JBuilder 2007
Concepts

Concepts

Getting Started .......................................................... 10
  What's New ...................................................................... 11
  Introducing JBuilder 2007 .............................................. 15
  Tour of the User Interface (UI) ....................................... 17
  Help on Help .................................................................. 19
Migrating from Previous Versions of JBuilder ......................... 21
  JBuilder Project Migration Overview ............................. 22
  JBuilder 2006/JBuilder 2007 Differences ......................... 24
  JBuilder/Eclipse Dialog Box Equivalents ......................... 26
  JBuilder/Eclipse Menu Command and Keyboard Equivalents 27
  Project Properties ......................................................... 29
  Project Nodes ............................................................... 31
  Run Configuration ......................................................... 32
  Source Control .............................................................. 33
Importing Legacy Projects .................................................... 34
Java EE Applications Development ......................................... 36
  Java EE Applications Overview .................................... 37
  Runtime Servers .......................................................... 39
  The Web Tools Project (WTP) in JBuilder 2007 ............... 40
Developing EJB Applications ................................................ 41
  Enterprise Java Beans (EJB) Overview .......................... 42
  Enterprise Java Bean (EJB) Applications Overview .......... 43
  EJB Environment and Resources Overview .................... 44
  Deploying Enterprise Java Beans (EJBs) Overview .......... 45
  Entity Bean Overview .................................................... 46
  Message Bean Overview ................................................ 48
  EJB Security Roles Overview ......................................... 49
  Session Bean Overview .................................................. 50
Designing Web Services ..................................................... 51
  Web Services Overview .................................................. 52
  Web Services Explorer Overview .................................... 54
  Apache Axis Toolkit ....................................................... 56
Developing Web Applications .............................................. 59
  Web Applications Overview .......................................... 60
Developing Modeling Applications ....................................... 61
  Modeling Applications Overview .................................... 62
ProjectAssist and TeamInsight Concepts ................................. 63
  ProjectAssist and TeamInsight Overview ....................... 64
  Liferay: The TeamInsight Project Portal .......................... 68
  Subversion: Source Code Repository ............................. 70
  CVS: Source Code Repository ........................................ 72
  StarTeam: Source Code Repository, Change Request Tracking, and Task Provider 74
  Continuum/Maven: Continuous Build System .................. 76
  Mylar Concepts ............................................................ 77
  Bugzilla: Defect Tracking System ................................... 78
  XPlanner: Project and Team Management ........................ 79
Working with Peers .......................................................... 81
  Peer to Peer Collaboration ............................................. 82
Procedures

Tasks

JBuilder Project Migration .......................................................................................................................... 86
Building an Imported Project .................................................................................................................. 87
Import a Legacy Java RMI/JNI Project from JBuilder ................................................................. 89
Import Java EE Project From Existing Legacy JBuilder Project .................................................. 90
Import Java SE Project From Legacy JBuilder Project ................................................................. 92
Import Legacy Java VisiBroker Project ......................................................................................... 93
Importing a Source Controlled Project from a Previous Version of JBuilder ........................... 94
Running an Imported Project ............................................................................................................. 95
Setting Import Properties .................................................................................................................... 96

Java EE Applications ............................................................................................................................... 97
Creating a Java EE Project .................................................................................................................... 98
Developing Java EE Applications ........................................................................................................ 100
Import a Java EE Project ...................................................................................................................... 101
Publishing a Java EE Application to a Server Runtime ................................................................ 102
Running an Application on a Runtime Server ................................................................................ 103
Setting Up a Runtime Server ............................................................................................................ 104
Setting Up and Using a Borland Application Server ........................................................................ 106

Enterprise Java Bean (EJB) Applications ............................................................................................... 108
Adding a Business Method to an EJB .............................................................................................. 111
Adding a CMP Field to a CMP Entity Bean ..................................................................................... 112
Adding a Create Method to an EJB 2.x Entity Bean ....................................................................... 113
Adding a Find Method to an EJB 2.x Entity Bean .......................................................................... 114
Adding a Home Method to an EJB 2.x Entity Bean ....................................................................... 115
Adding a New Field to an Enterprise Java Bean (EJB) ..................................................................... 116
Adding a New Method to an EJB ....................................................................................................... 117
Adding a New Named Native Query to an EJB 3.0 Entity Bean ..................................................... 119
Adding a New Named Query to an EJB 3.0 Entity Bean ................................................................. 120
Adding a New Post-Load Method to an EJB 3.0 Entity Bean ......................................................... 121
Adding a New Post-Persist Method to an EJB 3.0 Entity Bean ....................................................... 122
Adding a New Post-Remove Method to an EJB 3.0 Entity Bean .................................................... 123
Adding a New Post-Update Method to an EJB 3.0 Entity Bean ..................................................... 124
Adding a New Pre-Persist Method to an EJB 3.0 Entity Bean ....................................................... 125
Adding a New Pre-Remove Method to an EJB 3.0 Entity Bean ..................................................... 126
Adding a New Pre-Update Method to an EJB 3.0 Entity Bean ........................................................ 127
Adding a Post-Construct Method to an EJB 3.0 Session Bean ..................................................... 128
Adding a Pre-Destroy Method to an EJB 3.0 Session Bean ......................................................... 129
Adding a Primary Key Join Field to an Entity Bean ........................................................................ 130
Adding a Result Set Mapping to an EJB 3.0 Entity Bean ............................................................... 131
Adding a Select Method to an EJB 2.x Entity Bean ....................................................................... 132
Adding a Timeout Method to an EJB 3.0 Session Bean ............................................................... 133
Adding an Interceptor Method to an EJB 3.0 Session Bean ............................................................ 134
Building a Package of Enterprise Java Beans (EJBs) ...................................................................... 135
Create an EJB Modeling Project with XDoclet Annotations ...................................................... 136
Creating a Bean-Managed-Persistence (BMP) Entity Bean .......................................................... 137
Creating a Container-Managed-Persistence (CMP) Entity Bean ................................................. 138
Creating a Message Bean .................................................................................................................. 139
Creating a Message Destination for a Message Bean ...................................................................... 140
Creating a Message Destination Link for a Message Bean ........................................................... 141
Creating a New Enterprise Java Bean (EJB) Project ....................................................................... 142
Creating a New Session Bean .......................................................................................................... 143
Creating a One-Way Relationship Between Entity Beans ............................................................. 144
Opening the Web Services Explorer ................................................................. 206
Running a Web Service ....................................................................................... 207
Setting Service Properties in the Web Services Explorer ................................ 208
Setting WSDL Properties in the Web Services Explorer ............................... 210
Testing the Web Service with the Client ............................................................ 212
Working in the Web Services Explorer .............................................................. 213
Opening the Web Services Explorer ................................................................. 214
Activating the Web Services Explorer for Existing Components .................. 215
Creating a New Web Service .............................................................................. 216
Creating a New WSDL Web Service in the Web Services Explorer .............. 217
Setting Service Properties in the Web Services Explorer ............................... 218
Setting WSDL Properties in the Web Services Explorer ............................... 220
Web Applications .............................................................................................. 222
Creating a Web Application Project ................................................................. 223
Enabling XDoclet ............................................................................................... 224
Modeling Applications ....................................................................................... 225
Create a Dynamic Web Java Persistence API (JPA) Modeling Project .......... 226
Create a Java Persistence API (JPA) Modeling Project ................................. 227
Create an EJB Modeling Project with XDoclet Annotations ......................... 228
Creating a Java Modeling Project ...................................................................... 229
Creating a Modeling Project ............................................................................. 230
Creating an Enterprise Java Bean (EJB) Modeling Project ............................ 231
EJB Modeling Project from a Java Project ....................................................... 232
Importing a Java Project as a Java Modeling Project ..................................... 233
Importing a Modeling Project .......................................................................... 234
Importing an Enterprise Java Bean (EJB) Modeling Project .......................... 235
Setting Up Database Connections .................................................................... 236
Connecting to an InterBase Database .............................................................. 237
Connecting to JDataStore .................................................................................. 238
ProjectAssist Procedures .................................................................................. 239
Adding and Configuring Projects through ProjectAssist ............................... 240
Adding and Configuring TeamInsight Users .................................................... 241
Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task) ... 242
Installing the ProjectAssist Components File .................................................. 243
Installing the ProjectAssist Components File for JBuilder 2007 Enterprise Borland ALM Edition ... 246
Mail Notification for TeamInsight Users ............................................................ 250
TeamInsight Procedures ................................................................................... 251
Adding Mylar Repositories for Bugzilla and XPlanner ................................. 253
Adding Mylar Repositories for StarTeam Change Requests or Task Planning .... 255
Adding Team Members in XPlanner (Administrator Task) ............................. 257
Administering the Liferay Portal ....................................................................... 259
Changing Your Passwords for the TeamInsight Tools .................................... 261
Checking Out a Project, Making Changes, and Checking Your Changes Into the Repository ............................ 262
Configuring Your TeamInsight Client ............................................................... 263
Creating and Starting Project Iterations in XPlanner (Administrator Task) .... 265
Creating or Generating Bug Reports in Bugzilla ............................................ 266
Logging in to TeamInsight Bugzilla ................................................................. 267
Managing Bug Reports in Bugzilla ................................................................. 268
Monitoring Iteration Metrics in XPlanner ....................................................... 269
Moving or Continuing a Story or Task in XPlanner ........................................ 270
Opening the TeamInsight Viewer and the Liferay Portal ............................... 272
Planning a Product Feature: Creating a User Story in XPlanner .................... 274
Planning Your Work: Creating Tasks in XPlanner .......................................... 275
Querying Bugzilla for Bug Reports ................................................................. 276
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tracking Your Time and Completing Tasks in XPlanner</td>
<td>278</td>
</tr>
<tr>
<td>Using Continuum/Maven for Continuous Integration Builds</td>
<td>280</td>
</tr>
<tr>
<td>Using the Subversion Viewer for Browsing the Project Repository</td>
<td>282</td>
</tr>
<tr>
<td>Peer to Peer Collaboration</td>
<td>284</td>
</tr>
<tr>
<td>Chatting with Peers</td>
<td>285</td>
</tr>
<tr>
<td>Enabling Peer to Peer Collaboration</td>
<td>286</td>
</tr>
<tr>
<td>Managing Contact Groups</td>
<td>288</td>
</tr>
<tr>
<td>Opening a Peer to Peer Session</td>
<td>290</td>
</tr>
<tr>
<td>Sending Data To Peers</td>
<td>292</td>
</tr>
<tr>
<td>Setting Collaboration Preferences</td>
<td>294</td>
</tr>
<tr>
<td>Sharing Team-Enabled Projects with Peers</td>
<td>296</td>
</tr>
</tbody>
</table>
## Reference

### IDE Reference

<table>
<thead>
<tr>
<th>Reference</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Import Dialogs</td>
<td>299</td>
</tr>
<tr>
<td>Project Import Wizard</td>
<td>300</td>
</tr>
<tr>
<td>Axis Web Service Dialogs Reference</td>
<td>301</td>
</tr>
<tr>
<td>New Dynamic Web Project: New Dynamic Web Project Wizard</td>
<td>302</td>
</tr>
<tr>
<td>New Dynamic Web Project: Project Facets</td>
<td>303</td>
</tr>
<tr>
<td>New Dynamic Web Project: Web Module</td>
<td>304</td>
</tr>
<tr>
<td>Dynamic Web JPA Modeling Dialogs Reference</td>
<td>305</td>
</tr>
<tr>
<td>New Dynamic Web Project: New Dynamic Web JPA Modeling Project</td>
<td>306</td>
</tr>
<tr>
<td>New Dynamic Web Project: Persistence unit settings page</td>
<td>307</td>
</tr>
<tr>
<td>New Dynamic Web Project: Project Facets</td>
<td>308</td>
</tr>
<tr>
<td>New Dynamic Web Project: Web Module</td>
<td>309</td>
</tr>
<tr>
<td>EJB Modeling Projects from XDoclet Dialogs Reference</td>
<td>310</td>
</tr>
<tr>
<td>EJB Modeling Project from XDoclet annotated WTP project</td>
<td>311</td>
</tr>
<tr>
<td>New EJB Modeling Dialogs Reference</td>
<td>312</td>
</tr>
<tr>
<td>EJB Modeling Project from Java Project</td>
<td>313</td>
</tr>
<tr>
<td>EJB Modeling Project from Java Project: Create New EJB Project from Java Project</td>
<td>314</td>
</tr>
<tr>
<td>EJB Modeling Project from Java Project: Project Facets</td>
<td>315</td>
</tr>
<tr>
<td>JPA Modeling Dialogs Reference</td>
<td>316</td>
</tr>
<tr>
<td>New JPA Modeling Project: Create a JPA modeling project</td>
<td>317</td>
</tr>
<tr>
<td>New JPA Modeling Project: Persistence unit settings page</td>
<td>318</td>
</tr>
<tr>
<td>New JPA Modeling Project: Java Settings</td>
<td>319</td>
</tr>
<tr>
<td>ProjectAssist and TeamInsight Dialogs</td>
<td>320</td>
</tr>
<tr>
<td>New ProjectAssist File Link</td>
<td>321</td>
</tr>
<tr>
<td>New ProjectAssist File</td>
<td>322</td>
</tr>
<tr>
<td>New ProjectAssist File:Select Stack Components</td>
<td>323</td>
</tr>
<tr>
<td>Maven Project from Archetype</td>
<td>324</td>
</tr>
<tr>
<td>New ProjectAssist file: Choose disk scan paths</td>
<td>325</td>
</tr>
<tr>
<td>TeamInsight Viewer</td>
<td>326</td>
</tr>
<tr>
<td>Edit Repository Query or New XPlanner Query</td>
<td>327</td>
</tr>
<tr>
<td>TeamInsight User Mail Notification</td>
<td>328</td>
</tr>
<tr>
<td>ProjectAssist Mail Preferences</td>
<td>329</td>
</tr>
<tr>
<td>Passwords for Authorization</td>
<td>330</td>
</tr>
<tr>
<td>Preinstalled Component Scan: Choose scan type</td>
<td>331</td>
</tr>
<tr>
<td>ProjectAssist Configuration Editor: Projects</td>
<td>332</td>
</tr>
<tr>
<td>ProjectAssist Configuration Editor: Stacks</td>
<td>334</td>
</tr>
<tr>
<td>ProjectAssist Configuration Editor: Users</td>
<td>336</td>
</tr>
<tr>
<td>New StarTeam Query</td>
<td>337</td>
</tr>
<tr>
<td>StarTeam Repository Settings</td>
<td>338</td>
</tr>
<tr>
<td>Peer to Peer Dialogs Reference</td>
<td>339</td>
</tr>
<tr>
<td>Peer To Peer Preferences</td>
<td>340</td>
</tr>
<tr>
<td>Peers View</td>
<td>341</td>
</tr>
<tr>
<td>New Contact Group</td>
<td>342</td>
</tr>
<tr>
<td>Send Stack Trace</td>
<td>343</td>
</tr>
<tr>
<td>Send Web Link</td>
<td>344</td>
</tr>
<tr>
<td>Send VCS Link</td>
<td>345</td>
</tr>
</tbody>
</table>
Concepts
Concepts

This section lists the conceptual information provided with JBuilder 2007.

In This Section

Getting Started
This section contains overview information to get started with JBuilder 2007.

Migrating from Previous Versions of JBuilder
This section provides information on migrating from previous versions of JBuilder.

Java EE Applications Development
This section contains overview information regarding Java EE application development within JBuilder 2007.

ProjectAssist and TeamInsight Concepts
TeamInsight is a set of project tools that enable development teams to coordinate their work and to optimize their efforts. ProjectAssist provides the server install, configuration and assimilation of these components by the ProjectAssist Administrator.

Working with Peers
This section contains information on peer to peer collaboration.
**Getting Started**

The topics in this section describe how to get started using JBuilder 2007.

**In This Section**

- Introducing JBuilder 2007
  JBuilder makes collaborative development fast and reliable for Java, open source and the web

- What's New

- Tour of the User Interface (UI)
  Describes the JBuilder 2007 User Interface (UI).

- Migrating from Previous Versions of JBuilder
  This section provides information on migrating from previous versions of JBuilder.

- Help on Help
  Describes online Help and typographic conventions.
What's New

This topic describes the new features and functions offered with JBuilder 2007.

JBuilder 2007 is the leading cross-platform environment for building industrial-strength enterprise Java applications and the only integrated development environment (IDE) currently supporting virtual peer to peer programming. JBuilder 2007 speeds project development and database application development with intuitive two-way visual designers and rapid deployment to application servers.

The Eclipse Platform

JBuilder 2007 powers productivity on the open-source Eclipse-based platform with support for:

- Java 2 Standard Edition (J2SE™) 5.0 (JDK® 1.5)
- UML® code visualization
- Distributed refactoring
- Code audits
- Enterprise unit testing
- Support for multiple version-control systems
- Build in quality with integrated performance tools and Java EE profiling.

Other features include:

- New refactoring
- Additional search options
- Improved error navigation

Re-factoring options now include:

- Introduce Parameter
- Extract Inner
- Hoist Declaration
- Split Variable Declaration

Among the many new search capabilities, developers can now search for files by name in any directory tree including those defined by a folder or directory view node in the active project.

Searching the content of files identified in a prior search is now available and searching is faster due to a new identifier cache. The editor’s new optional global gutter view shows, in one glance, all errors in the current file and the relative location of those errors in the file.

Modeling

JBuilder 2007 uses the Borland Together modeling product to implement its modeling perspective. JBuilder 2007 provides round-trip integration between the source code and modeling views. When you change the model, the source code changes to match. The model also reflects changes to the source code.

JBuilder 2007 provides modeling support for:

- Java SDK 1.5, Java SE, and Java EE
- Enterprise Java Beans (EJB) 2.x and 3.0
- Java Persistence API (JPA) projects
- Axis toolkit-based Web Services projects
- Database Application Development with CodeGear's InterBase and JDataStore database systems

JBuilder 2007 can import a model from an XML schema, an Xdoclet-annotated WTP project, a Java project, or an EJB project.

**Ability to Migrate Your Projects from Previous Versions of JBuilder**

The Java perspective contains a code editor, a Package Explorer that is similar to the previous JBuilder Project pane, an Outline view that is similar to the previous JBuilder Structure pane, and a tabbed lower pane, for searching and error display, that is similar to the previous JBuilder Message pane. There is also a Debug perspective, a Java Browsing perspective, and a Java Type Hierarchy perspective that are similar to panes in the previous JBuilder IDEs.

You can import any type of JBuilder project created with a previous version of JBuilder into the JBuilder 2007/Eclipse workspace, using one of the Import JBuilder Project wizards (Java or Java EE). Java EE conversion includes conversion from XML descriptors to XDoclet annotations (EJB 2.1) or to EJB 3.0 annotations.

**Enhanced EJB Modeler**

JBuilder's EJB Modeler is an enhanced version of previous JBuilder EJB designers. It includes support for EJB 2.x and 3.0 features, and support for JSR 181.

**Server Runtime Installation for Web Applications, Web Services, and EJBs.**

JBuilder 2007 allows you to install pre-configured versions of several of the most popular runtime servers, including:

- Apache Geronimo 1.1.1
- Apache Tomcat 5.5
- JBoss 4.0.5 GA
- GlassFish V1 UR 1

*Tip:* A best practice when using the runtime servers is to choose one of the versions that has (CodeGear or Borland) after it. These versions have been extended to support specific JBuilder 2007 features.

JBuilder 2007 is also compatible with:

- Borland Application Server 6.6 or 6.7, with Tibco or OpenJMS
- IBM WebSphere 6.0
- IBM WebSphere 6.1
- BEA WebLogic Application Server 8.1
- BEA WebLogic Application Server 9.2
- BEA WebLogic Application Server 10.0
- Oracle 10.1.3 Application Server
- Oracle Containers for Java (OC4J) 10.1.3

In addition, the Web Services Designer helps you implement your WSDL, WSIL, and UDDI files with ease.
Developer Versions of InterBase and JDataStore

JBuilder 2007 includes two database systems — Borland's InterBase and JDataStore.

You can create a visual model of your database application using JBuilder 2007's Modeling Perspective.

ProjectAssist and TeamInsight

Designed to help organizations manage and balance complex development projects across teams and locations and across open source and proprietary software, ProjectAssist and TeamInsight provides a blended development "stack in the box" that contains:

- Requirements Tracking
- Bug Tracking
- Source Code Management
- Project Management across organizations and time zones

To accomplish this, JBuilder 2007 embraces and integrates the most popular "best of breed" open source plug-ins, tools and frameworks and provides a certified and managed turn-key development solution on which organizations of any size can rely.

This team-coordination feature is based on two user types, the Administrator (who performs the ProjectAssist server configuration) and the User (who uses the installed TeamInsight client tools to develop or test software with the integrated products listed above).

Peer to Peer Collaboration and Project Wikis

In addition to the team enhancing features of the ProjectAssist and TeamInsight component install and team tools, other specific features of JBuilder 2007 provide for the overall team collaboration and cooperation. Peer to peer collaboration features allow two or more users to collaborate across a local area network (LAN) and send data. Project peers can chat with one another, organize contacts into groups, and share projects.

The following peer-to-peer collaboration features enable developers to collaborate as if working in front of the same workstation:

- Shared editing
- Joint debugging
- Active differencing

An Eclipse Wiki is bundled with JBuilder 2007 and provides a powerful project documentation tool for team members to coordinate efforts, disperse unified project information, and take full advantage of the open-source community. The Eclipse Wiki plug-in is provided with JBuilder 2007; however, it must be enabled and configured according to documentation provided on the web or in the Eclipse Wiki Help files included with JBuilder 2007.

3rd Party Plug-ins

This version of JBuilder includes many other third-party plug-ins that you can use with your existing Eclipse implementations. You can also add your other Eclipse plug-ins to this version of Eclipse with JBuilder. For a full-list of the included products and plug-ins, Choose the 'Contents' section of the Welcome page.
Related Concepts

Help on Help
Tour of the User Interface (UI)
Migrating from Previous Versions of JBuilder
Java EE Applications Overview
Enterprise Java Bean (EJB) Applications Overview
Web Applications Overview
Web Services Overview
Modeling Applications Overview
ProjectAssist and TeamInsight Overview
Peer to Peer Collaboration
**Introducing JBuilder 2007**

JBuilder 2007 is the first enterprise-class integrated development environment (IDE) built on the open-source Eclipse platform. It embraces and integrates the most popular "best of breed" plug-ins, tools and frameworks and provides a certified and managed turn-key development solution organizations of any size can rely on. Designed to increase developer velocity while bringing balance and confidence to Java development with both commercial and open source components, JBuilder 2007 provides the ability to collaborate on development projects and integrate open source and commercial development in a single, managed environment through the ProjectAssist/TeamInsight feature and components and:

- Provides the same collaborative capabilities, Optimizeit profiling, EJB design, and enterprise-class Rapid Application Development (RAD) features JBuilder's reputation is built on, with improvements made possible by the Eclipse platform. You can migrate your existing JBuilder projects to the JBuilder 2007 platform, as well as adding other Eclipse plugins that you're currently using.
- Installs pre-configured versions of several of the most popular runtime servers, including: JBoss, Apache Tomcat, Geronimo, and Glassfish to get you developing Web Applications, Web Services, and EJBs faster. Yet JBuilder 2007 is also compatible with Borland Application Server, WebLogic, WebSphere, Oracle and other products that you may have purchased separately. In addition, the Web Services Explorer helps you implement your WSDL, WSIL, and UDDI files with ease.
- Includes a Modeling Perspective that delivers Borland's Together — an innovative and highly productive visual "drag and drop" environment with enterprise-class project management capabilities designed to increase the speed and productivity of individuals and development teams
- Increases the development velocity of Java teams and individuals via ProjectAssist's Requirements Tracking, Bug Tracking, Source Code Management, and Project Management.
- Decreases database application development time with developer versions of Borland's InterBase and JDataStore applications.

