96
 - Plug-in** 97
 - Plug-in Dependency** 97
 - Plug-in Registry** 96
 - Plug-ins 780
 - Problems** 276, 428, 533, 550, 553, 556, 798
 - Progress** 811
 - Projects** 276, 337
 - Properties** 511, 524, 527, 531
 - reference 289
 - Registers** 276
 - repositioning 6
 - resizing 6
 - RFRS 348
 - saveState 294, 342
 - saving local information 340
 - saving state 340
 - selection 312
 - selection listener 338
 - selection provider 317, 337
 - setFocus 294
 - show 119
 - Show View** 6
 - sorting 308
 - Tasks** 276, 428, 533
 - testing 345
 - Threads and Monitor** 276
 - toolbar 318
 - Type Hierarchy** 27
 - Types** 277, 337
 - Variables** 62–63, 277
 - viewerContribution 271
 - Welcome** 4
 - views
 - open view handler 237
 - viewShortcut 434, 436
 - ViolationResolutionGenerator 558–559
 - visibility
 - id 278
 - targetID 278
 - visible 433
 - actionSet 244
 - view 432, 434
 - visibleWhen 220, 222–224, 227, 229
 - activeEditorId 229
 - iterate 225
 - with 224
 - with variable 223–224, 229

- VisualAge for Java
 - IDE xxxvii, 8, 35, 46–47, 135
 - layout manager origins 178
- VisualAge Micro Edition 137
- VisualAge Smalltalk xxxvii
- VisualWorks 136
- vmargs 3
- VM arguments 57

- W**
- waitForJobs 101, 103, 253–254
- Watch** command 62
- WebBrowserEditor 789
- Web Browser** preferences 16
- Web delivery 679
- web directory 680
- Web page, opening programmatically 600
- Web resources location field 680
- Web Resources options 680
- Welcome** view 4
- Widget
 - class 143, 145
 - classes 144
 - events 142
- widgetDefaultSelected 158–159
- Widgets
 - constructors 144
 - disposal 142
 - Hierarchy 144
 - Layout 178
 - lifecycle 141
 - Resource Management 188
 - style bits 142
 - SWT 138
 - UI Thread 148
- widgetSelected 154, 158–159, 161–162, 165–166, 168–169, 177, 179, 310, 314, 318, 476
- Window
 - CANCEL 446
 - OK 447
- Window > Customize Perspective...** 14, 244, 252
- Window > Navigation > Quick Access** command 28
- Window > Open Perspective > Debug** command 10
- Window > Open Perspective > Java Browsing** 7
- Window > Open Perspective > Java** command 7
- Window > Open Perspective > Other** command 9
- Window > Open Perspective > Other...** command 50
- Window > Open Perspective** command 435
- Window > Preferences...** command 14
- Window > Reset Perspective** command 252
- Window > Show View > Other...** command 50, 96, 131
- Window > Show View** command 6, 435
- WindowBuilder Pro xlv, 347, 825
- windowImages 678
- Window** menu
 - items 12
 - Open Perspective** 435
 - Show View** 435
- Windows registry 3
- WindowTester xlv
- WindowTester Pro 827
- Wizard
 - class 465
 - extends 473
- wizard
 - class 470, 472
 - icon 470
 - id 470, 472
 - name 470, 472
- WizardDialog 417, 446, 464–465, 473
- WizardPage 475, 481
- Wizards
 - API 465
 - buttons 464
 - class hierarchy 464
 - content 475
 - content area 464
 - Default Plug-in Project 581
 - described 464
 - dialog settings 475
 - error message 478
 - example 473
 - example page 475
 - Export** 87
 - extension points 469
 - Externalize Strings** 621
 - Extract Strings** 475
 - help 466
 - IExportWizard 469
 - IImportWizard 470
 - INewWizard 470, 472
 - IWizard 465
 - IWizardContainer 468
 - IWizardPage 467
- Java Class** 24
- Java Package Wizard** 24
- Java Project** 19
- JUnit TestCase** 63, 67–68
- launching 469
- look and feel 482
- manually launching 473
- message 478
- nested 469
- New Extension** 424, 430, 485, 487, 596, 656
- New Extension Point** 639
- New Fragment Project** 630
- New Java Class** 249

Wizards (*continued*)

- New Plug-in Project** 581
- New Project** 72, 581, 663, 679
 - new project switches perspective 483
 - open new file in editor 483
 - page content based on prior page 480
- Plug-in project 72
- progress monitor 466
- Reference Projects 781
- RFRS 482
- setErrorMessages 478
- setHelpAvailable 466
- setMessage 479
- setNeedsProgressMonitor 466
- setVisible 481
- show new object 483
- warning message 478
- WizardDialog 465

WizardSelectionPage 469

Workbench

- CVS 49
- CVS label decorators 54
- File Extension Associations 47
- layout customization 8
- local history 46
- navigating 26
- obtaining progress monitor 419
- overview 3
- page 289
- preferences 15
- progress service 419
- refactoring 42
- Runtime 96
- status bar 418
- templates 40
- writing code 35

WorkbenchAdapter 788

WorkbenchLabelProvider 308, 788

WorkbenchPart 289–290, 353–354

WorkbenchPartLabelProvider 308

Workbench window

- actions 242
- menu 243

workbench.xml file 342

worked 416, 474, 809, 811

Working set name field 34

Working Sets

- defined 33
- New Working Set dialog** 34
- Select Working Sets dialog** 34

Working set scope 30

Workspace

- addSaveParticipant 420
- defined 110
- launch configuration plug-ins 95
- metadata 122, 345, 501
- modify operation 414
- scope 29–30

WorkspaceModifyOperation

- class 414, 416
- extends 414

Workspace preferences 16, 572

writeUTF 332

Writing Code 35

X

XML

- IMemento 340
- reading and writing 344

XMLBuddy 49

XMLMemento 343–345

XMLRootElementContentDescriber 358

Z

ZIP file 112