For more detailed information about these new features, click the 'What's New' link at the bottom of this topic, or go directly to the overview for each of the features described above. If you can't wait to get started coding, you can go to the Help Table of Contents, and choose your favorite tasks from a list of Procedures.

**Getting Started**

Review the following topics to become familiar with JBuilder 2007 features and tools:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introducing JBuilder 2007</td>
<td>This topic provides a high level overview of JBuilder 2007, introduces important features and concepts, and introduces available help resources.</td>
</tr>
<tr>
<td>What's new in JBuilder 2007</td>
<td>This topic provides a high level overview of JBuilder 2007, with information about migrating to JBuilder 2007 from a previous JBuilder release.</td>
</tr>
<tr>
<td>Using the Eclipse Help System</td>
<td>This topic provides information about Eclipse platform functionality.</td>
</tr>
</tbody>
</table>

**Using Online Help**

Use the Online Help system to find conceptual, procedural, and reference information about CodeGear's JBuilder 2007 plug-in for Eclipse. Where appropriate, this Help system provides references to online help provided by Eclipse, as well as to online help provided by third parties and other plug-ins that are shipped with JBuilder 2007.
Cheat Sheets
You can also view Cheat Sheets through the Help menu. Cheat Sheets walk you through specific tasks and open the wizards associated with these tasks to give you easy-access to the components you need. View the Cheat Sheets by selecting the Help tab followed by the Cheat Sheets… command..

Video Demos
JBuilder 2007 Narrated videos of specific JBuilder 2007 features, such as importing projects, and developing web services are also available at the CodeGear Developer Network TV and at the JBuilder product web site.

Developer Support and Resources
CodeGear provides a variety of support options and information resources to help developers get the most out of CodeGear products. These options include a range of CodeGear Technical Support programs, as well as free services on the Internet, where you can search our extensive information base and connect with other users of CodeGear products. Access the CodeGear Developers Network (CDN) using the link below.

Related Concepts
- Getting Started
- What's New
- Help on Help
- JBuilder 2006/JBuilder 2007 Differences
- JBuilder Project Migration Overview
- Java EE Applications Overview
- Enterprise Java Bean (EJB) Applications Overview
- Web Services Overview
- Web Applications Overview
- Modeling Applications Overview
- The Web Tools Project (WTP) in JBuilder 2007
- ProjectAssist and TeamInsight Overview

Related Reference
- Using the Eclipse help system
- CodeGear Developers Network
Tour of the User Interface (UI)

This topic introduces the following JBuilder 2007 UI subjects:

- Eclipse-specific UI elements
- JBuilder Compatibility
- Peer to Peer Collaboration
- ProjectAssist/TeamInsight

Using the Eclipse Workbench

The JBuilder 2007 user interface is integrated into the Eclipse Workbench. To become familiar with the JBuilder 2007 UI, see the Eclipse help topic “Using the Workbench” which discusses the following user interface elements:

- Resources
- Resource Hierarchies
- Linked Resources
- Path Variables
- Working Sets
- Builds
- Local History

In addition to reviewing the primary workbench interfaces, see the following Workbench Help subtopics:

- Perspectives
- Editors
- Views
- Toolbars
- Markers
- Bookmarks
- Label decorations
- External tools
- Accessibility features in Eclipse
- Tasks

Tip: The Eclipse Workbench and Perspective views will be new to previous JBuilder users. Explore the help topics below to become familiar with these key UI elements.

JBuilder Compatibility

Previous JBuilder projects are easily converted into projects with the Project Import Wizard. Review the following topics to become familiar with compatibility features:

- JBuilder 2006/JBuilder 2007 Differences
- JBuilder Project Migration Overview
- Importing a Java EE Project from JBuilder
Peer to Peer Collaboration

Peer to peer collaboration is easily instantiated **Peers View**. Review the following topics to become familiar with Peer to Peer collaboration features:

- Peer to Peer Collaboration
- Enabling Peer to Peer Collaboration

ProjectAssist and TeamInsight

The ProjectAssist and TeamInsight open-source development tools are packaged in JBuilder 2007 to provide project management and coordination tools to development groups. Review the following topics to become familiar with ProjectAssist and TeamInsight:

- Installing and Configuring the ProjectAssist Server (Administrator Task)
- ProjectAssist and TeamInsight Overview
- TeamInsight Procedures

Related Concepts

- ProjectAssist and TeamInsight Overview
- JBuilder 2006/JBuilder 2007 Differences
- JBuilder Project Migration Overview
- Importing Legacy Projects
- Peer to Peer Collaboration
- ProjectAssist and TeamInsight Overview

Related Tasks

- JBuilder Project Migration
- Peer to Peer Collaboration
- Building an Imported Project
- Enabling Peer to Peer Collaboration
- Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
- TeamInsight Procedures

Related Reference

- Eclipse Help Topic: “Workbench”
- Eclipse Help Topic: “Perspectives”
- Eclipse Help Topic: “Working with Perspectives”
- Eclipse Help Topic: “Working with editors and views”
Help on Help

You can access multiple Help resources to find information about JBuilder 2007.

Online Help

Online Help provides detailed information about the features available in JBuilder 2007 for JBuilder Project Migration, Web Service and EJB design, software release management tools, and Impact Analysis. To view the JBuilder 2007 Help table of contents, choose Help ▸ Help Contents, then JBuilder 2007. To locate help topics, enter a search term into the Search field. You can narrow the search scope to selected topics by clicking the Search Scope button.

Online Help uses a multi-tiered, top-down approach to help you become familiar with the tools and features of JBuilder 2007. When you open online Help, conceptual, task, and reference information is available. Conceptual information gives you access to general overview information. Task information provides step-by-step instructions to perform many of the tasks described at the conceptual level. Reference information includes topics on the wizards and dialog boxes. For additional help resources, see the Readme.

Refer to the following list to determine the type of online Help information that specifically addresses your needs.

- If you are new to JBuilder 2007 or just want a product overview, see the “Getting Started” conceptual and task areas.
- To learn about migrating your previous JBuilder projects into the Eclipse workspace, see the “JBuilder Project Migration” conceptual and task areas.
- For information about the managing your projects with the ProjectAssist features, see the “ProjectAssist” and “TeamInsight” conceptual and task areas.

Cheat Sheets

You can also use Cheat Sheets, located in the Help menu to follow step-by-step procedures that allow you to open wizards and dialog boxes as you perform specific tasks. Some of the Cheat Sheets available with this release include: Importing a JBuilder 2006 Project as a Java EE Project, Getting Started with an EJB 3.0 Modeling Project, ProjectAssist Administration, and Importing Entity Beans from a Database.

Videos

Although they have not been incorporated into the regular online help yet, videos are another good way to learn about JBuilder 2007. Narrated videos of specific JBuilder 2007 features, such as importing projects, and developing web services are also available at Borland Developer Network TV (http://bdn.borland.com/bdntv/java) and the JBuilder Product Web Site (http://www.borland.com/us/products/jbuilder/index.html).

Typographic Conventions

The following typographic conventions are used throughout the JBuilder 2007 online Help.

<table>
<thead>
<tr>
<th>Typographic convention</th>
<th>Used to indicate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monospace type</td>
<td>Source code and text that you must type, file names, and directories.</td>
</tr>
<tr>
<td>Boldface</td>
<td>Reserved language keywords or compiler options, references to dialog boxes and tools.</td>
</tr>
<tr>
<td>Italics</td>
<td>Book titles and new terms.</td>
</tr>
<tr>
<td>KEYCAPS</td>
<td>Keyboard keys, for example, the CTRL or ENTER key.</td>
</tr>
</tbody>
</table>
Migrating from Previous Versions of JBuilder

JBuilder 2007 provides a migration path from previous versions of JBuilder to JBuilder 2007 (JBuilder on Eclipse), so you can develop, test, and run your previously-created JBuilder projects in the Eclipse workspace.

In This Section

- JBuilder Project Migration Overview
  Describes the migration path from previous versions of JBuilder to JBuilder 2007 (JBuilder on Eclipse).

- JBuilder 2006/JBuilder 2007 Differences
  Summarizes differences between JBuilder 2006 and JBuilder 2007 development environments

- JBuilder/Eclipse Dialog Box Equivalents
  Summarizes some previous JBuilder IDE and Eclipse dialog box equivalents.

- JBuilder/Eclipse Menu Command and Keyboard Equivalents
  Summarizes some JBuilder and Eclipse menu commands and keyboard equivalents.

- Project Properties
  Describes JBuilder 2007 project properties

- Project Nodes
  Describes JBuilder 2007 project nodes

- Run Configuration
  Describes JBuilder 2007 run configurations

- Source Control
  Describes JBuilder 2007 source control options

- Importing Legacy Projects
  Describes various project import scenarios with JBuilder 2007
JBuilder Project Migration Overview

You can import any type of Java project created in a previous release of JBuilder into the JBuilder 2007 Eclipse workspace, including:

- Java SE (formerly J2SE) projects
- Java EE (formerly J2EE) projects
- VisiBroker projects
- RMI/JNI projects

The project import wizard does not copy the JBuilder source files and folders directly into the workspace, instead links are created using the Eclipse resource link capability. The Eclipse project file in the workspace maps a resource name, for example, /src, to an absolute path name, for example, C:/MyProject/src/java. New files that are added to the project are added to the original source folder. Projects under source control may be able to check it into the JBuilder 2007/Eclipse workspace.

Created Workspace Files and Folders

The following files and folders are created in the workspace folder for a project imported from a previous version of JBuilder:

- .classpath: The linked resources file (XML source).
- .project: The Java project file (XML source).
- /bin: The output folder.

**Warning:** If the Enabled Linked Resource option on the Linked Resources page of the Preferences dialog box (Window ▶ Preferences ▶ General ▶ Workspace ▶ Linked Resources) is off, the project import may fail. If this happens, the following message will be displayed in the Import Status dialog box: Error creating source path link for <project name>. Linked resources are not supported by this application.

The Eclipse build process uses the standard JDK compiler, not the previous JBuilder compiler, Borland Make for Java. Before you build your imported project, you can check compiler options on the Java Compiler page of the Properties dialog box.

Using the Import Project Wizard

The import project wizard can translate Java EE and Java SE legacy JBuilder projects to JBuilder 2007. Following are the notable differences in projects:

- Splitting of the legacy module into multiple project types
- Server configuration for compiling, running and deploying
- XML descriptors are converted to XDoclet annotations

When a project is converted the Project import log displays conversion information including data regarding any artifacts that were not created in the conversion process. Not all J2EE artifacts are converted. Currently EJB Web EAR and EJB web services client projects are supported. Application client and JBoss service archives must be converted manually.
Projects with Generated Source

When a project, such as a VisiBroker or RMI project, has auto-generated sources that are output to the /Generated Sources folder in the classes folder, the /Generated Sources folder is not imported. However, when you build the project the source files are automatically generated and placed in a /Generated Sources folder in the Eclipse workspace. The Derived setting on the Info page of the Properties dialog box (Properties ▶ info from the context menu in the Package Explorer with the folder selected) indicates that this folder is auto-generated.

Unsupported Properties

Some project properties are not supported in Eclipse or are translated to the Eclipse equivalent on import. The following table illustrates those items:

<table>
<thead>
<tr>
<th>Project Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/Additional Settings Folder</td>
<td>Not imported; no equivalent.</td>
</tr>
<tr>
<td>/doc Folder</td>
<td>Not imported. Regenerate with File ▶ Export ▶ Javadoc.</td>
</tr>
<tr>
<td>/bak Folder</td>
<td>Not imported.</td>
</tr>
<tr>
<td>jbInit() Method</td>
<td>Left in code.</td>
</tr>
<tr>
<td>@todo Tags</td>
<td>Left in code.</td>
</tr>
</tbody>
</table>

Related Concepts

- JBuilder 2006/JBuilder 2007 Differences
- JBuilder/Eclipse Dialog Box Equivalents
- JBuilder/Eclipse Menu Command and Keyboard Equivalents
- Project Properties
- Project Nodes
- Run Configuration
- Source Control
- Importing Legacy Projects

Related Tasks

- Import Java EE Project From Existing Legacy JBuilder Project
- Import Java SE Project From Legacy JBuilder Project
JBuilder 2006/JBuilder 2007 Differences

The Eclipse platform, modeled as a plug-in development environment, provides an end-to-end Java development platform. Plug-ins help create an adaptable and extensible system. The Eclipse environment provides perspectives, editors, and views that can be added to, configured, or replaced through the implementation of plug-ins.

The JBuilder plug-in for Eclipse adds views and editors to the existing Eclipse Java perspective, as well as providing a modeling perspective and an integrated set of development life-cycle management tools.

Perspectives

An Eclipse perspective provides a “flavor” for the Eclipse development environment and defines the initial set and layout of views in the Workbench. Each perspective provides a set of functionality aimed at accomplishing a specific type of task. As you work in the Workbench, you will probably switch perspectives frequently. Perspectives are available from the Window ▶ Open Perspective menu command. You can set perspective preferences with the Window ▶ Preferences ▶ General ▶ Perspectives command.

The Java perspective contains a code editor, a Package Explorer that is similar to the previous JBuilder Project pane, an Outline view that is similar to the previous JBuilder Structure pane, and a tabbed lower pane, for searching and error display, that is similar to the previous JBuilder Message pane. There is also a Debug perspective, a Java Browsing perspective, and a Java Type Hierarchy perspective that are similar to panes in the previous JBuilder IDEs.

JBuilder 2007 adds views and editors to the Java perspective that are specific for developer needs, such as tools for editing code, viewing and editing requirements and change requests, profiling, and creating unit tests. JBuilder 2007 also adds a Modeling perspective so that you can do most of these tasks while looking at a modeling view of your Java code. You can customize these perspectives with the Window ▶ Customize Perspective command.

Editors

Most Eclipse perspectives contain one or more editors for editing code. Eclipse editors include a Java source code editor, a text editor, and a GUI visual editor. JBuilder 2007 includes the modeling designer, requirements editor, and a change request editor. You can open as many editors as you wish, though only one editor is active at a time. The main menu and toolbar only contain items applicable to the active editor.

Views

Views provide alternative presentations of data. Views have unique context menus and may have unique toolbars. A view can be displayed on its own, or as a tabbed page in a multi-view presentation. JBuilder 2007 provides multiple views, including the Modeling Perspective, the Requirements view, and the Profiling view.

Note: Each view contains a toolbar with a drop-down menu icon.

Tips

There are many slight differences between the previous JBuilder IDEs and the Eclipse user interface. The tips below can help you learn to navigate the Eclipse Java perspective quickly. Note that these tips are not extensive or exhaustive, and cover just some of the frequently used features.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Tip</th>
</tr>
</thead>
<tbody>
<tr>
<td>Editor</td>
<td>If a Java file has errors, a red “X” icon is displayed in the left margin of the editor. Hovering the mouse over the icon displays the error as a tooltip.</td>
</tr>
</tbody>
</table>
### Editor
- When the editor cannot display tabs for all open files due to space constraints, the number of files not displayed is shown on a toolbar button. Click the button to see a file list.
- When using Code Assist (code insight in JBuilder 2006), a tooltip with available Javadoc is displayed.
- The Navigate → Open Type Hierarchy command displays the type hierarchy of a specific source code element.
- Hovering the mouse over a symbol displays Javadoc for that symbol, if available.
- Clicking the mouse on an identifier marks all uses of that identifier in the current file. Locations where the identifier is used are marked in the gutter.
- Typing a left-facing parenthesis, brace, or quote automatically adds the termination/closing mark.
- Placing a caret in a symbol highlights all of its occurrences in the open file.
- The gutter indicates lines of code that have changed.
- Using the Navigate menu, you can search for references by a range of scopes, from the workspace to the current project to the current class hierarchy to just a selected group of files.
- Previous searches are available from a drop-down menu in the Search view.
- You can use the Java Search page of the Search dialog box (Search → Java) to search for the particular usage of a symbol.
- Use the Change Method Signature refactoring to modify the signature of a method.
- Optimize imports and code formatting can be applied to a group of files.
- You can search for references on a selected import statement (Search → References).
- When a change is made, saved, and compiled during a debugging session, obsolete frames are automatically popped off the stack and the frame pointer is automatically set to the highest possible valid frame.
- To evaluate an expression, first execute the code (Run → Execute), then display the results (Run → Display).
- Only one Eclipse instance can be debugged at a time.
- Icons in the Variables view indicate the type of variable, for example, members or local variables.
- In the Debug perspective, right-click an application and choose Terminate All to remove all terminated launches.
- Breakpoints can be configured to stop when a condition changes, not just on a true/false condition. Breakpoints can also be configured to stop only in a particular thread.

### Related Concepts
- JBuilder Project Migration Overview
- JBuilder/Eclipse Menu Command and Keyboard Equivalents
- JBuilder/Eclipse Dialog Box Equivalents

### Related Tasks
- Importing Legacy Projects

### Related Reference
- Perspectives in the Eclipse Workbench User Guide
- Editors in the Eclipse Workbench User Guide
- Views in the Eclipse Workbench User Guide
# JBuilder/Eclipse Dialog Box Equivalents

The following tables show previous JBuilder IDE and Eclipse dialog box equivalents.

## Project Properties dialog box

<table>
<thead>
<tr>
<th>JBuilder</th>
<th>Eclipse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paths page</td>
<td>Java Build Path page</td>
</tr>
<tr>
<td>JDK option on the Paths page</td>
<td>Libraries tab on the Java Build Path page</td>
</tr>
<tr>
<td>Output path on the Paths page</td>
<td>Output Path option on the Java Build Path page</td>
</tr>
<tr>
<td>Source tab on the Paths page</td>
<td>Source tab on the Java Build Path page</td>
</tr>
<tr>
<td>Documentation tab on the Paths page</td>
<td>Javadoc Location page</td>
</tr>
<tr>
<td>Required Libraries tab on the Paths page</td>
<td>Libraries tab on the Java Build Path page</td>
</tr>
<tr>
<td>Compiler options on the Java page</td>
<td>Compiler page in the Preferences dialog box</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>JBuilder</th>
<th>Eclipse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure Libraries dialog box</td>
<td>User Libraries page in the Preferences dialog box</td>
</tr>
<tr>
<td>Configure JDKs dialog box</td>
<td>Installed JREs page in the Preferences dialog box</td>
</tr>
<tr>
<td>Browser page in the Preferences dialog box</td>
<td>Web Browser page in the Preferences dialog box</td>
</tr>
<tr>
<td>Editor page in the Preferences dialog box</td>
<td>Web Browser page in the Preferences dialog box</td>
</tr>
<tr>
<td>Documentation tab on the Paths page</td>
<td>Javadoc Location page</td>
</tr>
<tr>
<td>Required Libraries tab on the Paths page</td>
<td>Libraries tab on the Java Build Path page</td>
</tr>
</tbody>
</table>

26
JBuilder/Eclipse Menu Command and Keyboard Equivalents

The following tables show JBuilder and Eclipse menu command and keyboard equivalents.

**Note:** If no keyboard shortcut is listed, none is available.

### File menu

<table>
<thead>
<tr>
<th>JBuilder</th>
<th>Eclipse</th>
</tr>
</thead>
<tbody>
<tr>
<td>File ▶ New (CTRL+N)</td>
<td>File ▶ New (CTRL+N)</td>
</tr>
<tr>
<td>File ▶ Open File (CTRL+O)</td>
<td>File ▶ Open File</td>
</tr>
<tr>
<td>File ▶ Close (CTRL+F4)</td>
<td>File ▶ Close (CTRL+F4)</td>
</tr>
<tr>
<td>File ▶ Close All (CTRL+SHIFT+F4)</td>
<td>File ▶ Close All (CTRL+SHIFT+F4)</td>
</tr>
<tr>
<td>File ▶ Save (CTRL+S)</td>
<td>File ▶ Save (CTRL+S)</td>
</tr>
<tr>
<td>File ▶ Save All (CTRL+SHIFT+S)</td>
<td>File ▶ Save All (CTRL+SHIFT+S)</td>
</tr>
</tbody>
</table>

### Edit menu

<table>
<thead>
<tr>
<th>JBuilder</th>
<th>Eclipse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit ▶ Undo (CTRL+Z)</td>
<td>Edit ▶ Undo (CTRL+Z)</td>
</tr>
<tr>
<td>Edit ▶ Redo (CTRL+SHIFT+Z)</td>
<td>Edit ▶ Redo (CTRL+Y)</td>
</tr>
<tr>
<td>Edit ▶ Cut (CTRL+X)</td>
<td>Edit ▶ Cut (CTRL+X)</td>
</tr>
<tr>
<td>Edit ▶ Copy (CTRL+C)</td>
<td>Edit ▶ Copy (CTRL+C)</td>
</tr>
<tr>
<td>Edit ▶ Paste (CTRL+V)</td>
<td>Edit ▶ Paste (CTRL+V)</td>
</tr>
<tr>
<td>Edit ▶ Format All (ALT+SHIFT+K)</td>
<td>Source ▶ Format (CTRL+SHIFT+F)</td>
</tr>
<tr>
<td>Edit ▶ Code Insight (CTRL+SPACE)</td>
<td>Edit ▶ Content Assist (CTRL+SPACE)</td>
</tr>
<tr>
<td>Edit ▶ Code Insight ▶ Parameter Insight (CTRL+SHIFT+SPACE)</td>
<td>Edit ▶ Parameter Hints (CTRL+SHIFT+SPACE)</td>
</tr>
<tr>
<td>Edit ▶ Code Insight ▶ Javadoc QuickHelp (CTRL+O)</td>
<td>Navigate ▶ Open External Javadoc (SHIFT+F2)</td>
</tr>
<tr>
<td>Edit ▶ Select All (CTRL+A)</td>
<td>Edit ▶ Select All (CTRL+A)</td>
</tr>
</tbody>
</table>

### Search menu

<table>
<thead>
<tr>
<th>JBuilder</th>
<th>Eclipse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search ▶ Find (CTRL+F)</td>
<td>Edit ▶ Find/Replace (CTRL+F)</td>
</tr>
<tr>
<td>Search ▶ Find In Path (CTRL+P)</td>
<td>Search ▶ Search ▶ Java Search (CTRL+H)</td>
</tr>
<tr>
<td>Search ▶ Replace (CTRL+R)</td>
<td>Edit ▶ Find/Replace (CTRL+F)</td>
</tr>
<tr>
<td>Search ▶ Search Again (F3)</td>
<td>Edit ▶ Find Next (CTRL+K)</td>
</tr>
<tr>
<td>Search ▶ Search Incremental (CTRL+E)</td>
<td>Edit ▶ Incremental Find (CTRL+J)</td>
</tr>
<tr>
<td>Search ▶ Go To Line (CTRL+G)</td>
<td>Navigate ▶ Go To Line (CTRL+L)</td>
</tr>
<tr>
<td>Search ▶ Go To Class Member (CTRL+SHIFT+G)</td>
<td>Select class member, then Navigate ▶ Go To ▶ Next Member (CTRL+SHIFT+UP)</td>
</tr>
<tr>
<td>Search ▶ Go To Previous Method</td>
<td>Select method, then Navigate ▶ Go To ▶ Previous Member (CTRL+SHIFT+DOWN)</td>
</tr>
<tr>
<td>Search ▶ Go To Next Method</td>
<td>Navigate ▶ Open Type (CTRL+SHIFT+T)</td>
</tr>
<tr>
<td>Search ▶ Find Classes (CTRL+MINUS)</td>
<td>Navigate ▶ Open Declaration (F3)</td>
</tr>
<tr>
<td>Search ▶ Find Definition (CTRL+ENTER)</td>
<td>Search ▶ References ▶ Project</td>
</tr>
<tr>
<td>Search ▶ Find References ▶ Javadoc QuickHelp (CTRL+SHIFT+ENTER)</td>
<td>Search ▶ References ▶ Hierarchy</td>
</tr>
<tr>
<td>Search ▶ Find Referring Classes</td>
<td></td>
</tr>
</tbody>
</table>

### Refactor menu
<table>
<thead>
<tr>
<th>JBuilder</th>
<th>Eclipse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refactor►Optimize Imports (CTRL+I)</td>
<td>Source►Organize Imports (CTRL+I)</td>
</tr>
<tr>
<td>Refactor►Rename</td>
<td>Refactor►Rename (ALT+SHIFT+R)</td>
</tr>
<tr>
<td>Refactor►Move</td>
<td>Refactor►Move (ALT+SHIFT+V)</td>
</tr>
<tr>
<td>Refactor►Inline</td>
<td>Refactor►Inline (CTRL+SHIFT+I)</td>
</tr>
<tr>
<td>Refactor►Change Parameters (CTRL+S)</td>
<td>Refactor►Change Method Signature (ALT+SHIFT+C)</td>
</tr>
<tr>
<td>Refactor►Extract Interface From</td>
<td>Refactor►Extract Interface</td>
</tr>
<tr>
<td>Refactor►Extract Method (CTRL+SHIFT+E)</td>
<td>Refactor►Extract Method (ALT+SHIFT+M)</td>
</tr>
<tr>
<td>Refactor►Surround with Try/Catch (CTRL+SHIFT+C)</td>
<td>Source►Surround with Try/Catch Block</td>
</tr>
</tbody>
</table>

### Project menu

<table>
<thead>
<tr>
<th>JBuilder</th>
<th>Eclipse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project►Make Project (CTRL+F9)</td>
<td>Project►Build All (CTRL+B)</td>
</tr>
<tr>
<td>Project►Rebuild Project</td>
<td>Project►Build All (CTRL+B)</td>
</tr>
<tr>
<td>Project►Make &lt;File&gt; (CTRL+SHIFT+F9)</td>
<td>Project►Build All (CTRL+B)</td>
</tr>
<tr>
<td>Project►Rebuild &lt;File&gt;</td>
<td>Project►Build All (CTRL+B)</td>
</tr>
<tr>
<td>Project►Make Project Group</td>
<td>Project►Build Working Set</td>
</tr>
</tbody>
</table>
Project Properties

Once a project has been imported into the Eclipse workspace, you can right click the project node and choose Properties to view project properties, including the build and output paths, library settings, and compiler options.

Paths

When a JBuilder Java project is imported, without doing a checkout, the project’s source path remains in the /src folder in the project’s original location. If you check out a project from version control, all source files are placed in the Eclipse workspace and the source path is relative to the workspace.

The JBuilder classpath is analogous to the Java build path in Eclipse. The build path is displayed on the Java Build Path page of the Properties dialog box. By default, the output path is the /bin folder in the Eclipse workspace, not the /classes folder, as in JBuilder. You can change the path on the Java Build Path page of the Properties dialog box.

Note: The Eclipse Package Explorer does not display projects in their build order. To see the build order for multiple projects, open the Build Order page of the Preferences dialog box (Window ▶ Preferences ▶ General ▶ Workspace ▶ Build Order).

Libraries

Libraries are saved to the Eclipse workspace. Libraries that are required for the project are displayed on the Libraries tab of the Java Build Path page in the Properties dialog box.

The project import compares JDK version labels and translates the project JDK to the JRE in the eclipse/jre folder of your Eclipse installation. Subsequent imports of additional projects search for a JDK with the same version as an already-imported JDK. If one exists, that JDK is used, instead of creating multiple, identical JREs.

Note: The project import brings in both project libraries and libraries that those libraries require.

Compiler Options

Imported compiler options are display on the Java Compiler page of the Properties dialog box. If the compiler compliance level for the workspace is different from that of the imported project, the import makes a project-specific override. For VisiBroker projects, the Compiler Compliance Level on the Java Compiler page needs to be set to 1.4 or 1.3.

JBuilder-Specific Properties Pages

Property pages are supplied for JBuilder specific properties, such as VisiBroker or RMI/JNI projects. To view these property pages, right-click an IDL or Java interface file after the project import. Press F1 on these pages for more information.
Related Concepts

- JBuilder Project Migration Overview
- Source Control
- Project Nodes
- Run Configuration
- Importing Legacy Projects

Related Tasks

- Setting Import Properties
- Building an Imported Project
- Running an Imported Project
Project Nodes

A JBuilder project can have multiple nodes, including J2EE nodes, archive nodes, Javadoc nodes, a Generated Source node, a build node, and so on. Not all nodes can be imported into the Eclipse workspace. The following nodes are not imported:

- Archive node
- Javadoc node
- Generated Source node

If your project has an Archive node, you can recreate the archive with the File ▶ Export ▶ Archive File command. You can regenerate Javadoc with the File ▶ Export ▶ Javadoc command.

On project import, auto-generated source files are not imported or automatically regenerated. However, when you build your imported project, the generated source files are created in the /Generated Source folder of the Eclipse workspace. The /Generated Source folder is added to the source path on the Source tab of the Java Build Path page of the Properties dialog box (Project ▶ Properties ▶ Java Build Path). In Eclipse, auto-generated files are referred to as Derived files. The Derived setting is on the Info page of the Properties dialog box (Properties ▶ Info from the context menu in the Package Explorer with the folder selected).

Related Concepts

- JBuilder Project Migration Overview
- Source Control
- Project Properties
- Run Configuration
- Importing Legacy Projects

Related Tasks

- Setting Import Properties
- Building an Imported Project
- Running an Imported Project
Run Configuration

When a Java project from JBuilder is imported into the Eclipse workspace, the run configuration is also imported. This configuration includes run and debug settings. You can view the run configuration in the Run dialog box (Run ➤ Run). Configurations are sorted by type in the tree on left. In Eclipse, a run configuration is known as a launch configuration.

Related Concepts

- JBuilder Project Migration Overview
- Source Control
- Project Properties
- Project Nodes
- Importing Legacy Projects

Related Tasks

- Setting Import Properties
- Building an Imported Project
- Running an Imported Project
Source Control

JBuilder projects under source control can be checked out to the Eclipse workspace. When you check out directly from the repository into the workspace, an Eclipse project is created without any of the project elements that are not files. The check out pulls all source files into the workspace.

JBuilder projects can be checked out from the following source control systems:

- Subversion
- ClearCase
- CVS
- StarTeam
- Visual SourceSafe

**Note:** Subversion, StarTeam and CVS are the only source code control systems supported by the ProjectAssist/TeamInsight features. StarTeam and CVS can be assimilated from an existing installation only.

**Warning:** CVS and Subversion projects that are checked into a local repository cannot be checked out.

If the project is under source control, the Enable VCS Plugin option on both Import JBuilder Project wizards is enabled. If the project is under source control and you do not select this option, the JBuilder project is imported from its original location.

**Note:** Before the check out, you may be required to log into the repository or synchronize the local version with the version in the repository.

Related Concepts

- JBuilder Project Migration Overview
- Project Properties
- Project Nodes
- Run Configuration
- Importing Legacy Projects

Related Tasks

- Setting Import Properties
- Building an Imported Project
- Running an Imported Project
Importing Legacy Projects

Import the following types of Java project created with a previous version of JBuilder:

- Java EE
- Java SE
- VisiBroker
- RMI/JNI

Java EE Project

The project import wizard creates a project for each module from legacy JBuilder project modules, with shared source code. The modules table lists the Java EE modules found in the legacy project including Java versions and the corresponding JBuilder 2007 project created during the conversion process. The module table displays Java and Java EE versions (project facets) for JBuilder 2007 project.

EJB Projects

JBuilder 2007 supports EJB 2.x development using XDoclet annotations and EJB 3.0 development using Java EE 5.0 annotations. The project import wizard converts legacy JBuilder project XML descriptors to either XDoclet annotations (for EJB 2.1) or to Java EE 5.0 annotations (for EJB 3.0).

Note: For EJB 2.1 interfaces are not copied to the EJB project (or the utility project) from the legacy JBuilder project (interfaces are generated using XDoclet).

Web Services

Java SE Project

Use the import wizard to import legacy JBuilder home directory files and libraries as required to properly import a Java SE project. The following legacy artifacts are imported in the project conversion:

- Libraries
- Runtime Configurations
- Javadoc Options
- Java Compiler Options default file encoding
- Java files

Java RMI/JNI Project

The java.rmi package provides classes for Java Remote Method Invocation (RMI). Use Remote Method Invocation (RMI) enables to create distributed Java-to-Java applications, where the methods of remote Java objects can be invoked from other Java virtual machines, possibly on different hosts. RMI uses object serialization to marshal and un-marshal parameters and does not truncate types, supporting true object-oriented polymorphism.

Build the RMI/JNI project in JBuilder to expand any build macros that used as VM arguments. Expand the project in the Package Explorer and select an RMI or JNI file. Use the Properties dialog to display the Properties for <filename> dialog box. Use the RMI/JNI Properties page to view property settings imported from JBuilder.
VisiBroker Project

Open the VisiBroker page of the Preferences dialog box to verify the directory where VisiBroker tools are installed. Expand the project in the Package Explorer and select an IDL file or a Java interface file that will be translated from IDL, to IDL, or to IIOP. Use Properties to display the Properties for <filename> dialog box and confirm Property settings have been imported.

- For IDL to Java files, choose the VisiBroker IDL Properties page and verify options in the IDL2Java Settings area of the dialog box.
- For Java to IDL files, choose the VisiBroker Java Properties page and verify options in the Java2IDL Settings area of the dialog box.
- For Java to IIOP files, choose the VisiBroker Java Properties page and verify options in the Java2IIOP Settings area of the dialog box.

Related Concepts

- JBuilder Project Migration Overview
- JBuilder 2006/JBuilder 2007 Differences

Related Tasks

- Import Java EE Project From Existing Legacy JBuilder Project
- Import Java SE Project From Legacy JBuilder Project
- Creating a Java EE Project
Java EE Applications Development

The topics in this section describe developing Java EE applications with.

In This Section

Java EE Applications Overview
An overview of Java EE applications in JBuilder 2007.

Runtime Servers
Concept topic providing an overview of runtime server support.

The Web Tools Project (WTP) in JBuilder 2007
Describes using WTP with the IDE.

Developing EJB Applications
Concept topic on developing Enterprise Java Bean (EJB) applications using JBuilder 2007.

Designing Web Services
Concept topic with information on designing web services using JBuilder 2007.

Developing Web Applications
Concept topic for developing web applications in JBuilder 2007

Developing Modeling Applications
Concept topic regarding developing modeling application through JBuilder 2007.
Java EE Applications Overview

JBuilder 2007 is a Java integrated development environment (IDE). Coupled with a supported application server, the JBuilder 2007 development platform ensures creation of distributed enterprise applications that are:

- Reliable and scalable to process business transactions quickly and accurately
- Secure to protect user privacy and the integrity of enterprise data
- Readily available to meet the increasing demands of the global business environment

Tip: Use the links below to discover these topics in detail.

Java EE with JBuilder

Use JBuilder 2007 to speed up and simplify development of client server application, web applications, and UML based diagramming with support for:

- Enterprise Java Beans (EJB)
- Web Tools Project (WTP) conversion to EJB modeling projects
- Web Services
- Java Persistence API (JPA)

Tip: Use the links below to discover these topics in detail.

Java EE applications

To efficiently create Java EE applications using JBuilder 2007 become familiar with the following topics:

- Creating EJB and web services projects
- Setting Up a Runtime Server
- Publishing a Java EE Application to a Server Runtime
- Running an Application on a Runtime Server

Tip: Use the links below to discover these topics in detail.

Supported Runtime Servers

JBuilder 2007 provides support and installs the following runtime servers:

- Apache Geronimo 1.1.1
- Apache Tomcat 5.5
- JBoss 4.0.5 GA
- GlassFish V1 UR 1

In addition, JBuilder 2007 is also compatible with several other runtime servers. Please refer to the related concept topic on runtime servers.
Related Concepts

Creating a Java EE Project
Web Applications Overview
Runtime Servers
Web Services Overview
Modeling Applications Overview
Enterprise Java Bean (EJB) Applications Overview

Related Tasks

Setting Up a Runtime Server
Publishing a Java EE Application to a Server Runtime
Running an Application on a Runtime Server

Related Reference

Eclipse help topic “J2EE architecture Web Application Developer's Guide”
Runtime Servers

A server runtime environment is used to test, debug and run a project. It provides the environment, libraries and infrastructure that a "server" needs. A server is an instance of the server runtime used to host web applications and other server-side components.

The following runtimes are bundled with JBuilder 2007:

1. Apache Geronimo 1.1.1
2. Apache Tomcat 5.5
3. JBoss 4.0.5 GA
4. GlassFish V1 UR 1

Tip: A best practice when using the runtime servers is to choose one of the versions that has (CodeGear or Borland) after it. These versions have been extended to support specific JBuilder 2007 features.

The following products are supported by JBuilder 2007 but must be purchased separately to be used as runtime servers:

- Borland Application Server 6.6 or 6.7, with Tibco or OpenJMS
- IBM WebSphere 6.0
- IBM WebSphere 6.1
- BEA WebLogic Application Server 8.1
- BEA WebLogic Application Server 9.2
- BEA WebLogic Application Server 10.0
- Oracle 10.1.3 Application Server
- Oracle Containers for Java (OC4J) 10.1.3

Related Concepts

- Java EE Applications Overview
- Web Applications Overview
- Web Services Overview
- Enterprise Java Beans (EJB) Overview

Related Tasks

- Creating a Java EE Project
- Setting Up a Runtime Server
- Publishing a Java EE Application to a Server Runtime
- Running an Application on a Runtime Server
- Creating a Web Application Project

Related Reference

- Borland Application Server Documentation
- Eclipse help topic "Server targeting for Web applications"
- Eclipse help topic "Web Projects"
- Eclipse help topic "Web archive (WAR) files"
- Eclipse help topic "Server targeting for web applications"
The Web Tools Project (WTP) in JBuilder 2007

The Eclipse Web Tools Platform (WTP) Project provides APIs for J2EE and Web-centric application development. It includes both source and graphical editors for a variety of languages, wizards and built in applications to simplify Web Service development, and tools and APIs to support deploying, running, and testing applications. The ultimate objective of the project is to provide highly reusable and extensible tooling for application production efficiency. WTP provides infrastructure for:

- Web Standard Tools
- J2EE Standard Tools

Tools provided will include editors, validators and document generators for artifacts developed in a wide range of standard languages and a specialized workbench supporting actions such as publish, run, start and stop of Web application code across target server environments.

The Web Standard Tools Project includes server tools which extend the Eclipse platform with servers as first-class execution environments. Server tools provide an extension point for generic servers to be added to the workspace, and to be configured and controlled.

JBuilder 2007 runtimes extend WTP runtimes in cases where the WTP runtime does not support a certain server version. Use these runtimes when working with application servers in JBuilder 2007. The following runtimes are provided:

1. Apache Geronimo 1.1.1
2. Apache Tomcat 5.5
3. JBoss 4.0.5 GA
4. GlassFish V1 UR1

Note: Supported Runtimes are noted with “(CodeGear or Borland)”.

Related Concepts

- Creating a Java EE Project
- Runtime Servers
- Web Services Overview
- Enterprise Java Bean (EJB) Applications Overview

Related Tasks

- Setting Up a Runtime Server
- Publishing a Java EE Application to a Server Runtime
- Running an Application on a Runtime Server

Related Reference

- Eclipse help topic “J2EE architecture Web Application Developer's Guide”
Developing EJB Applications

This section discusses developing applications with Enterprise Java Beans (EJB).

In This Section

- **Enterprise Java Beans (EJB) Overview**
  Describes Enterprise Java beans (EJBs).

- **Enterprise Java Bean (EJB) Applications Overview**
  Describes Enterprise Java Bean (EJB) applications.

- **EJB Environment and Resources Overview**
  Describes environment and resources references in Enterprise Java Beans (EJBs).

- **Entity Bean Overview**
  Describes entity beans.

- **Session Bean Overview**
  Describes session beans.

- **Message Bean Overview**
  Describes message beans.

- **Deploying Enterprise Java Beans (EJBs) Overview**
  Describes the deployment of Enterprise Java Beans (EJBs).

- **EJB Security Roles Overview**
  Describes the security roles in Enterprise Java Beans (EJBs).
Enterprise Java Beans (EJB) Overview

Enterprise Java Beans (EJBs) are server-side components for modular construction of enterprise applications. EJBs implement a component architecture for distributed transaction-oriented applications.

Entity beans provide access to your database tables to other Java beans on the application server. Session beans provide an external interface from the application server to your Java clients. Message beans respond to asynchronous communications from Java clients.

Accessing Data with Entity Beans

Entity beans represent business data and functionality. Entity beans provide a class interface to underlying relational database structure. Entity beans provide access to database fields, keys, and relationships. Entity beans handle communications with the database server. Entity beans provide data management functionality for session and message beans. For more information on entity beans, refer to the “Accessing Data with Entity Beans” link in the Related Information section at the bottom of this page.

Providing an Interface with Session Beans

Session beans represent business processes. A session bean is a short-lived bean that executes on behalf of a single client. Session beans provide an external interface for client applications. Public session bean methods handle business activities, accessing entity beans as needed and returning data to the client. For more information on session beans, refer to the “Providing an Interface with Session Beans” link in the Related Information section at the bottom of this page.

Asynchronous Communications with Message Beans

A message bean implements business process logic in response to receipt of an asynchronous message through the Java Messaging Service (JMS). A client sends a message to a JMS destination that is associated with your EJB container. When the message arrives, the container passes it to the OnMessage method of a message bean. The message bean then performs the requested service using the message as its input. For more information on session beans, refer to the “Asynchronous Communications with Message Beans” link in the Related Information section at the bottom of this page.

Note: The EJB 3.0 specification is significantly different from the EJB 2.x specification. JBuilder 2007 supports both EJB 2.x and EJB 3.0 applications. Please make sure that you use the correct version of EJB for your application.

Related Concepts

- Enterprise Java Bean (EJB) Applications Overview
- Entity Bean Overview
- Session Bean Overview
- Message Bean Overview

Related Tasks

- Creating a Java Class for a Web Service
- Modifying an Enterprise Java Bean (EJB)
Enterprise Java Bean (EJB) Applications Overview

Enterprise Java Beans are the server-side components for Java platform applications. EJBs provide database access to client-side Java applications. EJB applications move functionality into a thick server, allowing you to write the code once for the Java beans and use it everywhere in client applications.

Java beans are developed locally and deployed to a runtime server (also called a container or application server). The presentation client accesses EJBs on the runtime server.

Enterprise Java Beans (EJBs)

Java beans are developed to handle common data manipulation tasks, business processes, and asynchronous events. For more information on EJBs, refer to the “Enterprise Java Beans (EJBs)” link in the Related Information section at the bottom of this page.

Deploying EJBs to the Runtime Server

After EJBs have been developed and tested, they are deployed to a runtime server to be used by client applications. For more information on deploying EJBs, refer to the “Deploying EJBs to the Runtime Server” link in the Related Information section at the bottom of this page.

Note:  The EJB 3.0 specification is significantly different from the EJB 2.x specification. JBuilder 2007 supports both EJB 2.x and EJB 3.0 applications. Please make sure that you use the correct version of EJB for your application.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- Deploying Enterprise Java Beans (EJBs) Overview

Related Tasks

- Creating an Enterprise Java Bean (EJB) Project
- Importing an Enterprise Java Bean (EJB) Modeling Project
EJB Environment and Resources Overview

The enterprise bean's environment provides for customized bean data at runtime without the need to access or change the enterprise bean's source code. The environment provides beans with an object-independent way to refer to a value, a resource, or another component. The value of such environment references (or variables) is set at deployment time, based on the contents of the deployment descriptor.

EJB References

Each EJB reference describes the interface between the referencing enterprise bean and the referenced bean. You can define references between beans within the same JAR file or from an external enterprise bean (one that is outside the JAR file but in the same application).

Each EJB local reference describes the interface between the referencing enterprise bean and the local referenced bean.

For information on adding EJB References, refer to the “Creating an EJB Reference” link in the Related Information section at the bottom of this page.

Resource References

A resource reference identifies a resource factory reference of the enterprise bean. A set of resource references enables the application assembler or the bean deployer to locate all the references used by the bean. For information on adding a resource reference, refer to the “Creating a Resource Reference” link in the Related Information section at the bottom of this page.

Environment Entries

The environment entry allow you to customize the bean's business logic when the bean is assembled or deployed, without the need to access or change the bean's source code directly. Each enterprise bean defines its own set of environment entries. All instances of an enterprise bean share the same environment entries. Enterprise bean instances aren't allowed to modify the bean's environment at runtime. For information on adding an environment entry, refer to the “Creating an Environment Entry” link in the Related Information section at the bottom of this page.

Environment Resource References

A resource environment reference provides a logical name for a physical object. The client application uses the logical name to find the resource at runtime. For information on adding an environment resource reference, refer to the “Creating an Environment Resource Reference” link in the Related Information section at the bottom of this page.

Related Concepts

- Enterprise Java Bean (EJB) Applications Overview
- Deploying Enterprise Java Beans (EJBs) Overview

Related Tasks

- Creating an EJB Reference
- Creating a Resource Reference
- Creating an Environment Entry
- Creating an Environment Resource Reference
Deploying Enterprise Java Beans (EJBs) Overview

Enterprise Java Beans (EJBs) are deployed to a runtime server (also known as a container or application server) for later use by client applications.

Security in an EJB environment is handled by the container, not the bean business methods. EJB security roles provide a bean-independent way to provide access to enterprise bean methods.

EJBs abstract away from the underlying server architecture. Environment and resource references provide a way for enterprise beans to refer to available resources without relying on a particular server configuration.

Using the Runtime Server

Bean suppliers deploy their provided EJBs to a runtime server, where they are available to client applications. For more information on using the runtime server, refer to the “Using the Runtime Server” link in the Related Information section at the bottom of this page.

EJB Security Roles

EJB security information is handled separately from bean business methods. This allows the bean deployer to configure security in the most appropriate way for the operational environment. EJB security roles provide a way for bean developers to permit access to bean methods to different categories of users. For more information on security roles, refer to the “EJB Security Roles” link in the Related Information section at the bottom of this page.

EJB Environment and Resources

For more information on EJB environment and resources, refer to the “EJB Environment and Resources” link in the Related Information section at the bottom of this page.

Related Concepts

- Enterprise Java Bean (EJB) Applications Overview
- EJB Security Roles Overview
- EJB Environment and Resources Overview
Entity Bean Overview

This section describes EJB entity beans. Entity beans represent business data that is stored in a database. Each entity bean stands for an individual item instance in the underlying data store. Entity beans provide an object-oriented interface to the underlying relational database. Entity beans persist across client calls and can be shared by multiple clients.

Session beans access business data through entity beans. The entity beans implement common database functions, shielding the session beans from the underlying database schema.

Entity Bean Persistence

The state of an entity bean consists of its underlying data. The entity bean's state exists beyond the lifetime of the application. An entity bean is persistent because its state exists even after you shut down the database server or the applications it services. Entity beans manage their persistence by staying synchronized with the data store. Entity bean persistence can be managed either by the bean code itself (bean-managed persistence) or by the bean container (container-managed persistence). With bean-managed persistence, you write the SQL code to access the database. With container-managed persistence, the EJB container handles the database access for you.

Container-managed persistence is more portable than bean-managed persistence. The code for a CMP entity bean contains no SQL code and is thus independent of the underlying database. Container-managed persistence allows you to redeploy the entity bean on a different J2EE server without modifying or recompiling your code.

Entity Bean Relationships

Each entity bean has a unique object identifier. The unique identifier, or primary key, enables the client to locate a particular data item. An entity bean can be related to other entity beans in the same way that database rows and be related to rows in other tables. One entity bean contains the primary key for another entity bean as part of its data. The two entity beans can be joined by matching these fields and the resulting data used.

Note: The EJB 3.0 specification is significantly different from the EJB 2.x specification. JBuilder 2007 supports both EJB 2.x and EJB 3.0 applications. Please make sure that you use the correct version of EJB for your application.
Related Concepts

- Enterprise Java Beans (EJB) Overview

Related Tasks

- Creating a Container-Managed-Persistence (CMP) Entity Bean
- Creating a Bean-Managed-Persistence (BMP) Entity Bean
- Importing Entity Beans from a Database
- Creating a One-Way Relationship Between Entity Beans
- Creating a Relationship Between Entity Beans
- Adding a CMP Field to a CMP Entity Bean
- Creating the Primary Key for an Entity Bean
- Adding a Primary Key Join Field to an Entity Bean
- Adding a New Named Query to an EJB 3.0 Entity Bean
- Adding a New Named Native Query to an EJB 3.0 Entity Bean
- Adding a Result Set Mapping to an EJB 3.0 Entity Bean
- Adding a New Pre-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Pre-Remove Method to an EJB 3.0 Entity Bean
- Adding a New Pre-Update Method to an EJB 3.0 Entity Bean
- Adding a New Post-Load Method to an EJB 3.0 Entity Bean
- Adding a New Post-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Post-Remove Method to an EJB 3.0 Entity Bean
- Adding a New Post-Update Method to an EJB 3.0 Entity Bean
- Adding a Home Method to an EJB 2.x Entity Bean
- Adding a Find Method to an EJB 2.x Entity Bean
- Adding a Select Method to an EJB 2.x Entity Bean
Message Bean Overview

This section describes message beans. A message bean provides asynchrony to EJB applications by acting as a JMS (Java Messenging Service) message consumer. A message bean is associated with a JMS topic or queue and receives JMS messages sent by EJB clients or other beans. Like stateless session beans, message beans maintain no client-specific state. Clients send JMS messages to message beans. A message bean listens for messages, using a single onMessage method to process received messages. When a message arrives, the container ensures that a message bean corresponding to the message topic/queue exists, and calls its onMessage method with the client's message as the single argument.

A message bean retains no data or state for a specific client. A single message-driven bean can process messages from multiple clients.

Message Destinations

The message destination indicates the JMS destination type (topic or queue) to which the message bean will bind. The message bean functions as a full-fledged JMS client, indistinguishable from any other JMS client. In addition to functioning as asynchronous JMS clients, message beans also support message concurrency. Since message beans are stateless and managed by the container, they can both send and receive messages concurrently (the container simply grabs another bean out of the pool).

The OnMessage Method

When a message arrives, the container calls the message-driven bean's onMessage method to process the message. The onMessage method normally casts the message to one of the five JMS message types and handles it in accordance with the application's business logic. The onMessage method can call helper methods, or it can invoke a session or entity bean to process the information in the message or to store it in a database.

Note:  The EJB 3.0 specification is significantly different from the EJB 2.x specification. JBuilder 2007 supports both EJB 2.x and EJB 3.0 applications. Please make sure that you use the correct version of EJB for your application.

Related Concepts

Enterprise Java Beans (EJB) Overview

Related Tasks

Creating a Message Bean
Creating a Message Destination for a Message Bean
Creating a Message Destination Link for a Message Bean
EJB Security Roles Overview

Security roles define the permission required to run EJB methods. A security role is a set of logically related method permissions. The application assembler defines the security roles and method permissions for each set of deployed EJBs. A user must have at least one security role associated with a method in order to invoke the method.

Defining Security Roles

The application assembler specifies the methods of the remote and home interface that each security role is allowed to invoke. The assembler defines the method permissions relation in the deployment descriptor. Each method-permission element includes a list of security roles and a list of methods. The listed security roles may invoke the listed methods. A security role or a method can appear in multiple method-permission elements.

Running As a Security Role

An EJB, Java control, or web service method can run under the security role of the invoking user, or it can run under a different security role. This might be necessary when the EJB uses resources that have strict security requirements.

Related Concepts

- Enterprise Java Bean (EJB) Applications Overview
- EJB Security Roles Overview

Related Tasks

- Creating a Security Role
- Creating a Security Role Reference
- Creating a Run-As-Security Link
Session Bean Overview

This section describes session beans. Session beans provide an interface from the EJB container to client applications. Session beans implement business processes through entity beans.

Session beans represent business tasks, not persistent data. For example, a Session bean might perform a database search for a user and return the results to the user. Session beans can communicate with all other types of beans, and can thus be used for many tasks other than database transactions.

A session bean is composed of a component interface, a home interface, a bean implementation class, and a deployment descriptor. The component interface contains the client business methods of the bean. The home interface contains methods for the bean life cycle. The bean implementation class implements all the methods that allow the bean to be managed in the container. The deployment descriptor contains bean properties that can be edited at assembly or deployment time.

Session Bean States

A session bean is a short-lived bean that executes on behalf of a single client. Stateless session beans do not preserve state across method calls. Each call to a session bean invokes a standard stateless session bean with no memory of previous calls to the same session bean.

Stateful session beans preserve their states within and between transactions. If the client invokes method calls against the same bean stub, the calls are sent to the same bean instance in the container. Field variables in the bean instance retain their values as long as the client application retains the bean reference.

Component Interface

Business methods provide the component interface of the session bean to the client. Each business task provided by the session bean is represented in a business method. Clients call the business methods of the session bean to manage business tasks for them. For more information on adding a component method, consult the “Adding a Business Method” link in the Related Information section at the bottom of this page.

Note: The EJB 3.0 specification is significantly different from the EJB 2.x specification. JBuilder 2007 supports both EJB 2.x and EJB 3.0 applications. Please make sure that you use the correct version of EJB for your application.

Related Concepts

Enterprise Java Beans (EJB) Overview

Related Tasks

Creating a New Session Bean
Adding a Business Method to an EJB
Adding an Interceptor Method to an EJB 3.0 Session Bean
Adding a Post-Construct Method to an EJB 3.0 Session Bean
Adding a Pre-Destroy Method to an EJB 3.0 Session Bean
Adding a Timeout Method to an EJB 3.0 Session Bean
Designing Web Services

The web services features in JBuilder 2007 allow you to quickly design, deploy, run, and test a web service.

In This Section

- **Web Services Overview**
  Provides overview information on web services and the JBuilder 2007 implementation of web services.

- **Apache Axis Toolkit**
  Provides overview information on the Apache Axis web services toolkit.

- **Web Services Explorer Overview**
  The Web Services Explorer provides a design surface for quickly creating, implementing, and validating web services.
Web Services Overview

You can create software that performs a set of tasks, and then make it available to others by running it on a web server over a network, such as the Internet or a local area network. A web service’s public interfaces and bindings are defined and described in a service description language. Developers can then discover these service descriptions and use them to write a client application to invoke and access your services.

The following runtime servers and/or containers are provided with JBuilder 2007:

- Apache Geronimo 1.1.1
- Apache Tomcat 5.5
- JBoss 4.0.5 GA
- GlassFish V1 UR 1

A best practice when using the runtime servers is to choose one of the versions that has (CodeGear or Borland) after it. These versions have been extended to support specific JBuilder 2007 features.

The following web servers and/or containers are supported, but must be purchased separately:

- Borland Application Server 6.6 or 6.7, with Tibco or OpenJMS
- IBM WebSphere 6.0
- IBM WebSphere 6.1
- BEA WebLogic Application Server 8.1
- BEA WebLogic Application Server 9.2
- BEA WebLogic Application Server 10.0
- Oracle 10.1.3 Application Server
- Oracle Containers for Java (OC4J) 10.1.3

JBuilder 2007 helps you develop web services quickly. You can develop bottom-up web services by exporting a Java class or bean to a web service. Once you open your Java class as a web service, the service is immediately runnable.

Web Services Standards

The standards on which web services development is based are evolving technologies. The primary standards include:

- EJB (Enterprise JavaBean Technology))
- SOAP (Simple Object Access Protocol)
- WSDL (Web Services Description Language)
- UDDI (Universal Description, Discovery and Integration)
- WSIL (Web Services Inspection Language)
- JAX-RPC (Java API for XML-based Remote Procedure Call)
- WS-I (Web Services Interoperability)
- SAAJ (SOAP with Attachments API for Java)

See the Eclipse Web Application Development Guide for information on web services standards.
Web Service Properties

When the Web Services Explorer is open, you can set service and WSDL, WSIL, and UDDI properties. Service properties control the web service; WSDL, WSIL, and UDDI properties control the connection to the web service. Defaults are provided for the targeted server runtime and toolkit so that the service is immediately runnable.

Building and Running Web Services

When you export a Java class to the Web Services Explorer, JBuilder 2007 builds the dynamic web project hosting the web service or web services. Most of the web services artifacts are regenerated. Files in the /GeneratedSource/ folder are deleted and recreated. They are read-only.

You cannot edit the JUnit test case in the /GeneratedSource/ folder. Instead, you can edit the JUnit subclass in the /src/ folder. This file will not be overwritten at build time.

Before you can run the web service, you have to configure the runtime for your web server or container.

When you run the web service, the run configuration for the target web or application server is used. To view the run configuration, open the Run dialog box (Run ➔ Run). The run configurations for the server are in the Web Service node. The configurations for the client are in the Web Client node.

When you run the web service, the WSDL, WSIL, or UDDI file is validated. If the file is not valid, the web service won’t run.

JBuilder 2007 Web Services Tools

End-to-end web services generation in JBuilder 2007 uses both existing WTP features, as well as JBuilder 2007-only features like the Web Services Explorer. Much of the work, such as file generation, is done behind-the-scenes. JBuilder 2007 provides:

- Web Services Explorer— A design surface for visually creating and implementing web services.
- Properties pane – A pane for setting service and WSDL properties.
- Add Web Service From URL wizard – A wizard for creating a web service from a WSDL located at a URL.
- Convert into Web Services Client Project wizard – A wizard to convert an existing Java project with a WSDL into a web services client project.

Apache Axis Toolkit

JBuilder 2007 supports the Apache Axis web services toolkit.

The Apache Axis toolkit is an open source implementation of Simple Object Access Protocol (SOAP), an XML-based protocol for exchanging information. When you target your project for Apache Axis, the appropriate web services files, including a WSDL document, are generated for you. Methods of the selected class are exposed as a web service. Server-side classes, including as an implementation for the server, are created automatically.

Related Concepts

- Apache Axis Toolkit
- Web Services Explorer Overview

Related Tasks

- Designing a Bottom-Up Web Service Using the Apache Axis Runtime
- Designing a Top-Down Web Service Using the Apache Axis Runtime
- Working in the Web Services Explorer
Web Services Explorer Overview

The Web Services Explorer provides a design surface for quickly creating, implementing, and validating web services. Once you open the Web Services Explorer, the service representation for the exported JavaBean is displayed. In the Web Services Explorer, you set service and WSDL options with inspectors.

All activity in the Web Services Explorer is persisted on disk using a Web Services Deployment Unit (WSDU) file, named `axis.wsdu`. A toolkit-specific build file, named `build_axis.xml`, is created in the same folder. Any activity in the Web Services Explorer is then entered as an Ant task in this build file. The Ant build file invokes appropriate build tasks and other build functions.

Service Representation

When you export a JavaBean to a web service, a service representation is created in the Web Services Explorer. The representation contains fields. Each field has an associated inspector. For example, when you design a Java web service, the service representation contains Service, Methods, Server, and Settings fields.

To access an inspector, click a field in the service representation. Values in the inspectors are automatically filled in for you when you export a Java class to a web service. The default toolkit values are also filled in for you, although you can change them.

**Note:** When the top field of a service is checked, web services generation is enabled. If it is unchecked, the service is disabled, the inspectors are not available, and the service is not built.

You use the inspectors to set service and WSDL options. Changes are applied to the WSDL file at the next build. Service options include:

- Service style
- Use
- Type mapping version
- SOAP action
- Location URL
- Service name
- Binding name
- Port type name
- Deploy scope

WSDL options include:

- Include WSDL file
- Import schema
- Target namespace
- Output/Interface WSDL file
- Location import URL
- Implementation WSDL file
- Implementation namespace
Web Services Explorer UI

After you have created a dynamic web project and exported a JavaBean to a web service, a Web Services Explorer node appears in the Project Explorer. An Axis modules is displayed as a child node of the Web Services Explorer node.

Web services are displayed as service representations in the Web Services Explorer. The service representation contains fields, such as Server, Service, and WSDL. Each of these fields has an associated inspector for setting service options. Click a field to open its inspector.

The Web Services Explorer has a toolbar for creating and deleting services and for viewing source code. The context menu contains commands for viewing the implementation source.

Related Concepts

Web Services Overview

Related Tasks

Working in the Web Services Explorer
Apache Axis Toolkit

Apache Axis is an open source implementation of Simple Object Access Protocol (SOAP), an XML-based protocol for exchanging information. Axis uses the Simple API for XML (SAX) instead of the Document Object Model (DOM). It is a modular, flexible, and a high performance implementation of SOAP. The Apache Axis toolkit is JAX-RPC (Java API for XML-based Remote Procedure Call) compliant and supports WSDL 1.1. It also provides document/literal support for the WS-I Basic Profile 1.0 and JAX-RPC 1.1 specifications.

When you target your web service for Apache Axis, the toolkit generates the required web services files, including a WSDL document. Methods are exposed as a web service. Server-side classes are created automatically. You can choose to create a client project to test your web service.

Exporting a Web Service

The Axis toolkit supports exporting Java classes and stateless session beans as web services. When working in the Web Services Explorer on an Axis web service, you can set properties for the service. When you export a Java class or EJB, a WSDL is automatically generated according to the options you set. The WSDL is an XML file that describes the service.

Note: The WSDL is automatically added to the root of the /WebContent/ folder in your project.

When you export a class to a web service, the dynamic web project is enabled for Axis. The following occurs:

- The AXIS JAR files are copied to the /WebContent/WEB-INF/lib folder.
- Entries are added to the web.xml file for the Axis servlets.
- The Java2WSDL builder is added to the project.
- The WSDD builder is added to the project.
- The server-side Axis informational pages are added to the /WebContent/ folder.
- A run configuration is created to start the web or application server.

Importing a Web Service

You can import a WSDL from a URL with the Add Web Service from URL wizard (File ➤ New ➤ Other ➤ Web Services ➤ Web Service Client from URL). The Axis toolkit generates client-side classes for consuming a service and a JUnit test case to test interaction with the web service. The files generated by the toolkit, which are saved in a package name based on the WSDL target namespace or one that you specify, are dependent upon the properties you set for the service in the Web Services Explorer.

Building a Web Service

If Project ➤ Build Automatically is enabled, the web service is built automatically when you modify the web service in the Web Services Explorer. If this option is off, you need to build the project with build commands.

As you work in the Web Services Explorer, an Ant build file, build-wsdl2java.xml is generated and written to the /WebContent/ folder in your project. The appropriate Ant tasks are generated and saved to this Ant build file. At build time, the Ant build file is passed to the toolkit to generate the appropriate web services files.

The generated source is read-only and should not be modified, with these exceptions:

- The client-side JUnit test case subclass is not overwritten when you regenerate web services. Any modifications you make to this file won't be overwritten. This file is in the /src/ folder of the Client project.
Bean types are not overwritten when you regenerate web services.

**Axis Java2WSDL Parameters**

The Axis java2wsdl Ant task generates a WSDL from Java classes. Java classes form a WSDL. Mappings from namespaces to packages are provided as nested mapping elements. These parameters are the same settings that you make in the Web Services Explorer. This task doesn't do any dependency checking.

**Axis WSDL2Java Parameters**

The Axis wsdl2java Ant task generates Java classes from a WSDL. Mappings from namespaces to packages are provided as nested mapping elements. These parameters are the same settings that you make in the Web Services Explorer. This task doesn't do any dependency checking.

**Web Services Files**

When you use the Web Services Explorer to create web services, files are generated by the Axis toolkit. These web can include server-side classes, deployment files for the server, and a WSDL document to describe an exported service. At build time, these files are passed to the Apache Axis toolkit to generate the appropriate web services files and deployment information. The web services files generated by the Axis toolkit are accessible in the `/WebContent/` folder for the server and the `/WebContent/, /Generated_Source/` and `/src/` folders for the client.

**Web Services Explorer Files**

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>build_java2wsdl.xml</td>
<td>Server- and client-side. The Ant build file for a bottom-up web service.</td>
</tr>
<tr>
<td>build_wsdl2java..xml</td>
<td>Server- and client-side. The Ant build file for a top-down web service.</td>
</tr>
<tr>
<td>server-config.wsdd</td>
<td>Server-side. XML file created by the Axis toolkit at build time. Provides deployment information to the server.</td>
</tr>
</tbody>
</table>

**Client-side Files**

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;service name&gt;Proxy.java</td>
<td>The client proxy. (Client <code>/Generated_Source/</code> folder)</td>
</tr>
<tr>
<td>&lt;service name&gt;Service.java</td>
<td>A service interface that defines a get method for each port listed in the service element of the WSDL. This service interface defines a factory class to get a stub instance. (Client <code>/Generated_Source/</code> folder)</td>
</tr>
<tr>
<td>&lt;service name&gt;ServiceLocator.java</td>
<td>A locator class that is the client-side implementation of the service interface. (Client <code>/Generated_Source/</code> folder)</td>
</tr>
<tr>
<td>&lt;service name&gt;TestCase.java</td>
<td>A JUnit test case for testing the web service. (Client <code>/Generated_Source/</code> folder)</td>
</tr>
<tr>
<td>&lt;service name&gt;TestCaseImpl.java</td>
<td>A subclass of the JUnit test case. Add tests to this subclass instead of the generated JUnit test case class. (Client <code>/Generated_Source/</code> folder)</td>
</tr>
<tr>
<td>&lt;PortType name&gt;.java</td>
<td>An interface for each portType in the WSDL. The implementation of this interfaces calls remote methods. (Client <code>/Generated_Source/</code> folder)</td>
</tr>
<tr>
<td>&lt;binding name&gt;Stub.java</td>
<td>A client-side stub class that acts as a proxy for a remote web service. Allows you to call the web service as if it were a local object. This class implements the <code>&lt;PortType name&gt;.java</code> interface. (Client <code>/Generated_Source/</code> folder)</td>
</tr>
<tr>
<td>data types</td>
<td>Java files for all other types. Holders needed for the web service. (Client <code>/Generated_Source/</code> folder)</td>
</tr>
</tbody>
</table>
Related Concepts

Web Services Overview

Related Tasks

Working in the Web Services Explorer
Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Designing a Top-Down Web Service Using the Apache Axis Runtime
Developing Web Applications

The JBuilder 2007 web development environment provides the tools needed to develop simple (consisting of only static Web pages) or more advanced dynamic web application based on the Java EE specification. JBuilder 2007.

In This Section

- [Web Applications Overview](#)
  This topic describes web application development in JBuilder 2007.
Web Applications Overview

Static or detailed dynamic web applications are quickly developed with JBuilder 2007. Static web projects include images, HTML files and cascading style sheets. Dynamic web projects contain dynamic Java EE resources. Web application are deployed within a web project to the server in the form of a Web archive (WAR) file. The web application is viewed as a web site from a web browser.

Static Web Applications

For web applications that require only basic content use the static Web project type. Static web applications can be converted to dynamic Web projects.

Dynamic Web Applications

Dynamic web projects may include Java Server Pages and servlets and are based on the Java EE model which defines a Web application directory structure.

Related Concepts
  - Runtime Servers

Related Tasks
  - Creating a Web Application Project
  - Setting Up a Runtime Server
  - Publishing a Java EE Application to a Server Runtime
  - Running an Application on a Runtime Server

Related Reference
  - Eclipse help topic "Server targeting for Web applications"
  - Eclipse help topic "Web Projects"
  - Eclipse help topic "Web archive (WAR) files"
  - Eclipse help topic "Server targeting for web applications"
  - Eclipse help topic "Web Projects"
  - Eclipse help topic "Creating a static web project"
  - Eclipse help topic "Dynamic web projects and applications"
  - Eclipse help topic "Web page design"
Developing Modeling Applications

Modeling applications are developed in the Together visual editor.

In This Section
- Modeling Applications Overview
  - Describes modeling applications
Modeling Applications Overview

This section provides information on modeling applications. Modeling provides a visual approach to Java programming. The modeling perspective gives you an overview of your programming projects. In the model, you can browse class relationships and explore different aspects of your project. In the Modeling Perspective, you can select programming objects from the palette and drop them into your project. You can switch between the source code and model views.

JBuilder 2007 provides round-trip integration between the source code and modeling views. When you change the model, the source code changes to match. The model also reflects changes to the source code.

JBuilder 2007 provides modeling support for Java, Enterprise Java Bean (EJB) and Java Persistence API (JPA) projects. JBuilder 2007 includes Borland's Together modeling framework, as well as some of the other Eclipse modeling environments. JBuilder 2007 can import a model from an XML schema, an Xdoclet-annotated WTP project, a Java project, or an EJB project.

JBuilder 2007 also includes the InterBase and JDataStore database systems designed to develop and test database applications. You can create Java database applications using the Modeling Perspective, which includes the Together modeling framework. Together allows dragging and dropping of components to develop the database using a visual editor.

Modeling Perspective

The modeling perspective provides a graphical view of a Java project. In the modeling perspective, you can view packages, classes, interfaces, enumerations, Enterprise Java Beans (EJBs), and the links between them. You can make changes directly to the models and those changes are reflected in the underlying code.

Model Diagrams

You can view a model diagram of any portion of your project. The diagram shows the structure and relationships of the objects in your project.

Related Tasks

Creating a Modeling Project
Importing a Modeling Project

Related Reference

Together
Borland InterBase Home Page
Borland JDataStore Home Page
ProjectAssist and TeamInsight Concepts

This section describes the concepts of the ProjectAssist and TeamInsight installed developer tools. These tools are installed on (or assimilated through) a ProjectAssist server and development team members can access the tools to manage and coordinate their work through TeamInsight portlets (Bugzilla, Continuum/Maven, CVS, Liferay, StarTeam, Subversion, and XPlanner).

In This Section

ProjectAssist and TeamInsight Overview
Describes how to coordinate software development teamwork by installing the ProjectAssist server and then configuring the various TeamInsight components.

Liferay: The TeamInsight Project Portal
Describes the Liferay web portal supported by TeamInsight for the client.

Subversion: Source Code Repository
Describes source control as managed by the Subversion component of TeamInsight.

CVS: Source Code Repository
Describes source version control as managed by an assimilated CVS component of TeamInsight.

Continuum/Maven: Continuous Build System
Describes the continuous build system provided by the TeamInsight Continuum/Maven tools.

Mylar Concepts
Describes using Mylar task repositories for an assimilated StarTeam installation tasks and bug tracking.

Bugzilla: Defect Tracking System
Describes bug tracking as implemented by the Bugzilla component of TeamInsight.

StarTeam: Source Code Repository, Change Request Tracking, and Task Provider
Describes source control as managed by the Subversion component of TeamInsight.

XPlanner: Project and Team Management
Describes the concepts behind the design and use of XPlanner to track and manage development projects.
ProjectAssist and TeamInsight Overview

ProjectAssist and TeamInsight are JBuilder 2007 features that install and facilitate the use of a suite of development tools. The TeamInsight tools enhance the performance of your software development team. These tools help coordinate teamwork and thereby optimize your team's efforts. The tools are installed and configured on a ProjectAssist server by the ProjectAssist Administrator.

Only the JBuilder 2007 editions support the ProjectAssist installation features. The JBuilder 2007 Enterprise edition supports the ProjectAssist installation or assimilation features for the following software products: Version Control (CVS or Subversion), Build System (Continuum), Defect Tracking (Bugzilla) or Task Provider (XPlanner). The JBuilder 2007 BALM Edition supports the ProjectAssist installation features for all the Enterprise edition components and also supports the assimilation of an existing StarTeam installation (for Version Control, Defect Tracking or as a Task Provider).

As part of the ProjectAssist install, the Administrator defines projects and team members for the projects. The team members can then coordinate their efforts through the use of the various TeamInsight tools. The TeamInsight tool selection is determined by the JBuilder 2007 product edition (Enterprise or Enterprise BALM Edition for StarTeam options). The following tools can be available through a ProjectAssist installation or assimilation:

- Liferay to open the team's web portal, which summarizes the current status of the project and provides access to several TeamInsight components.
- CVS, StarTeam or Subversion for version control, which allows team members to check source files in and out, and to synchronize the repository files.
- Subversion Viewer (Sventon) to browse the Subversion source repository.
- Continuum/Maven to establish an automatic build environment linked with the repository and to monitor build and quality status.
- Bugzilla or StarTeam to record and track defects and change requests in the source code.
- StarTeam or XPlanner to monitor development progress by creating and tracking projects, iterations of projects, user stories, and individual tasks.
- Burn down chart and current iteration details from XPlanner.

Defining Users and User Roles

As part of the ProjectAssist server install and configuration, the ProjectAssist Administrator defines projects and adds users for those projects. For each ProjectAssist component, the ProjectAssist Administrator categorizes users into one of two user roles (administrator or developer) or no access for that component. When adding a user, all components default to the Developer role for all users. The exception is the shared MySQL component, which defaults to No Access for everyone but the ProjectAssist Administrator.

**Administrator**
If you are the ProjectAssist server Administrator, you install ProjectAssist on a server or assimilate existing project components over single/multiple machines. The ProjectAssist Administrator adds projects, users, assigns user access rights, and sends user notifications regarding the server install and the TeamInsight client installation. The ProjectAssist Administrator sample default administrator, Joe Bloggs, is assigned Administrator privileges for all TeamInsight components by default. He can be cloned to establish a Administrator identity for a new user. As part of the server configuration of users, the ProjectAssist Administrator assigns the user unique roles for each TeamInsight tool by right-clicking on the user name. For example, a user can be assigned XPlanner administrative privileges yet retain Developer privileges on other TeamInsight tools.

**Developer**
Users are granted access to repositories based on assigned project rights. Users receive access to the ProjectAssist components from the ProjectAssist server Administrator. Configuration information is downloaded from either the attachment to the ProjectAssist user notification message or from the Liferay portal.
For each TeamInsight tool, a user can be assigned a No Access role by the ProjectAssist Administrator. No access means the user has no read, write nor execute access to the component. The ProjectAssist Administrator can assign a No Access role by right clicking on the user name.

**Liferay Portal Summarizing Project Status**

The Liferay portal provides reports from several TeamInsight tools, including details about the Subversion repository, progress in the current XPlanner project, bug tracking status, project build status, and QA Lab summary information.

The Liferay portal contains the following portlets:

- Subversion repository status (JBoss Labs Kosmos Subversion)
- CVS repository information for project repositories
- Current iteration details (XPlanner)
- Burn down chart and current iteration details from XPlanner
- Build status (Continuum)
- Bug/defect status (Bugzilla)
- QALab summary and QALab classes (QALab)
- StarTeam Task, Change Requests, and/or StarTeam version control repository information

To match your project needs, the Liferay portal can be customized to display other portlets.

**Subversion, CVS, or StarTeam Repository for Version Control**

TeamInsight can include an assimilated CVS repository or a local or remote Subversion repository for version control in the JBuilder 2007 Enterprise edition. In the JBuilder 2007 Enterprise BALM edition, the TeamInsight tools can also include an assimilated StarTeam installation.

Version control systems manages project files and directories in the repository over time. Team members can simultaneously check out the projects stored in a shared repository. When team members check in their changes, the version control system synchronizes the repository using an edit-update-commit paradigm and manages branching in the source repository.

If Subversion is chosen as the version control system, TeamInsight gives each team member access to the Subversion Viewer, a read-only browser for Subversion repositories (Sventon). Team members can access the Subversion Viewer through the TeamInsight Viewer. The Subversion Viewer enables users to browse, download, view logs and locks, and diff the files in the repository.

If CVS is assimilated as the version control system, TeamInsight can display a CVS web view of the repositories. The CVS web view URL is specified through the Stacks tab in the ProjectAssist Designer. The designated URL should be a pre-installed web view of the repository. Team members can access the specified CVS web view URL through a tab at the bottom of the TeamInsight Viewer.

**Continuum/Maven Continuous Build System**

Continuum is a continuous integration build system that builds and tests code on a regular basis. Continuum monitors the source repository on a specified schedule and triggers a build if any changes have been made in the repository. As developers check in code throughout the day, builds are triggered. Many builds can occur daily. A build can be defined as anything from testing and compiling a single project to the assembly and testing of a deliverable from multiple projects. Continuum allows for rapid turnaround of integrated builds, thus supporting quicker identification of critical build issues.

The Maven tool can build and manage Java-based projects. Maven is a software project management and comprehension tool that is based on the concept of a project object model (POM). Maven allows a project to build
using the POM and a set of plugins that are shared by all projects using Maven, providing a uniform build system. Maven can manage a project's build, reporting and documentation from this central piece of information. Maven creates a way to build the projects, a clear definition of project content, an easy way to publish project information, and a way to share Java Archives (JARs) across several projects. The result is a tool that can be used for building and managing Java-based projects.

**Bugzilla for Bug Tracking**

Bugzilla is a bug tracking system that allows development teams to report, track and repair defects in their products while remaining team connected. It is a powerful tool that allows effective team organization and communication. Bugzilla also expands the use of the term “bug” to track other things such as feature requests so it can also be used for some general project tracking.

**Borland ALM StarTeam for Change Requests, Tasks or Version Control**

The JBuilder 2007 BALM Edition supports the ProjectAssist installation for the assimilation of an existing StarTeam installation. During the ProjectAssist server installation with JBuilder 2007 Enterprise BALM Edition, the user can choose to include StarTeam, or alternate installations or assimilations, for version control, change requests or tasks:

- Version Control System (CVS, StarTeam, or Subversion)
- Continuous Build system (Continuum)
- Defect or Change Request system (Bugzilla or StarTeam)
- Task Provider (StarTeam or XPlanner)

You can also open a StarTeam task or bug repository using Mylar, a task-focused interface integrated into the JBuilder 2007 product.

**XPlanner for Project Planning**

XPlanner is an open-source Web-based planning and monitoring tool designed for projects using eXtreme programming (XP) or agile project management. XPlanner provides:

- Project iterations (sprints)
- User stories (product feature requirements)
- User tasks (work assignments)
- Metrics for tracking progress

You can also open an XPlanner task repository using Mylar, a task-focused interface integrated into JBuilder 2007.

**Mylar for Bugzilla, StarTeam and XPlanner**

JBuilder 2007 enables you to include the Bugzilla or assimilated Borland ALM StarTeam repository bugs and StarTeam or XPlanner repository tasks in the Eclipse Task List view. You can then use Mylar to define queries against those repositories, such as a “My Bugs” query. The Mylar plugin automates task-focused user capabilities for Bugzilla, StarTeam and XPlanner. After you activate a task, Mylar remembers the context of your subsequent work, such as the files associated with the active task. Later when you return to the task, the preserved context enables you to work more efficiently.
Eclipse Wiki

As part of the team collaboration features of JBuilder 2007, a Wiki for team members is now available inside the IDE. A project Wiki can be a powerful project documentation tool for team members to coordinate efforts, disperse unified project information, and take full advantage of the open-source community. Editing Wiki files inside the IDE provides automatic page creation and dynamic linking (to other pages and Java source in the workspace). Checking the .wiki files into the version control system permits team members that checkout the project to view and edit the wiki. Wiki content can also be exported into a set of static web pages, including conversion of Java files into HTML.

The Eclipse Wiki plug-in is provided with JBuilder 2007. However, it must be enabled and configured according to documentation provided on the web or in the Eclipse Wiki Help files included with JBuilder 2007.

**Note:** Refer to the Eclipse Wiki Help files and the following “Related Reference” link for more information on enabling and configuring your Wiki plugin.

Related Concepts

- Liferay: The TeamInsight Project Portal
- Subversion: Source Code Repository
- CVS: Source Code Repository
- Bugzilla: Defect Tracking System
- Continuum/Maven: Continuous Build System
- StarTeam: Source Code Repository, Change Request Tracking, and Task Provider
- XPlanner: Project and Team Management

Related Reference

Eclipse Wiki Editor Plugin Web Information
Liferay: The TeamInsight Project Portal

The TeamInsight tools support a Liferay web portal accessible from client workstations. Team members typically access the Liferay portal to check project status.

The Liferay portal contains portlets or windows that display status reports from several of the TeamInsight tools. Several of the portlets also contain a link to the TeamInsight tool that generated the information on the portlet. For example, the XPlanner windows in the portal both contain a link to XPlanner.

Liferay Portal Gathers Reports From TeamInsight Tools

By default, the Liferay portal contains the following reports from the TeamInsight tools (if installed) and can be accessed through a tab on the TeamInsight Viewer:

- Status report for the Subversion repository, compiled by the Kosmos monitoring plugin
- CVS repository information for project repositories
- Burn down chart and current iteration details from XPlanner
- Build status from Continuum
- Bug tracking status from Bugzilla
- QALab Summary and QALab Classes
- StarTeam Task, Change Requests and/or StarTeam version control repository information

Default Contents of the Liferay Portal

The Liferay project portal contains status reports from the TeamInsight tools (if installed) as follows:

- **JBoss Labs Subversion**: Gives the location of project repositories and, for each repository, the current revision number, the total number of committers, activity in the last 7 days, and the age of the latest touch (change). Each field, except for the Latest touch age, contains a link to details or expanded information. For example, the link for the project repository field opens a pair of charts (Repository entry history and Files by file type). The link for the current revision number displays revision details (Most active files). This portlet's information comes from the Kosmos Subversion monitoring tool from JBoss Labs.

- **CVS Repository Information**: Displays data on the project repository, commit log, active files, developer details, and setup information.

- **Burn Down Chart**: Displays a graph representing the burn down rate (total remaining hours of work over time) for the current iteration. Tabs display similar XPlanner information from other iterations. Users can link directly to XPlanner.

- **Current Iteration Details**: Summarizes the hours completed on the stories in the current project iteration. Tabs display similar XPlanner information from other iterations. Users can link directly to XPlanner.

- **Build Status**: Displays a summary of the most recent project builds. Includes links to the Results for each build, and a link to Continuum. The **Project Health** link opens the project web page for Continuum.

- **Bugzilla Status**: Lists bugs reported for the project, as well as statistics about bug reports. The Bugzilla Status portlet has six tabs and a link to go to the Bugzilla bug management tool. The Important tab lists the most important bugs, while the Newest tab lists the most recently reported bugs. The Severity and Priority tabs display graphs. The Assignee tab displays a pie chart showing the relative number of bugs assigned to each team member. The Trends tab displays statistics about bugs over time.

- **QALab Summary and QALab Classes**: Summarizes statistics about recent QA results obtained from the Cobertura and PMD open-source plugins. QALab works with Continuum/Maven to compile results in a qalab.xml file for each run. QALab Classes lists all the classes that are responsible for the results.
- **StarTeam Repository Information**: Displays data on the StarTeam project repository.
- **StarTeam Tasks**: Lists open tasks for the project, as well as setup information, including Project Name, Login Name, StarTeam view, folder, host and endpoint data.
- **StarTeam Change Requests**: Lists change requests reported for the project, as well as assignee and setup information. The StarTeam Change Request Status portlet has three tabs. The Newest tab lists the most recently reported bugs. The Assignee tab lists bugs assigned to each team member. The Setup tab displays setup information, including Project Name, Login Name, StarTeam view, folder, host and endpoint data.

**Related Concepts**
- XPlanner: Project and Team Management
- Subversion: Source Code Repository
- CVS: Source Code Repository
- Bugzilla: Defect Tracking System
- StarTeam: Source Code Repository, Change Request Tracking, and Task Provider

**Related Tasks**
- Administering the Liferay Portal
- Opening the TeamInsight Viewer and the Liferay Portal
- Changing Your Passwords for the TeamInsight Tools
- Adding Mylar Repositories for Bugzilla and XPlanner

**Related Reference**
- The Bugzilla Guide
- QALab Introduction
- JBoss Kosmos
- What is Cobertura?
- PMD
Subversion: Source Code Repository

Subversion (as Subclipse) is the source code version control manager. TeamInsight also provides a read-only browser for the Subversion repository.

The Subversion Repository Resides on the Server

During installation by ProjectAssist, the Administrator creates the Subversion repository for the development project. All team members can access the source code repository simultaneously. Subversion maintains full copies of all previous versions of the project.

The top level in the project directory typically contains subdirectories named trunk, branches, and tags. Subversion manages branching in the repository.

Team Members Use Edit-Update-Commit Work Pattern

Subversion, as Subclipse, is the version control tool in Eclipse and JBuilder 2007. Subversion allows users to check out copies of the same files and projects. Team members use an Edit-Update-Commit work pattern.

To check out a project, team members click Project ➤ Checkout Project. Checking out a project creates a private local copy for the team member. After making changes to the source, the team member uses the right-click menu command Team ➤ Update to synchronize the private copies with the repository. Finally, the team member checks in the files to the repository using Team ➤ Commit, and Subversion synchronizes the repository.

Tip: For online help on Subclipse, click Help ➤ Help Contents ➤ Subclipse.

Subversion Viewer Provides Read-Only Access to the Repository

TeamInsight provides an open-source, read-only browser for Subversion repositories. You access the Subversion Viewer through the Liferay web portal.

The Subversion Viewer enables you to:

- Browse and download files for specific revisions
- Search the current directory and below, including CamelCaseSearch
- View the log of changes
- View the current file locks
- Diff files between revisions or directories
- View the directory flattened into one level

For easy viewing, the viewer highlights source code using JHighlight. You can also customize the style sheets that the viewer uses.
Related Concepts
  Continuum/Maven: Continuous Build System

Related Tasks
  Using the Subversion Viewer for Browsing the Project Repository
  Checking Out a Project, Making Changes, and Checking Your Changes Into the Repository

Related Reference
  Subclipse Online Help
CVS: Source Code Repository

CVS (Concurrent Versions System) is an open-source code version control manager. An existing CVS installation can be assimilated during the ProjectAssist server installation of the TeamInsight components.

CVS permits team members access to the source code repository simultaneously. CVS maintains copies of all previous versions of a project.

Assimilation of an existing CVS installation into the ProjectAssist component structure allows:

- addition of Mavenized projects to existing CVS repository.
- integration with Continuum for continuous builds and metrics (with a cvs.exe, or equivalent executable file, on the path of the machine running Continuum.
- accessibility to users of new projects added via the TeamInsight .ticx file.
- integration of CVS repository statistics in TeamInsight Viewer portal (if CVS repository history is enabled).

The CVS Repository Resides on a Server

During installation by ProjectAssist, the Administrator can assimilate an existing CVS server repository for the project. The ProjectAssist Administrator must have a user account on the assimilated CVS with read and write access in order to add projects. The ProjectAssist Administrator cannot add or modify users on the CVS installation. Users should have their own CVS for repository access.

Using an assimilated CVS installation also requires a secure CVS PServer port for anonymous CVS access.

Team Members Use Edit-Update-Commit Work Pattern

CVS is one of the version control options available in Eclipse and JBuilder 2007. CVS allows users to check out copies of the same files and projects. Team members use an Edit-Update-Commit work pattern.

Individuals without CVS accounts may be able to check out projects by pulling them anonymously but they cannot check in any changes. (Refer to the following Importing the TeamInsight .ticx file subtopic.

To check out a project, team members click Project ▶ Checkout Project. Checking out a project creates a private local copy for the team member. After making changes to the source, the team member uses the right-click menu command Team ▶ Update to synchronize the private copies with the CVS server repository. Finally, the team member checks in the files to the repository using Team ▶ Commit, and CVS synchronizes the repository.

CVS Viewer Specified by URL

The TeamInsight Viewer allows you to specify a URL of a CVS viewer. Specify a URL for a pre-installed web view of the CVS repository. The URL is used as the CVS view in the TeamInsight viewer. The URL address may be left blank and no CVS view will be available through the TeamInsight Viewer.

Importing the TeamInsight .ticx file

Assimilating an existing CVS installation into the JBuilder 2007 ProjectAssist component stack and Importing the TeamInsight .ticx file to the team members local machine adds:

- TeamInsight viewer portal if there was a URL specified in the component configuration
- Project check out from CVS repository. Because user names are embedded in PSERVER locations, an attempt is made to match an existing CVS repository location (as seen in the CVS Repository Exploring perspective).
Otherwise, if the project is checked out as anonymous, the user cannot check in any changes. Locations are matched by host in the following manner:

- **No Matching Locations:** The user is prompted for a user name and password. The user should enter existing CVS credentials to check out with read/write access (if this is the access of the user's existing credentials). Otherwise, a user may pull files as an anonymous user. In this case, a new CVS location is created.
- **Non-anonymous Match:** The user's existing name is used. A password prompt is issued if that location does not have its password saved, and has not been accessed during the current session.
- **Anonymous Match:** If there is only an anonymous match, this match is used.

You can change the user name of any CVS location saved by the IDE, and any projects that used that connection are changed to match. This allows user checkout as anonymous, and then a name change of the CVS location so the user can check in changes later.

For any project, under the project's CVS properties, you can "Change Sharing" to any compatible location at any time.

**Related Concepts**
- [Continuum/Maven: Continuous Build System](#)

**Related Tasks**
- [Checking Out a Project, Making Changes, and Checking Your Changes Into the Repository](#)

**Related Reference**
- [CVS Online Documentation](#)
- [Installing CVS Secure PServer for Anonymous CVS Access](#)
**StarTeam: Source Code Repository, Change Request Tracking, and Task Provider**

(JBuilder 2007 Enterprise BALM Edition) During installation of the ProjectAssist server and TeamInsight components, Borland's ALM **StarTeam** product can be assimilated as the source code version control manager, change request tracker, and/or task provider system.

Borland® StarTeam® is a configuration management tool, designed for coordinating and managing the entire software delivery process. StarTeam promotes team communication and collaboration through centralized control of project activities. It provides integrated requirements management, change and configuration management, project and task management, defect (referred to as change requests in StarTeam terms), and file versioning. The assimilation of a StarTeam installation into your TeamInsight component stack adds to the team-centric features of TeamInsight and ProjectAssist by further unifying teams within the centralized environment and allowing you to leverage your current StarTeam software.

**StarTeam Assimilation through ProjectAssist**

During installation by ProjectAssist, the Administrator can assimilate an existing StarTeam installation into the ProjectAssist component stack for use by all TeamInsight members.

The ProjectAssist Administrator selects the stack components to install during the ProjectAssist installation through the **New ProjectAssist File: Select Stack Components** dialog. With the JBuilder 2007 Enterprise BALM Edition, the TeamInsight tool choices are:

- Version Control System (CVS, StarTeam, or Subversion)
- Defect or Change Request Tracking system (Bugzilla or StarTeam)
- Continuous Build system (Continuum)
- Task Provider (StarTeam or XPlanner)

StarTeam or CVS assimilation through ProjectAssist requires pre-existing StarTeam or CVS servers.

**Note:** The StarTeam Cross Platform Client needs to be installed on the machine where the Continuum installation resides in order for Continuum builds to work with StarTeam.

**Mylarized Views for StarTeam Tasks and Change Request Tracking**

With assimilation of an existing StarTeam installation, TeamInsight tools users can add a Mylar Connector for StarTeam tasks and change requests. TeamInsight members should go to the **Window ➤ Configure Mylar ➤ projectname** menu path to create a Connector for StarTeam Change Requests and StarTeam Tasks, and to see queries of their StarTeam Change Requests and Tasks.
Related Concepts

- ProjectAssist and TeamInsight Overview
- Liferay: The TeamInsight Project Portal
- Mylar Concepts
- Continuum/Maven: Continuous Build System

Related Tasks

- Installing the ProjectAssist Components File for JBuilder 2007 Enterprise Borland ALM Edition
- Adding Mylar Repositories for StarTeam Change Requests or Task Planning
- Checking Out a Project, Making Changes, and Checking Your Changes Into the Repository

Related Reference

- StarTeam Product Page
Continuum/Maven: Continuous Build System

ProjectAssist provides continuous integration of source code. Continuum, a continuous integration tool, builds and tests code on a regular basis. The continuous integration system monitors a source control repository (such as Subversion, CVS, or StarTeam) at regular intervals and can trigger a build when changes are made to the repository. A build can include anything from compiling and testing a single project to assembling and testing of a deliverable from multiple dependent projects. The Continuum tool ensures that a project build succeeds at any point in the development cycle by allowing immediate identification of defects.

The Continuum component of the ProjectAssist install works with Maven 2 projects. A Maven 2 project typically consists of multiple modules. Each module has its own pom.xml file. ProjectAssist determines whether a project is a Maven 2 project by detecting the existence of a pom.xml file in the root of the project. Maven projects can also be quickly created using the Maven archetype wizard.

Related Concepts
- ProjectAssist and TeamInsight Overview
- Subversion: Source Code Repository

Related Tasks
- Using the Subversion Viewer for Browsing the Project Repository
- Using Continuum/Maven for Continuous Integration Builds
- Checking Out a Project, Making Changes, and Checking Your Changes Into the Repository

Related Reference
- Maven Project from Archetype
- Continuum Online Resources and Documents
- Maven Online Resources and Documents
Mylar Concepts

Mylar is a task-focused user interface available with Eclipse. Mylar makes multi-tasking easier thus reducing information overload. Task and bug information is integrated into repositories that offer offline editing for ease of use and increased productivity.

Once tasks and bugs are integrated into the Mylar view, Mylar monitors work activity to identify information relevant to the tasks or bugs. Mylar uses this context to filter information, providing only the related and useful information in the user interface. The information you need to do your job efficiently is right at your fingertips through the Mylar view of the Task List.

The Mylar view of the Task List allows the definition of queries against task or bug repositories that use a Mylar connector. Mylar provides task-focused user capabilities for JIRA, Bugzilla, Trac, and generic Web repository. JBuilder 2007 adds Mylar support for StarTeam and XPlanner, in addition to the generic Mylar plug-ins. After you activate a task, Mylar remembers the context of your subsequent work, such as the files associated with the active task. Later when you return to the task, the preserved context enables you to work more efficiently.

JBuilder 2007 enables you to automatically add Bugzilla and XPlanner repositories (and also your own bug/task queries) to the Eclipse Task List view with Mylar. With assimilation of an existing StarTeam installation, you can add a Mylar repository to the Task List view for StarTeam tasks and bugs, and define queries against those tasks and bugs.

Using Mylar, you can:

- Connect to task- or bug-tracking repository
- Define a query against the repository so that bugs or tasks are represented as Mylar tasks in the development environment
- Define tasks related to the repository
- View task or bug reports locally or in an embedded browser
- Activate tasks and focus on the active task
- Save task context, including files and file hierarchy
- Work with tasks offline and synchronizes with the repository at a later time

Related Concepts

- ProjectAssist and TeamInsight Overview
- Bugzilla: Defect Tracking System
- StarTeam: Source Code Repository, Change Request Tracking, and Task Provider
- XPlanner: Project and Team Management

Related Tasks

- Configuring Your TeamInsight Client
- Adding Mylar Repositories for Bugzilla and XPlanner
- Adding Mylar Repositories for StarTeam Change Requests or Task Planning

Related Reference

- External Documentation for Mylar from Eclipse.org
- External Documentation about Mylar Connectors to Repositories
- External Article: Task-Focused Programming with Mylar
Bugzilla: Defect Tracking System

Bugzilla is a bug tracking system that allows development teams to report, track and repair defects in their products while remaining team connected. It is a powerful tool that allows effective team organization and communication. Bugzilla also expands the use of the term “bug” to track other things such as feature requests so it can also be used for some general project tracking.

Bugzilla allows team members to:

- Track bugs and code changes
- Communicate easily with teammates regarding defects and bug status
- Submit and review patches to the product code
- Manage quality assurance (QA) in a coordinated fashion

Bugzilla can be opened in either the Bugzilla TeamInsight Viewer or through a web browser. Bugzilla Status, a portlet on the Liferay project portal, lists several categories of bugs (important, severity, and so forth) and includes a link to the Bugzilla server component.

The Bugzilla Repository Resides on the Server

During ProjectAssist installation and definition of TeamInsight projects, the Administrator creates the Bugzilla repository for the development project. All team members can access the bug tracking repository simultaneously when they have installed the TeamInsight client on their machines.

Bugzilla is web-based and relies on an installed web server (Apache) and a database (MySQL). These required services are also installed during the ProjectAssist server install.

Note: Initially, all defined users are assigned the same password by the ProjectAssist Administrator. Each user must login to Bugzilla and change this initial password to a more secure password.

Bugzilla and Mylar

JBuilder 2007 enables you to add the Bugzilla repository bugs a to the Eclipse Task List view, and to use Mylar to define queries against those bugs. Refer to the Mylar concepts and tasks for more information.

Related Concepts

- ProjectAssist and TeamInsight Overview
- Liferay: The TeamInsight Project Portal
- Bugzilla: Defect Tracking System
- Mylar Concepts

Related Tasks

- Adding Mylar Repositories for Bugzilla and XPlanner
- Logging in to TeamInsight Bugzilla
- Creating or Generating Bug Reports in Bugzilla
- Querying Bugzilla for Bug Reports
- Managing Bug Reports in Bugzilla

Related Reference

- Bugzilla Resources and Documents
- The Bugzilla Guide
XPlanner: Project and Team Management

XPlanner is an open-source Web-based planning and monitoring tool designed for projects that use eXtreme programming (XP) or agile project management.

Agile Projects Work Well in XPlanner

The organization of XPlanner is similar to the organization of agile projects. XPlanner contains pages for Projects, Iterations, Stories, and Tasks.

In agile project management, the development cycle is divided into short iterations or sprints. During each iteration, team members complete a coherent increment of the final project. Each iteration is typically one to four weeks long. The team plans each product feature by creating one or more user stories that describe the feature. Within each story, the team then creates tasks that describe the work required to complete the feature.

Team members commit to completing a specific set of tasks, which are defined in each iteration. Team members meet on a frequent, often daily, basis in a scrum meeting to report progress they've made and to discuss obstacles blocking their way. Team leaders, known as scrum masters, aim to ensure that the team succeeds in completing the assigned tasks.

Projects Contain Iterations

The Top page in XPlanner lists the projects defined in XPlanner, their selectable IDs, and the name of the initial iteration in the project. Only the Administrator for ProjectAssist can create projects that are linked using the TeamInsight tools.

On a Project page in XPlanner, you see a list of the iterations defined in the project, along with their scheduled start and stop dates, and a list of the stories defined for each iteration.

Iterations Contain User Stories

For each project in XPlanner, you can create any number of iterations or sprints.

Iterations or sprints are coherent increments of the total work in a project. Iterations can include backlog, future plans, current sprints, and past or future sprints. Each sprint lasts a scheduled amount of time in which the team completes tasks that have been assigned to them in XPlanner. You can define an iteration without starting it and then return to start the iteration later.

On the Iteration page in XPlanner, you see a list of the defined user stories associated with that iteration.

User Stories Contain Tasks

The team divides their work on a project work into user stories (descriptions of features in the final product) and tasks (the steps required to create a feature). On the Story page in XPlanner, you see a list of the tasks defined for that story.

User Stories Describe Project Features

For each planned feature, the team creates one or more user stories in XPlanner.
**Tasks Describe the Steps in Creating the Feature**

For each user story, the team creates one or more tasks in XPlanner and estimates the time required to complete the task. During each iteration, the team tracks the time they spend on each task and marks tasks they have completed.

**Tree Structure Aids Navigation in XPlanner**

The tree structure located at the top of XPlanner windows displays your location in the XPlanner database. For example, the Top page does not display a tree structure at all, because you have not opened the database yet. However, after you open a project, an iteration, and a story, the tree structure on the Story page displays the names of the project, iteration, and story.

**Related Concepts**

- ProjectAssist and TeamInsight Overview
- Mylar Concepts

**Related Tasks**

- Adding Mylar Repositories for Bugzilla and XPlanner
- Adding Team Members in XPlanner (Administrator Task)
- Creating and Starting Project Iterations in XPlanner (Administrator Task)
- Planning a Product Feature: Creating a User Story in XPlanner
- Planning Your Work: Creating Tasks in XPlanner
- Tracking Your Time and Completing Tasks in XPlanner
- Moving or Continuing a Story or Task in XPlanner
- Monitoring Iteration Metrics in XPlanner

**Related Reference**

- XPlanner Documentation Available from XPlanner.org
Working with Peers

The peer to peer feature in JBuilder 2007 allows you to chat with peers and share data. You can share projects through a repository. You can also set up contact groups to effectively collaborate with a group of peers.

In This Section

Peer to Peer Collaboration
Peer to Peer Collaboration

Peer to peer collaboration features allow two or more users to collaborate across a local area network (LAN) and send data. Peers are discovered automatically when they are on the same LAN. You can collaborate with peers who are using JBuilder 2007, as well as with peers using JBuilder 2006.

You enable the peer to peer subsystem on the Peer to Peer page of the Preferences dialog box (Windows ► Preferences ► Peer to Peer).

You open the Peers view through the menu path (Window ► Show View ► Other ► Peer to Peer ► Peers). The Peers View shows you peers who are currently online. You can change your user status in the Peers view by clicking on your name and using the dropdown status menu. Peers are displayed in the Peers pane, on the left side of the Peers view.

Chatting with Peers

You use the peer to peer feature to chat with peers on your LAN. You chat with peers in the , on the right side of the Peers pane. A record of the chat, the chat log, is maintained and written to a file. The default location for the chat log file is set on the Peer to Peer page of the Preferences dialog box. One chat log, with all sessions recorded, is maintained for each peer. You can view or delete the log at any time.

Note: The messages are recorded in the chat log of each individual member of the collaboration session.

The Collaboration pane displays the running chat, as well as links to files, stack traces, web pages, diagrams, or version control system (VCS) links that have been sent to you by a peer.

Open the chat log in the Peers View by right-clicking and selecting the View Chat Log context menu, The chat log is UTF-8 encoded. If a peer is using an international locale and fonts, and you are using the US locale and fonts, the peer may not display correctly in the Peers pane. However, this information is still saved in the log file. The filename for the chat log will be corrupted as well, since the peer name is part of the filename. If you view the chat log file on a machine with the appropriate fonts, the filename and contents of the file will display correctly.

Collaborating with Contact Groups

You can use contact groups to organize your peer list. For example, you could create a group of people working on specific product features. Then, instead of selecting each member individually, you can select the group to open a session. You can add and remove groups and peers within groups. One peer can appear in multiple groups.

Sharing Projects with Peers

You share projects through a repository. To share a project, you can send the VCS link to a peer. The VCS link contains an identifier for the VCS plug-in, a reference to the VCS location for the project, and the name of the project. Your peer opens the VCS link to check out the project locally.

Peers View

The Peers view consists of the Peers pane and the Collaboration pane. The Peers pane, on the left side of the view, shows your status (Available, Away, or Offline), an informational node with your IP address, the available peers, as well as any contact groups you have created.

Individual tabbed pages in the Collaboration pane, on the right side of the view, show the peer(s) you are chatting with and the chat. Data that you have sent is displayed in this pane, as well as links that you have received (to files, web pages, diagrams, stack traces, or VCS links to a project). Tooltips in the Peers pane display the peer's user name, associated icon, IP address, status, and description.
Note: If a peer is using an international locale and fonts, and you are using the US locale and fonts, the peer may not display correctly in the Peers pane.

**Eclipse Wiki**

As part of the team collaboration features of JBuilder 2007, a Wiki for team members is now available inside the IDE. A project Wiki can be a powerful project documentation tool for team members to coordinate efforts, disperse unified project information, and take full advantage of the open-source community. Files with a .wiki extension become part of the project. Editing inside the IDE provides automatic page creation and dynamic linking (to other pages and Java source in the workspace). Checking the .wiki files into the version control system permits team members that checkout the project to view and edit the wiki. Wiki content can also be exported into a set of static web pages, including conversion of Java files into HTML.

The Eclipse Wiki plug-in is provided with JBuilder 2007. However, it must be enabled and configured according to documentation provided on the web or in the Eclipse Wiki Help files included with JBuilder 2007.

Note: Refer to the Eclipse Wiki Help files and the following “Related Reference” link for more information on enabling and configuring your Wiki plugin.

**Related Tasks**

- Enabling Peer to Peer Collaboration
- Opening a Peer to Peer Session
- Managing Contact Groups
- Chatting with Peers
- Sharing Team-Enabled Projects with Peers
- Sending Data To Peers

**Related Reference**

- Peers View
- New Contact Group
- Peer To Peer Preferences
- Send Stack Trace
- Send Web Link
- Send VCS Link
- Eclipse Wiki Editor Plugin Web Information
Procedures
Tasks

This section lists all of the task oriented help topics included for JBuilder 2007.

In This Section

- **JBuilder Project Migration**
  This section lists the tasks you need to perform to migrate a project from a previous version of JBuilder.

- **Java EE Applications**
  Java EE procedure topic links

- **Enterprise Java Bean (EJB) Applications**
  This section provides information on how to work with Enterprise Java Bean (EJB) 2.x and 3.0 applications in JBuilder 2007 applications

- **Web Services**
  This section provides tasks for designing web services.

- **Web Applications**
  This section provides a starting point for web applications topics.

- **Modeling Applications**
  This section provides information on creating modeling applications in JBuilder 2007.

- **Setting Up Database Connections**
  Provides links to information about creating database applications with JBuilder 2007.

- **ProjectAssist Procedures**
  ProjectAssist provides the server install and repository configuration or assimilation by the ProjectAssist Administrator.

- **TeamInsight Procedures**
  TeamInsight is a set of project tools that enable development teams to coordinate their work and to optimize their efforts.

- **Peer to Peer Collaboration**
  This section provides tasks for peer to peer collaboration.
JBuilder Project Migration

The migration path from JBuilder to Eclipse allows you to import every type of JBuilder project into the Eclipse workspace, including J2SE projects, J2EE projects, VisiBroker projects, RMI/JNI projects, and project groups, as well as projects that are under source control. The migration allows you to develop, test, and run your project in Eclipse.

In This Section

Building an Imported Project
Describes steps to build an imported JBuilder project.

Import a Legacy Java RMI/JNI Project from JBuilder
Describes steps to import a legacy RMI/JNI project.

Import Java EE Project From Existing Legacy JBuilder Project
Steps to import a legacy JBuilder project to a Java EE project.

Import Java SE Project From Legacy JBuilder Project
Steps to import a Legacy JBuilder project to a Java SE project.

Import Legacy Java VisiBroker Project
Describes steps to import a legacy VisiBroker project.

Importing a Source Controlled Project from a Previous Version of JBuilder
Describes how to import a source controlled project from a previous version of JBuilder into JBuilder 2007.

Running an Imported Project
Describes how to run a Java project imported from JBuilder

Setting Import Properties
Describes how to set properties for importing a JBuilder project
Building an Imported Project

JBuilder 2007 builds using a standard JDK compiler. This topic describes the auto-build feature, manually building an imported project, change build order and changing the output path.

To configure compiler options:

1. Right click the project in the Navigator ▶ View and select Properties ▶ Java Compiler.
2. Configure the compiler and the project preferences. Click Apply and OK to exit project properties.

To deactivate auto-build and enable manual build:

1. From the workbench click Project. If a checkmark is visible next to Build Automatically, click Build Automatically once to deactivate.
2. If no checkmark is visible it is already deactivated and manual builds may occur.

Note: The auto-build feature is on by default for new or imported projects. When auto-build is on, builds occur after every set of resource changes, to keep the .class file updated. When auto-build is off, builds must be invoked manually.

To build an imported project:

1. From the workbench select Project ▶ Build Project to perform an incremental build on the selected project.
2. Choose Project ▶ Build All to incrementally build all open projects.
3. Choose Project ▶ Clean to delete all previous build output for the selected project. If auto-build is on, a full build is invoked.

To change the build order:

1. Open the Build Order page of the Preferences dialog box by clicking Window ▶ Preferences ▶ General ▶ Workspace ▶ Build Order.
2. Deactivate the Use Default Build Order option.
3. Select the desired project and use the Up and Down buttons to rearrange the build order.
4. Click OK to close the dialog box and save the new build order.

To change the output path:

1. Select Project ▶ Properties ▶ Java Build Path to open the Java Build Path page of the Properties dialog box.
2. Change the folder in the Default Output Folder field.
3. Click OK to close the dialog box and save the output path.
Related Concepts

- JBuilder Project Migration Overview
- Importing Legacy Projects

Related Tasks

- Import Java EE Project From Existing Legacy JBuilder Project
- Import Java SE Project From Legacy JBuilder Project
- Setting Import Properties
- Building an Imported Project
- Running an Imported Project

Related Reference

- Eclipse Help Topic "Java builder"
Import a Legacy Java RMI/JNI Project from JBuilder

This task describes the steps to import a legacy RMI/JNI project.

To import a Java RMI/JNI project from JBuilder:

1. Use the required steps to import the project (see Related Procedures).
2. Expand the project in the Package Explorer and select an RMI or JNI file.
3. Right-click the file and choose Properties to display the Properties for <filename> dialog box.
4. Choose the RMI/JNI Properties page to view imported property settings.
   - RMI options are set in the RMI Compiler Settings area of the dialog box.
   - JNI options are set in the JNI Compiler Settings area of the dialog box.

Related Concepts

JBuilder Project Migration Overview

Related Tasks

Import Java SE Project From Legacy JBuilder Project
Import Java EE Project From Existing Legacy JBuilder Project
Import Legacy Java VisiBroker Project
Setting Import Properties
**Import Java EE Project From Existing Legacy JBuilder Project**

Java EE is supported in legacy JBuilder versions via the creation of Java EE modules in a single JBuilder project with shared source code. JBuilder 2007 is based on the Eclipse framework that supports the WTP model. The WTP model requires the creation of a project for each module. The modules table in the second page of the wizard lists the Java EE modules found in legacy JBuilder projects including Java EE, Java versions, and the corresponding JBuilder 2007 project created during the conversion process. Click on each row in the module table to display the Java and Java EE versions (project facets) for a JBuilder 2007 project.

To set up the runtime server

1. From the workbench select **Window ▶ Preferences ▶ Server ▶ Installed Runtimes** and click **Add**.
2. Select a runtime to correspond with the **Server** set up in the .jpx project to be imported.

   **Note:** To learn how to add a runtime server see Related Procedures.

3. Set the **Application Server Home Directory**, select the JRE, and click **OK**.

To activate the Project Import Wizard:

1. From the workbench select **File ▶ New ▶ Project ▶ Legacy JBuilder ▶ Java EE Project from Existing JBuilder .jpx**.

   **Note:** This wizard can also be accessed from **File ▶ Import**.

2. Select **Browse** to locate the .jpx file.

To Import libraries:

1. The Legacy JBuilder home directory contains file and libraries needed to properly import the project. If the default entry does not point to the correct .jbuilder directory, click **Browse** to locate it.

2. Review the **Library Status** table, if each library required for the import has a **green checkmark** next to it, click **Next** and continue to step 4.
   
   If any of the libraries has a **red X** next to it the library with the required directory references could not be located. In this case, go to step 3.

3. Add additional directories to be searched by selecting **Add**. Browse to the desired resource and click **Next**.

4. Click **Browse** to locate and select the runtime server.

To Set Project settings:

1. Review the **Modules** table to see each imported module is related to a project in the imported project.
2. Accept the default **Project Settings** to create a new utility module. Select the **Java Version** and go to step 3.

   **Note:** The option to create a utility project is automatically selected when a legacy JBuilder project containing more than one Java EE module is selected. A utility project is a Java project containing source code for all Java EE projects in a the workspace. This is the recommended conversion option when importing a legacy JBuilder project containing multiple modules to prevent the duplication of source code in the Java EE projects in JBuilder 2007. Creating a utility project also allows the creation of a EAR project if the legacy project does not contain a
EAR module. The EAR project is automatically include all Java EE projects, including the utility project. The utility project is included as a classpath dependency in EJB projects via the J2EE Module Dependencies properties for the EJB project. It is also included as a J2EE dependency for web projects resulting in the JAR created by the utility project being bundled into the resultant web archive’s lib directory.

3 Select the EJB project in the modules table and select **EJB 3.0** from the drop down options at the bottom of the wizard and click **Finish**.

**Note:** JBuilder 2007 supports **EJB 2.x** development using XDoclet annotations and **EJB 3.0** development using **Java EE 5.0** annotations. Legacy JBuilder projects containing XML descriptors are converted to either XDoclet annotations (for **EJB 2.1**) or to **Java EE 5.0** annotations (for **EJB 3.0**). The import wizard allows conversions from **EJB 2.1** to **EJB 3.0** using existing XML descriptors. **EJB 2.1** interfaces will not be copied over to the EJB project (or the utility project) from the legacy JBuilder project since interfaces are generated using XDoclet.

The project is now converted and project files are available in the **Navigation View**.

**Warning:** Migrating large projects can be time and memory intensive. Close all unnecessary applications before migrating a large project.

Related Concepts

- [JBuilder Project Migration Overview](#)

Related Tasks

- [Setting Import Properties](#)
- [Setting Up a Runtime Server](#)
- [Building an Imported Project](#)
- [Running an Imported Project](#)
- [Import Java SE Project From Legacy JBuilder Project](#)

Related Reference

- [Project Import Wizard](#)
Import Java SE Project From Legacy JBuilder Project

These tasks describe the steps to import a Legacy JBuilder project to a Java SE project.

**Activate the Project Import Wizard:**

1. From the workbench select File ➤ New ➤ Project ➤ Legacy JBuilder ➤ Java Project from Existing JBuilder .jpx Project.

   Tip: This wizard can also be accessed from File ➤ Import.

2. Select Browse to locate the .jpx file.

**Importing Projects:**

1. The Legacy JBuilder home directory contains file and libraries needed to properly import the project. If the default entry does not point to the correct .jbuilder directory, click Browse to locate it.

2. Review the Library Status table, if each library required for the import has a green checkmark next to it, click Next, and continue to step 4.

   If any of the libraries has a red X next to it the directories could not be located. In this case, go to step 3.

3. Add additional directories to be searched by selecting Add. Browse to the desired resource and click Next.

4. Accept the default Project Settings and click Finish.

   The following legacy artifacts are imported:

   - Libraries
   - Runtime Configurations
   - Javadoc Options
   - Java Compiler Options default file encoding
   - Java files

**Related Concepts**

- [JBuilder Project Migration Overview](#)

**Related Tasks**

- [Setting Import Properties](#)
- [Setting Up a Runtime Server](#)
- [Building an Imported Project](#)
- [Running an Imported Project](#)
- [Import Java EE Project From Existing Legacy JBuilder Project](#)

**Related Reference**

- [Project Import Wizard](#)
Import Legacy Java VisiBroker Project

This task describes how to import a Legacy Java VisiBroker project

To import a Legacy Java VisiBroker project:

1. Use the required steps to import the project (see Related Procedures).
2. Select **Window ▶ Preferences ▶ VisiBroker** to open the **VisiBroker** page of the **Preferences** dialog box.
3. Verify the directory where VisiBroker tools are installed.
4. Click **Apply** and **OK** to save the settings.
5. Expand the project in the **Package Explorer** and select an **IDL** file or a Java interface file to translate from **IDL**, to **IDL**, or **IIOP**.
6. Right-click the file and choose **Properties** to display the **Properties for <filename>** dialog box.
   - For IDL to Java files, choose the **VisiBroker IDL Properties** page and verify options in the **IDL2Java Settings** area of the dialog box.
   - For Java to IDL files, choose the **VisiBroker Java Properties** page and verify options in the **Java2IDL Settings** area of the dialog box.
   - For Java to IIOP files, choose the **VisiBroker Java Properties** page and verify options in the **Java2IIOP Settings** area of the dialog box.

Related Concepts

   [JBuilder Project Migration Overview](#)

Related Tasks

   [Import Java SE Project From Legacy JBuilder Project](#)
   [Import Java EE Project From Existing Legacy JBuilder Project](#)
   [Import a Legacy Java RMI/JNI Project from JBuilder](#)
   [Setting Import Properties](#)
Importing a Source Controlled Project from a Previous Version of JBuilder

Use the following steps to import a source controlled Legacy JBuilder project to JBuilder 2007.

To import a Java project from JBuilder

1. Follow the steps to use the project import wizard for a Java EE, Java SE, Java RMI/JNI, or Java VisiBroker project (see Related Procedures), and add the following step for a source controlled project.

2. Click the Enable VCS Plugin For This Project option. Log onto the server to check out the project. The project is checked out into the Eclipse workspace.

   **Warning:** CVS and Subversion projects that are checked into a local repository cannot be checked out.

3. Click Finish to import or check out the project.

Related Concepts

- [JBuilder Project Migration Overview](#)

Related Tasks

- [Import Java SE Project From Legacy JBuilder Project](#)
- [Import Java EE Project From Existing Legacy JBuilder Project](#)
- [Import Legacy Java VisiBroker Project](#)
- [Import a Legacy Java RMI/JNI Project from JBuilder](#)
- [Setting Import Properties](#)
Running an Imported Project

The run configuration is automatically imported when you import a project from a previous JBuilder version.

To run an imported project:

1. Select Run ➤ Run to open the Run dialog box.
2. Expand the node that matches the type of imported project in the Configurations list and choose the name of the configuration. Typically, the configuration name is the same as the project name.
3. Click the Run button to run the project.

Note: If the project uses macros in the run VM arguments, compile the project before importing it. Compilation expands the macros. If the project is not compiled, it will not run.

Related Concepts
- JBuilder Project Migration Overview
- Importing Legacy Projects

Related Tasks
- Import Java SE Project From Legacy JBuilder Project
- Import Java EE Project From Existing Legacy JBuilder Project
- Setting Import Properties
- Building an Imported Project
- Running an Imported Project
Setting Import Properties

Before importing a Java EE or VisiBroker project, configuring the application server and VisiBroker locations, you need to set properties.

To set properties for importing Java EE projects

1. From the workbench click **File Import**.
2. Click the J2EE node and select from the following import file options:
   - App Client JAR file
   - EAR File
   - J2EE Utility JAR
   - RAR file
3. After selecting the import file click **Next**, and follow the prompts to complete the import properties configuration.

To set properties for VisiBroker project imports

1. Select **Window ▶ Preferences ▶ VisiBroker** to open the **VisiBroker** page of the **Preferences** dialog box.
2. Enter the directory where VisiBroker tools are installed in the **VisiBroker Tools Directory** field. Typically, this is the **bin** folder of the Borland Enterprise Server installation.
3. Click **Apply** and **OK** to save project settings.
4. Select **Project ▶ Properties ▶ Builders** to open the **Builders** page of the **Properties** dialog box. Make sure the **VisiBroker Builder** option in the **Configure The Builders For This Project** list is selected.
5. Click **OK** to save project settings.

Related Concepts

- JBuilder Project Migration Overview
- Importing Legacy Projects

Related Tasks

- Import Java SE Project From Legacy JBuilder Project
- Import Java EE Project From Existing Legacy JBuilder Project
- Setting Import Properties
- Running an Imported Project
- Building an Imported Project
Java EE Applications

Java EE components are assembled into an application and are deployed to production, to be run and managed by the Java EE server. Use the following links to discover detailed information about creating Java EE applications using JBuilder 2007.

In This Section

Creating a Java EE Project
Use this topic to get started creating a Java EE project with JBuilder 2007

Developing Java EE Applications
Describes task-related Java EE project development using JBuilder 2007.

Import a Java EE Project
Topic details steps required to import a Java EE project into the IDE.

Publishing a Java EE Application to a Server Runtime
Describes how to publish a Java application to a runtime server.

Running an Application on a Runtime Server
Describes how to set up a runtime server.

Setting Up a Runtime Server
Steps to create a runtime server instance in a project.

Setting Up and Using a Borland Application Server
Steps to setup a Borland Application Server (BAS) runtime server.
Creating a Java EE Project

The Java EE perspective includes the following workbench views:

- Java Servlet and JavaServer Pages (JSP)
- Application clients and applets components that run on the client
- Technology web components that run on the server
- Enterprise JavaBeans (EJB)

Create a Java EE Project

1. From the main window, click File New Project.
2. Click the + next to the J2EE folder to reveal all options.
3. Depending on the project requirements, choose from one of the following project types:
   - Application Client Project
   - Connector Project
   - Enterprise Application Project
   - Utility Project
4. Click Next.
5. Type a project name in the Project Name text field.
6. Select the desired configuration parameters for:
   - Project Contents
   - Target Runtime
   - Configurations
   - EAR Membership
7. Click Next.
8. Configure the desired Project Facet parameters and click Next.
9. Configure the desired Source Folder or accept the default value and click Finish.
Related Concepts

Java EE Applications Overview
Creating a Java EE Project
Web Services Overview
Enterprise Java Bean (EJB) Applications Overview
Modeling Applications Overview
Runtime Servers

Related Tasks

Setting Up a Runtime Server
Publishing a Java EE Application to a Server Runtime
Running an Application on a Runtime Server

Related Reference

Eclipse help topic (J2EE) “Reference”
Eclipse help topic “J2EE Applications”
Eclipse help topic “Working with projects”
Eclipse help topic “Project Explorer view in the J2EE perspective”
Eclipse help topic “J2EE architecture”
Eclipse help topic “J2EE perspective”
Developing Java EE Applications

Use the following steps to develop a new Java EE project with JBuilder 2007.

To create a new project:

1. Set the workbench perspective to Java:
   - Window ➤ Perspective ➤ Java
2. Select File ➤ New ➤ Project.
3. Select the J2EE Node and click Next.
4. Set the Project Name, Target Runtime, , and Configurations preferences and click Finish.

Related Concepts

- Java EE Applications Overview
- Runtime Servers

Related Tasks

- Creating a Java EE Project
- Setting Up a Runtime Server
- Publishing a Java EE Application to a Server Runtime
- Running an Application on a Runtime Server
- Enterprise Java Bean (EJB) Applications

Related Reference

- "Eclipse Help Topic “Changing the Java compiler version for a J2EE project"
- "Eclipse Help Topic “J2EE Applications”"
Import a Java EE Project

To import a Java EE project into the IDE:

2. Select File ▶ New ▶ Project to invoke the New Project Wizard.
3. Browse to the EJB folder and select EJB Project.
4. Configure the following project preferences:
   - Project Name
   - Target Runtime
   - Configurations
   - Project Contents
5. For Project Contents, deactivate the Use Default checkbox and browse to the desired directory to select the Java EE project to be imported.
   - Click Next to invoke the Project Facets dialog.
6. Activate the checkbox next to Java Version 5.0 and click Finish. The Java EE project is now open in the Navigation View.

Related Concepts
- Java EE Applications Overview

Related Tasks
- Developing Java EE Applications
- Setting Up a Runtime Server
- Running an Application on a Runtime Server
Publishing a Java EE Application to a Server Runtime

This section describes how to publish a Java application to a runtime server. The server runtime has to be configured in the Servers view with a Java EE project from the current workspace added to the runtime server for deployment. The publish action redeployes the selected projects for an application server.

To publish an application to a server runtime:

1. Select Servers Windows ► Show View ► Other ► Server ► Servers to open the Servers view window.
2. Installed server runtimes appear in the Servers view window. Right-click on the runtime server name and click Add and Remove Projects to deploy or undeploy any Java EE projects in your workspace.
3. Right-click on the runtime server name and click Publish, or click the Publish to the server icon at the top of the Servers view window. The Publish action redeployes available projects for the selected server.

Related Concepts
- Java EE Applications Overview
- Runtime Servers

Related Tasks
- Creating a Java EE Project
- Setting Up a Runtime Server
- Running an Application on a Runtime Server
- Enterprise Java Bean (EJB) Applications

Related Reference
- Eclipse help topic “Web application overview”
- Eclipse help topic “Server targeting for Web applications”
- Eclipse help topic “Running a Java program”
- Eclipse help topic “Debugging a servlet on a server”
Running an Application on a Runtime Server

JBuilder 2007 supports various Java EE runtime servers. This topic describes how to run an Java application on a runtime server using the JBoss application server technology.

To run an application on the JBoss server:

1. Open the project in JBuilder 2007.
2. In the Navigation view select the project folder.
3. Right click the highlighted project to reveal the drop down menu options and select Run as.
4. Click Run on Server and click JBoss in the Select the server type list and click Next.
5. Confirm the default configuration settings in the New JBoss Server window and click Next.
6. In the Add and Remove Projects window, confirm the project is listed in the Configured projects section and click Finish.

Related Concepts

- Runtime Servers
- Java EE Applications Overview

Related Tasks

- Creating a Java EE Project
- Setting Up a Runtime Server
- Publishing a Java EE Application to a Server Runtime
- Enterprise Java Bean (EJB) Applications

Related Reference

- Eclipse help topic “Web application overview”
- Eclipse help topic “Server targeting for Web applications”
- Eclipse help topic “Running a Java program”
- Eclipse help topic “Debugging a servlet on a server”
Setting Up a Runtime Server

Java EE 5.0 applications work with a runtime application server. There are several types of Java EE applications, including Enterprise Java Bean (EJB) 3.0 applications, web applications, and web services. This section describes how to set up a runtime server for any of these application types. JBuilder 2007 supports various Java EE runtime servers.

You create a server using the runtime environment best suited to the project by defining a pointer from the workbench to an existing installation of an application server. Use the following steps to set up a runtime server in JBuilder 2007.

Note: This topic describes setting up a runtime server. The tasks describe creating a runtime server using the JBoss application server technology. Most other runtime servers can be set up in a similar fashion.

To set up a runtime server:

1. From the Workbench select Window ➤ Preferences.
2. Select Servers ➤ Installed Runtimes.

   Tip: A best practice when using the runtime servers is to choose one of the versions that has (CodeGear or Borland) after it. These versions have been extended to support specific features.

   Note: The remaining steps describe creating an application server using the JBoss runtime environment.

3. Click Add, choose the appropriate JBoss (Borland) runtime environment, and click Next.
4. Accept the default JRE and click Browse to choose the Application Server Directory.
5. Select the root directory of JBuilder 2007 and choose the JBuilder 2007 folder.
6. Select the thirdparty folder, select the folder representing the desired JBoss runtime environment, and click Finish.

   The desired JBoss runtime now appears in the Installed Server Runtime Environments list.

To associate the desired runtime with the project folder:

1. Right click on the project folder in the Navigation View.
2. Select Properties
3. Select Targeted Runtimes
4. Activate the checkbox next to the desired server, click Apply and OK.

   The desired runtime is now associated with the project.
Related Concepts

- Runtime Servers
- Java EE Applications Overview
- Web Applications Overview
- Web Services Overview
- Enterprise Java Beans (EJB) Overview

Related Tasks

- Creating a Java EE Project
- Publishing a Java EE Application to a Server Runtime
- Running an Application on a Runtime Server

Related Reference

- Borland Application Server Documentation
- Eclipse help topic “Server targeting for Web applications”
- Eclipse help topic “Web application overview”
Setting Up and Using a Borland Application Server

Java EE 5.0 applications work with a runtime application server. This section describes how to set up a Borland Application Server (BAS) runtime.

You create a server using the runtime environment best suited to the project by defining a pointer from the workbench to an existing installation of an application server. Use the following steps to set up a BAS runtime server in JBuilder 2007.

To set up a BAS runtime server in JBuilder 2007:

1. From the Workbench select Window ▶ Preferences.
2. Select Servers ▶ Installed Runtimes.
3. Click Add, choose the appropriate BAS runtime environment (with OpenJMS or Tibco). Click Next.
4. Accept the default JRE and click Browse to choose the root of the BAS directory.
5. Select the root directory of JBuilder 2007 and choose the JBuilder 2007 folder.
6. Click Finish.

The desired BAS runtime now appears in the Installed Server Runtime Environments list.

7. To configure the server for deployment select Window ▶ Show View ▶ Other ▶ Server ▶ Servers

   **Note:** The default server setup for deployment is the j2eeSample configuration with the partition, WelcomePartition, which is the default managed partition in the sample configuration. You can only start managed partitions in JBuilder 2007; therefore, you must only setup managed partitions for startup and deployment from within the IDE.

To debug with the BAS Runtime in JBuilder 2007:

1. To prepare to debug a partition in JBuilder 2007, you must complete the following steps to configure a partition for remote debugging. Start the Borland Management Console.
2. Start the BAS server.
3. Locate the partition you want to debug under the Management Hub.
4. Right-click on the partition name and choose Properties. Select the Partition Process Settings tab.
5. Check the Enable JPDA Remote Debugging option. Set the transport address field to the desired port number. Uncheck the Suspend Partition Until Debugger Attaches option. Click OK.
6. Shut down the BAS server from the console. Launch JBuilder 2007. Configure the server runtime for deployment as described in the previous task.
7. Start the server in the Servers view. With the server selected, click Run ▶ Debug. In the Debug window, click on Remote Java Application in the left-side list. The icon meanings appear on the right-side. Click on the New icon at the top of the left-side list. The right-side pane now has a dialog to attach a Java virtual machine that accepts debug connections. Name the configuration in the Name field. Set the host name in the Host field to localhost. Set the port number to the partition's remote debug port number.
8. Click on Debug to start the debug session.
9. After the debugger launches successfully and stops at the breakpoint, add the project to the Default Source Lookup for the debugger if you encounter Source not found errors in the Debug perspective.
To create and run an EJB client:

1. Create a new Java class with a main method. In the main method, modify the Java Naming and Directory Interface (JNDI) code to lookup the EJB. Lookup codes does not need any server-specific properties for BAS.

2. Select Run ➤ Run.

3. In the Run window, click on Java Application in the left-side list. The icon meanings appear on the right-side. Click on the New icon at the top of the left-side list. The right-side pane now has a dialog to specify a new configuration. Name the configuration in the Name field. Set the main class to the EJB client class in the Main class field.

4. Click the Arguments tab. Set the VM arguments field to:
   
   ```
   -Dvbroker.agent.port=port_no -Djava.endorsed.dirs=/BorlandAppServer/lib/endorsed
   ```
   
   where port_no is the osagent port for the application server.

5. Start the BAS server and deploy the EJB application.

6. Go to Window ➤ Preferences. Type User Libraries in the type filter text area. Click on User Libraries.

7. Click on New in the User Libraries pane on the right-side of the screen. Enter EJB Stubs in the User library name field. Click OK.

8. Select the new library from the list. Click on Add JARs. Add the deployed EJB JAR from:
   
   ```
   /AppServer/var/domains/configurations/configuration_name/mos/partition_name
   ```
   
   where configuration_name and partition_name have been replaced with your server configuration data. Close the Preferences dialog window.

9. In the Navigator view, right-click on the project and select Properties. Select Java Build Path and Libraries.
    Click Add and add the EJB Stubs library to the project. Click on Add again and add the Client Library for BAS 6.7 to the project. Run the client configuration.

To stop the management agent after stopping the BAS server:

1. When the server is stopped in the IDE, only the configuration and partition are stopped which improves wait times during restarts. The management agent cannot be stopped from within the IDE.

2. To stop the management agent, launch the BAS console from /BorlandAppServer/bin.

3. Expand the Management Hubs node, right-click on the hub and select Shutdown.

Related Concepts

- Runtime Servers
- Java EE Applications Overview
- Web Applications Overview
- Web Services Overview
- Enterprise Java Beans (EJB) Overview

Related Tasks

- Creating a Java EE Project
- Setting Up a Runtime Server
- Publishing a Java EE Application to a Server Runtime
- Running an Application on a Runtime Server

Related Reference

- Borland Application Server Documentation
Enterprise Java Bean (EJB) Applications

The tasks in this area provide information on how to work with Enterprise Java Bean (EJB) 2.x and 3.0 applications in JBuilder 2007 applications.

In This Section

- **Adding a Business Method to an EJB**
  Describes how to add a business method to an EJB.

- **Adding a CMP Field to a CMP Entity Bean**
  Describes how to add a new CMP field to a CMP entity bean.

- **Adding a Create Method to an EJB 2.x Entity Bean**
  Describes how to add a create method to an EJB 2.x entity bean.

- **Adding a Find Method to an EJB 2.x Entity Bean**
  Describes how to add a find method to an entity bean.

- **Adding a Home Method to an EJB 2.x Entity Bean**
  Describes how to add a home method to an EJB 2.x entity bean.

- **Adding a New Field to an Enterprise Java Bean (EJB)**
  Describes how to add a new field to an EJB.

- **Adding a New Method to an EJB**
  Describes how to add a new method to an EJB.

- **Adding a New Named Native Query to an EJB 3.0 Entity Bean**
  Describes how to add a new named native query to an EJB 3.0 entity bean.

- **Adding a New Named Query to an EJB 3.0 Entity Bean**
  Describes how to add a new named query to an EJB 3.0 entity bean.

- **Adding a New Post-Load Method to an EJB 3.0 Entity Bean**
  Describes how to add a new post-load method to an EJB 3.0 entity bean.

- **Adding a New Post-Persist Method to an EJB 3.0 Entity Bean**
  Describes how to add a new post-persist method to an EJB 3.0 entity bean.

- **Adding a New Post-Remove Method to an EJB 3.0 Entity Bean**
  Describes how to add a new post-remove method to an EJB 3.0 entity bean.

- **Adding a New Post-Update Method to an EJB 3.0 Entity Bean**
  Describes how to add a new post-update method to an EJB 3.0 entity bean.

- **Adding a New Pre-Persist Method to an EJB 3.0 Entity Bean**
  Describes how to add a new pre-persist method to an EJB 3.0 entity bean.

- **Adding a New Pre-Remove Method to an EJB 3.0 Entity Bean**
  Describes how to add a new pre-remove method to an EJB 3.0 entity bean.

- **Adding a New Pre-Update Method to an EJB 3.0 Entity Bean**
  Describes how to add a new pre-update method to an EJB 3.0 entity bean.

- **Adding a Post-Construct Method to an EJB 3.0 Session Bean**
  Describes how to add a post-construct method to an EJB 3.0 session bean.

- **Adding a Pre-Destroy Method to an EJB 3.0 Session Bean**
  Describes how to add a pre-destroy method to an EJB 3.0 session bean.

- **Adding a Primary Key Join Field to an Entity Bean**
  Describes how to add a primary key join column to an entity bean.
Adding a Result Set Mapping to an EJB 3.0 Entity Bean
Describes how to add a result set mapping to an EJB 3.0 entity bean.

Adding a Select Method to an EJB 2.x Entity Bean
Describes how to add a select method to an EJB 2.x entity bean.

Adding a Timeout Method to an EJB 3.0 Session Bean
Describes how to add a timeout method to an EJB 3.0 session bean.

Adding an Interceptor Method to an EJB 3.0 Session Bean
Describes how to add an interceptor method to an EJB 3.0 session bean.

Building a Package of Enterprise Java Beans (EJBs)
Describes how to build a package of EJBs for later deployment to an application server.

Create an EJB Modeling Project with XDoclet Annotations
Create an EJB modeling with XDoclet Annotations.

Creating a Bean-Managed-Persistence (BMP) Entity Bean
Describes how to create a new BMP entity bean.

Creating a Container-Managed-Persistence (CMP) Entity Bean
Describes how to create a new CMP entity bean.

Creating a Message Bean
Describes how to create a new message bean

Creating a Message Destination for a Message Bean
Describes how to create a message destination for a message bean.

Creating a Message Destination Link for a Message Bean
Describes how to create a message destination link for a message bean.

Creating a New Enterprise Java Bean (EJB) Project
Describes how to create a new Enterprise Java Bean (EJB) project.

Creating a New Session Bean
Describes how to create a new session bean.

Creating a One-Way Relationship Between Entity Beans
Describes how to create a one-way relationship between entity beans.

Creating a Relationship Between Entity Beans
Describes how to create a relationship between entity beans.

Creating a Relationship With Primary Key Mapping Between Entity Beans
Describes how to create a relationship with primary key mapping between entity beans.

Creating a Resource Reference
Describes how to create a resource reference for an entity bean.

Creating a Run-As-Security Link
Describes how to create a run-as-security link in an EJB project.

Creating a Security Role
Describes how to create a security role in an EJB project.

Creating a Security Role Reference
Describes how to create a security role reference in an EJB project.

Creating an EJB 3.0 Application Exception Class
Describes how to create a new EJB 3.0 application exception class.
Creating an EJB 3.0 Embeddable Class
Describes how to create a new EJB 3.0 embeddable class.

Creating an EJB 3.0 Embeddable Class Reference
Describes how to create an EJB 3.0 embeddable class reference.

Creating an EJB 3.0 Embeddable ID Class Reference
Describes how to create an EJB 3.0 embeddable ID class reference.

Creating an EJB 3.0 Entity Listener Reference
Describes how to create an EJB 3.0 entity listener reference.

Creating an EJB 3.0 Interceptor Reference
Describes how to create an EJB 3.0 interceptor reference.

Creating an EJB 3.0 Mapped Superclass
Describes how to create a new EJB 3.0 mapped superclass.

Creating an EJB Reference
Describes how to create an EJB reference.

Creating an Enterprise Java Bean (EJB) Project
Describes how to create an Enterprise Java Bean (EJB) project.

Creating an Environment Entry
Describes how to create an environment entry for an entity bean.

Creating an Environment Resource Reference
Describes how to create an environment resource reference for an entity bean.

Creating an Injected EJB Reference
Describes how to create an injected EJB reference for a session bean.

Creating the Primary Key for an Entity Bean
Describes how to create the primary key for an entity bean.

Deleting a Field from an Enterprise Java Bean (EJB)
Describes how to delete a field from an Enterprise Java Bean (EJB).

Deleting a Method from an EJB
Describes how to delete a method from an EJB.

Enabling XDoclet
Describes how to enable XDoclet.

Import a Java EE Project
Topic details steps required to import a Java EE project into the IDE.

Importing Entity Beans from a Database
Describes how to import database tables into an EJB project as entity beans.

Modifying an Enterprise Java Bean (EJB)
Describes how to modify an enterprise java bean (EJB).

Removing an Enterprise Java Bean (EJB)
Describes how to remove an Enterprise Java Bean (EJB).

Viewing the Source Code of an Enterprise Java Bean (EJB)
Describes how to view the source code of an Enterprise Java Bean (EJB).
Adding a Business Method to an EJB

This section describes how to add a business method to an entity bean or a session bean.

To add a business method to an EJB in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the session bean.
4. Click on the new method to view its properties.
5. Select properties for your new business method.

To add a business method to an EJB using the Code Editor:

1. Open the source code for the EJB.
2. Add the new business method directly to the source code.
3. Add annotations.
4. For EJB 2.x projects, add local and remote setting interfaces.
5. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

Enterprise Java Beans (EJB) Overview

Related Tasks

Creating a Java Class for a Web Service
Viewing the Source Code of an Enterprise Java Bean (EJB)
Modifying an Enterprise Java Bean (EJB)
Deleting a Method from an EJB
Adding a CMP Field to a CMP Entity Bean

This section describes how to add a new Container-Managed Persistence (CMP) field to a CMP entity bean using either the Modeling Perspective or the Code Editor.

A CMP field is a virtual field in an entity bean. A CMP field refers to a column in a database table, and the entity bean implements getters and setter methods for the field.

To add a new CMP field to a CMP entity bean in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
4. Select CMP Field.
5. Enter the name of the new field.
6. Click in the new field to view and set its properties.

To add a CMP field to a CMP entity bean to an EJB using the Code Editor:

1. Open the source code for the EJB.
2. Add the new field directly to the source code.
3. Add annotations to the source code.
4. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview

Related Tasks

- Creating a Java Class for a Web Service
- Adding a New Field to an Enterprise Java Bean (EJB)
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Field from an Enterprise Java Bean (EJB)
Adding a Create Method to an EJB 2.x Entity Bean

This section describes how to add a create method to an EJB 2.x entity bean.

Note: The EJB 3.0 specification has eliminated this interface for entity beans.

To add a create method to an entity bean in the Modeling Perspective:

1. Open the class diagram for the entity bean.
2. Right click on the entity bean.
4. Click twice on the new method to open the in-place editor.
5. Enter the name and return type of the new create method.

To add a create method to an entity bean using the Code Editor:

1. Open source code for the entity bean.
2. Add the new create method directly to the source code.
3. Add annotations.
4. For EJB 2.x, add code to expose the create method in the home interface.
5. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

Enterprise Java Beans (EJB) Overview
Entity Bean Overview

Related Tasks

Creating a Java Class for a Web Service
Adding a New Method to an EJB
Adding a Find Method to an EJB 2.x Entity Bean
Adding a Home Method to an EJB 2.x Entity Bean
Adding a Select Method to an EJB 2.x Entity Bean
Viewing the Source Code of an Enterprise Java Bean (EJB)
Modifying an Enterprise Java Bean (EJB)
Deleting a Method from an EJB
Adding a Find Method to an EJB 2.x Entity Bean

This section describes how to add a find method to an entity bean.

Note: The EJB 3.0 specification has replaced this method with named queries.

To add a find method to an entity bean in the Modeling Perspective:
1. Open the class diagram for the entity bean.
2. Right click on the entity bean.
4. Click twice on the new method to open the in-place editor.
5. Enter the name, query, and return type of the new find method.

To add a find method to an entity bean using the Code Editor:
1. Open source code for the entity bean.
2. Add the new find method directly to the source code.
3. Add annotations.
4. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts
- Enterprise Java Beans (EJB) Overview
- Entity Bean Overview

Related Tasks
- Creating a Java Class for a Web Service
- Adding a New Named Query to an EJB 3.0 Entity Bean
- Adding a New Method to an EJB
- Adding a Create Method to an EJB 2.x Entity Bean
- Adding a Home Method to an EJB 2.x Entity Bean
- Adding a Select Method to an EJB 2.x Entity Bean
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Method from an EJB
Adding a Home Method to an EJB 2.x Entity Bean

This section describes how to add a home method to an EJB 2.x entity bean.

**Note:** The EJB 3.0 specification has eliminated this interface for Entity beans.

**To add a home method to an EJB 2.x entity bean in the Modeling Perspective:**

1. Open the class diagram for the session bean.
2. Right click on the entity bean.
3. Select **New ➤ Home Method**.
4. Click twice on the new method to open the in-place editor.
5. Enter the name and return type of the new home method.

**To add a home method to an EJB 2.x entity bean using the Code Editor:**

1. Open source code for the entity bean.
2. Add the new home method directly to the source code.
3. Add annotations.
4. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

**Related Concepts**

- Enterprise Java Beans (EJB) Overview
- Session Bean Overview

**Related Tasks**

- Creating a Java Class for a Web Service
- Adding a New Method to an EJB
- Adding a Home Method to an EJB 2.x Entity Bean
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Method from an EJB
Adding a New Field to an Enterprise Java Bean (EJB)

This section describes how to add a new field to an Enterprise Java Bean (EJB) using either the Modeling Perspective or the Code Editor.

To add a new field to an EJB in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
4. Select Attribute or the type of attribute to be added.
5. Click twice on the new attribute in the diagram.
6. Enter the name and data type of the new attribute.

To add a new field to an EJB using the Code Editor:

1. Open the source code for the EJB.
2. Add the new field directly to the source code.
3. Add annotations to the source code
4. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

Enterprise Java Beans (EJB) Overview

Related Tasks

Creating a Java Class for a Web Service
Adding a CMP Field to a CMP Entity Bean
Viewing the Source Code of an Enterprise Java Bean (EJB)
Modifying an Enterprise Java Bean (EJB)
Deleting a Field from an Enterprise Java Bean (EJB)
Adding a New Method to an EJB

This section describes how to add a new method to an EJB using either the Modeling Perspective or the Code Editor. Refer to the links at the bottom of this page for information on how to add a business, pre-persist, pre-remove, pre-update, post-persist, post-remove, post-update, or post-load method to a 3.0 EJB. Refer to the links at the bottom of this page for information on how to add a business, create, home, find, or select method to a 2.x EJB.

Note: The EJB 3.0 specification is quite different from the EJB 2.x specification. JBuilder 2007 provides support for both EJB 2.x and EJB 3.0 methods. Make sure that you are using the correct methods for your version of EJB.

To add a new method to an EJB in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
4. Select Operation or the type of method to be added.
5. Click twice on the new method in the diagram.
6. Enter the name and return type of the new method.

To add a new method to an EJB using the Code Editor:

1. Open source code for the EJB.
2. Add the new method and annotations directly to the source code.
3. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.
Related Concepts

- Enterprise Java Beans (EJB) Overview

Related Tasks

- Creating a Java Class for a Web Service
- Adding a Business Method to an EJB
- Adding an Interceptor Method to an EJB 3.0 Session Bean
- Adding a Post-Construct Method to an EJB 3.0 Session Bean
- Adding a Pre-Destroy Method to an EJB 3.0 Session Bean
- Adding a Timeout Method to an EJB 3.0 Session Bean
- Adding a New Pre-Update Method to an EJB 3.0 Entity Bean
- Adding a New Pre-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Pre-Remove Method to an EJB 3.0 Entity Bean
- Adding a New Post-Update Method to an EJB 3.0 Entity Bean
- Adding a New Post-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Post-Remove Method to an EJB 3.0 Entity Bean
- Adding a Create Method to an EJB 2.x Entity Bean
- Adding a Find Method to an EJB 2.x Entity Bean
- Adding a Home Method to an EJB 2.x Entity Bean
- Adding a Select Method to an EJB 2.x Entity Bean
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Method from an EJB
Adding a New Named Native Query to an EJB 3.0 Entity Bean

This section describes how to add a named native query to an EJB 3.0 entity bean using either the Modeling Perspective or the Code Editor.

**Note:** This method is only applicable to EJB 3.0 entity beans.

To add a new named native query to an EJB 3.0 entity bean in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
3. Select **New**.
4. Select **Named Native Query**.
5. Enter the name of the new named query.
6. Click on the named native query to view its properties.
7. Enter the query text in the **Query...Value** box.
8. Enter the result set in the **ResultSet Mapping...Value** box.

To add a new named native query to an EJB 3.0 entity bean using the Code Editor:

1. Open the source code for the EJB.
2. Add the new named native query and annotations directly to the source code.
3. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- [Enterprise Java Beans (EJB) Overview](#)

Related Tasks

- Creating a Java Class for a Web Service
- Adding a New Method to an EJB
- Adding a New Named Query to an EJB 3.0 Entity Bean
- Adding a Result Set Mapping to an EJB 3.0 Entity Bean
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Method from an EJB
Adding a New Named Query to an EJB 3.0 Entity Bean

This section describes how to add a named query to an EJB 3.0 entity bean using either the Modeling Perspective or the Code Editor.

**Note:** This method is only applicable to EJB 3.0 entity beans.

To add a new named query to an EJB 3.0 entity bean in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
3. Select **New**.
4. Select **Named Query**.
5. Enter the name of the new named query.
6. Click on the named query to view its properties.
7. Enter the query text in the **Query...Value** box.

To add a new named query to an EJB 3.0 entity bean using the Code Editor:

1. Open the source code for the EJB.
2. Add the new named query and annotations directly to the source code.
3. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- [Enterprise Java Beans (EJB) Overview](#)

Related Tasks

- [Creating a Java Class for a Web Service](#)
- [Adding a New Method to an EJB](#)
- [Adding a New Named Native Query to an EJB 3.0 Entity Bean](#)
- [Adding a Result Set Mapping to an EJB 3.0 Entity Bean](#)
- [Viewing the Source Code of an Enterprise Java Bean (EJB)](#)
- [Modifying an Enterprise Java Bean (EJB)](#)
- [Deleting a Method from an EJB](#)
Adding a New Post-Load Method to an EJB 3.0 Entity Bean

This section describes how to add a new post-load method to an EJB 3.0 entity bean using either the Modeling Perspective or the Code Editor.

**Note:** This method is only applicable to EJB 3.0 entity beans.

To add a new post-load method to an EJB 3.0 entity bean in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
3. Select **New**.
4. Select **Post-Load Method**.
5. Enter the name of the new method.
6. Click on the method to view and set its properties.

To add a new post-load method to an EJB 3.0 entity bean using the Code Editor:

1. Open the source code for the EJB.
2. Add the new method and annotations directly to the source code.
3. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- [Enterprise Java Beans (EJB) Overview](#)

Related Tasks

- [Creating a Java Class for a Web Service](#)
- Adding a New Method to an EJB
- Adding a Business Method to an EJB
- Adding a New Pre-Update Method to an EJB 3.0 Entity Bean
- Adding a New Pre-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Pre-Remove Method to an EJB 3.0 Entity Bean
- Adding a New Post-Update Method to an EJB 3.0 Entity Bean
- Adding a New Post-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Post-Remove Method to an EJB 3.0 Entity Bean
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Method from an EJB
Adding a New Post-Persist Method to an EJB 3.0 Entity Bean

This section describes how to add a new post-persist method to an EJB 3.0 entity bean using either the Modeling Perspective or the Code Editor.

**Note:** This method is only applicable to EJB 3.0 entity beans.

**To add a new post-persist method to an EJB 3.0 entity bean in the Modeling Perspective:**

1. Open the class diagram for the EJB.
2. Right click on the EJB.
3. Select **New**.
4. Select **Post-Persist Method**.
5. Enter the name of the new method.
6. Click on the method to view and set its properties.

**To add a new post-persist method to an EJB 3.0 entity bean using the Code Editor:**

1. Open the source code for the EJB.
2. Add the new method and annotations directly to the source code.
3. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

**Related Concepts**

- [Enterprise Java Beans (EJB) Overview](#)

**Related Tasks**

- [Creating a Java Class for a Web Service](#)
- [Adding a New Method to an EJB](#)
- [Adding a Business Method to an EJB](#)
- [Adding a New Pre-Update Method to an EJB 3.0 Entity Bean](#)
- [Adding a New Pre-Persist Method to an EJB 3.0 Entity Bean](#)
- [Adding a New Pre-Remove Method to an EJB 3.0 Entity Bean](#)
- [Adding a New Post-Update Method to an EJB 3.0 Entity Bean](#)
- [Adding a New Post-Load Method to an EJB 3.0 Entity Bean](#)
- [Adding a New Post-Remove Method to an EJB 3.0 Entity Bean](#)
- [Viewing the Source Code of an Enterprise Java Bean (EJB)](#)
- [Modifying an Enterprise Java Bean (EJB)](#)
- [Deleting a Method from an EJB](#)
Adding a New Post-Remove Method to an EJB 3.0 Entity Bean

This section describes how to add a new post-remove method to an EJB 3.0 entity bean using either the Modeling Perspective or the Code Editor.

**Note:** This method is only applicable to EJB 3.0 entity beans.

To add a new post-remove method to an EJB 3.0 entity bean in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
3. Select **New**.
4. Select **Post-Remove Method**.
5. Enter the name of the new method.
6. Click on the method to view and set its properties.

To add a new post-remove method to an EJB 3.0 entity bean using the Code Editor:

1. Open the source code for the EJB.
2. Add the new method and annotations directly to the source code.
3. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts
- Enterprise Java Beans (EJB) Overview

Related Tasks
- Creating a Java Class for a Web Service
- Adding a New Method to an EJB
- Adding a Business Method to an EJB
- Adding a New Pre-Update Method to an EJB 3.0 Entity Bean
- Adding a New Pre-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Pre-Remove Method to an EJB 3.0 Entity Bean
- Adding a New Post-Update Method to an EJB 3.0 Entity Bean
- Adding a New Post-Load Method to an EJB 3.0 Entity Bean
- Adding a New Post-Persist Method to an EJB 3.0 Entity Bean
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Method from an EJB
Adding a New Post-Update Method to an EJB 3.0 Entity Bean

This section describes how to add a new post-update method to an EJB 3.0 entity bean using either the Modeling Perspective or the Code Editor.

Note: This method is only applicable to EJB 3.0 entity beans.

To add a new post-update method to an EJB 3.0 entity bean in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
4. Select Post-Update Method.
5. Enter the name of the new method.
6. Click on the method to view and set its properties.

To add a new post-update method to an EJB 3.0 entity bean using the Code Editor:

1. Open the source code for the EJB.
2. Add the new method and annotations directly to the source code.
3. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts
- Enterprise Java Beans (EJB) Overview

Related Tasks
- Creating a Java Class for a Web Service
- Adding a New Method to an EJB
- Adding a Business Method to an EJB
- Adding a New Pre-Update Method to an EJB 3.0 Entity Bean
- Adding a New Pre-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Pre-Remove Method to an EJB 3.0 Entity Bean
- Adding a New Post-Load Method to an EJB 3.0 Entity Bean
- Adding a New Post-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Post-Remove Method to an EJB 3.0 Entity Bean
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Method from an EJB
Adding a New Pre-Persist Method to an EJB 3.0 Entity Bean

This section describes how to add a new pre-persist method to an EJB 3.0 entity bean using either the Modeling Perspective or the Code Editor.

**Note:** This method is only applicable to EJB 3.0 entity beans.

To add a new pre-persist method to an EJB 3.0 entity bean in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
3. Select **New**.
4. Select **Pre-Persist Method**.
5. Enter the name of the new method.
6. Click on the method to view and set its properties.

To add a new pre-persist method to an EJB 3.0 entity bean using the Code Editor:

1. Open the source code for the EJB.
2. Add the new method and annotations directly to the source code.
3. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- [Enterprise Java Beans (EJB) Overview](#)

Related Tasks

- [Creating a Java Class for a Web Service](#)
- [Adding a New Method to an EJB](#)
- [Adding a Business Method to an EJB](#)
- [Adding a New Pre-Update Method to an EJB 3.0 Entity Bean](#)
- [Adding a New Pre-Remove Method to an EJB 3.0 Entity Bean](#)
- [Adding a New Post-Update Method to an EJB 3.0 Entity Bean](#)
- [Adding a New Post-Load Method to an EJB 3.0 Entity Bean](#)
- [Adding a New Post-Persist Method to an EJB 3.0 Entity Bean](#)
- [Adding a New Post-Remove Method to an EJB 3.0 Entity Bean](#)
- [Viewing the Source Code of an Enterprise Java Bean (EJB)](#)
- [Modifying an Enterprise Java Bean (EJB)](#)
- [Deleting a Method from an EJB](#)
Adding a New Pre-Remove Method to an EJB 3.0 Entity Bean

This section describes how to add a new pre-remove method to an EJB 3.0 entity bean using either the Modeling Perspective or the Code Editor.

Note: This method is only applicable to EJB 3.0 entity beans.

To add a new pre-remove method to an EJB 3.0 entity bean in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
4. Select Pre-Remove Method.
5. Enter the name of the new method.
6. Click on the method to view and set its properties.

To add a new pre-remove method to an EJB 3.0 entity bean using the Code Editor:

1. Open the source code for the EJB.
2. Add the new method and annotations directly to the source code.
3. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview

Related Tasks

- Creating a Java Class for a Web Service
- Adding a New Method to an EJB
- Adding a Business Method to an EJB
- Adding a New Pre-Update Method to an EJB 3.0 Entity Bean
- Adding a New Pre-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Post-Update Method to an EJB 3.0 Entity Bean
- Adding a New Post-Load Method to an EJB 3.0 Entity Bean
- Adding a New Post-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Post-Remove Method to an EJB 3.0 Entity Bean
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Method from an EJB
Adding a New Pre-Update Method to an EJB 3.0 Entity Bean

This section describes how to add a new pre-update method to an EJB 3.0 entity bean using either the Modeling Perspective or the Code Editor.

Note: This method is only applicable to EJB 3.0 entity beans.

To add a new pre-update method to an EJB 3.0 entity bean in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
4. Select Pre-Update Method.
5. Enter the name of the new method.
6. Click on the method to view and set its properties.

To add a new pre-update method to an EJB 3.0 entity bean using the Code Editor:

1. Open the source code for the EJB.
2. Add the new method and annotations directly to the source code.
3. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview

Related Tasks

- Creating a Java Class for a Web Service
- Adding a New Method to an EJB
- Adding a Business Method to an EJB
- Adding a New Pre-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Pre-Remove Method to an EJB 3.0 Entity Bean
- Adding a New Post-Update Method to an EJB 3.0 Entity Bean
- Adding a New Post-Load Method to an EJB 3.0 Entity Bean
- Adding a New Post-Persist Method to an EJB 3.0 Entity Bean
- Adding a New Post-Remove Method to an EJB 3.0 Entity Bean
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Method from an EJB
Adding a Post-Construct Method to an EJB 3.0 Session Bean

This section describes how to add a post-construct method to an EJB 3.0 session bean.

Note: This method is only available for EJB 3.0 session beans.

To add a post-construct method to an EJB 3.0 session bean in the Modeling Perspective:

1. Open the class diagram for the session bean.
2. Right click on the session bean.
4. Click on the new method to view its properties.
5. Select properties for your new post-construct method.

To add a post-construct method to an EJB 3.0 session bean using the Code Editor:

1. Open the source code for the session bean.
2. Add the new post-construct method directly to the source code.
3. Add annotations.
4. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- Session Bean Overview

Related Tasks

- Creating a Java Class for a Web Service
- Creating a New Session Bean
- Adding a New Method to an EJB
- Adding a Business Method to an EJB
- Adding an Interceptor Method to an EJB 3.0 Session Bean
- Adding a Pre-Destroy Method to an EJB 3.0 Session Bean
- Adding a Timeout Method to an EJB 3.0 Session Bean
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Method from an EJB
Adding a Pre-Destroy Method to an EJB 3.0 Session Bean

This section describes how to add a pre-destroy method to an EJB 3.0 session bean.

**Note:** This method type is only available for EJB 3.0 session beans.

**To add a pre-destroy method to an EJB 3.0 session bean in the Modeling Perspective:**

1. Open the class diagram for the session bean.
2. Right click on the session bean.
3. Select **New ► Pre-Destroy Method**.
4. Click on the new method to view its properties.
5. Select properties for your new pre-destroy method.

**To add a pre-destroy method to an EJB 3.0 session bean using the Code Editor:**

1. Open the source code for the session bean.
2. Add the new pre-destroy method directly to the source code.
3. Add annotations.
4. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

**Related Concepts**

- Enterprise Java Beans (EJB) Overview
- Session Bean Overview

**Related Tasks**

- Creating a Java Class for a Web Service
- Creating a New Session Bean
- Adding a New Method to an EJB
- Adding a Business Method to an EJB
- Adding an Interceptor Method to an EJB 3.0 Session Bean
- Adding a Post-Construct Method to an EJB 3.0 Session Bean
- Adding a Timeout Method to an EJB 3.0 Session Bean
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Method from an EJB
Adding a Primary Key Join Field to an Entity Bean

This section describes how to add a primary key join column to an entity bean using either the Modeling Perspective or the Code Editor.

To add a primary key join column to an entity bean in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
4. Select Primary Key Join Column.
5. Enter the name of the new field.
6. Click in the new field to view its properties.
7. Enter the join definition in the Definition...Value box.
8. Enter the referenced column name in the Referenced Column Name...Value box.

To add a primary key join column to an entity bean to an EJB using the Code Editor:

1. Open the source code for the EJB.
2. Add the new field directly to the source code.
3. Add annotations to the source code.
4. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- Entity Bean Overview

Related Tasks

- Creating a Java Class for a Web Service
- Adding a New Field to an Enterprise Java Bean (EJB)
- Creating the Primary Key for an Entity Bean
- Creating a One-Way Relationship Between Entity Beans
- Creating a Relationship Between Entity Beans
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Field from an Enterprise Java Bean (EJB)
Adding a Result Set Mapping to an EJB 3.0 Entity Bean

This section describes how to add a result set mapping to an EJB 3.0 entity bean using either the Modeling Perspective or the Code Editor.

Note: This capability is only applicable to EJB 3.0 entity beans.

To add a new result set mapping to an EJB 3.0 entity bean in the Modeling Perspective:

1. Open the class diagram for the EJB.
2. Right click on the EJB.
5. Enter the name of the new result set mapping.
6. Click on the result set mapping to view its properties.
7. Enter the column results in the Column Results...Value box.
8. Enter the entity results in the Entity Results...Value box.

To add a new result set mapping to an EJB 3.0 entity bean using the Code Editor:

1. Open the source code for the EJB.
2. Add the new result set mapping and annotations directly to the source code.
3. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

Enterprise Java Beans (EJB) Overview

Related Tasks

Creating a Java Class for a Web Service
Adding a New Method to an EJB
Adding a New Named Query to an EJB 3.0 Entity Bean
Adding a New Named Native Query to an EJB 3.0 Entity Bean
Viewing the Source Code of an Enterprise Java Bean (EJB)
Modifying an Enterprise Java Bean (EJB)
Deleting a Method from an EJB
Adding a Select Method to an EJB 2.x Entity Bean

This section describes how to add a select method to an EJB 2.x entity bean.

Note:  The EJB 3.0 specification has eliminated this method for entity beans.

To add a select method to an entity bean in the Modeling Perspective:

1. Open the class diagram for the entity bean.
2. Right click on the entity bean.
4. Click twice on the new method to open the in-place editor.
5. Enter the name, query, and return type of the new select method.

To add a select method to an entity bean using the Code Editor:

1. Open source code for the entity bean.
2. Add the new select method directly to the source code.
3. Add annotations.
4. Save your changes.

Note:  Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

Enterprise Java Beans (EJB) Overview
Entity Bean Overview

Related Tasks

Creating a Java Class for a Web Service
Adding a New Method to an EJB
Adding a Create Method to an EJB 2.x Entity Bean
Adding a Find Method to an EJB 2.x Entity Bean
Viewing the Source Code of an Enterprise Java Bean (EJB)
Modifying an Enterprise Java Bean (EJB)
Deleting a Method from an EJB
Adding a Timeout Method to an EJB 3.0 Session Bean

This section describes how to add a timeout method to an EJB 3.0 session bean.

Note: This method type is only available for EJB 3.0 session beans.

To add a timeout method to an EJB 3.0 session bean in the Modeling Perspective:

1. Open the class diagram for the session bean.
2. Right click on the session bean.
4. Click on the new method to view its properties.
5. Select properties for your new timeout method.

To add a timeout method to an EJB 3.0 session bean using the Code Editor:

1. Open the source code for the session bean.
2. Add the new timeout method directly to the source code.
3. Add annotations.
4. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- Session Bean Overview

Related Tasks

- Creating a Java Class for a Web Service
- Creating a New Session Bean
- Adding a New Method to an EJB
- Adding a Business Method to an EJB
- Adding an Interceptor Method to an EJB 3.0 Session Bean
- Adding a Post-Construct Method to an EJB 3.0 Session Bean
- Adding a Pre-Destroy Method to an EJB 3.0 Session Bean
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Deleting a Method from an EJB
Adding an Interceptor Method to an EJB 3.0 Session Bean

This section describes how to add an interceptor method to an EJB 3.0 session bean.

**Note:** This method is only available for EJB 3.0 session beans.

To add an interceptor method to an EJB 3.0 session bean in the Modeling Perspective:

1. Open the class diagram for the session bean.
2. Right click on the session bean.
3. Select **New** ‣ **Interceptor Method**.
4. Click on the new method to view its properties.
5. Select properties for your new interceptor method.

To add an interceptor method to an EJB 3.0 session bean using the Code Editor:

1. Open the source code for the session bean.
2. Add the new interceptor method directly to the source code.
3. Add annotations.
4. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- [Enterprise Java Beans (EJB) Overview](#)
- [Session Bean Overview](#)

Related Tasks

- [Creating a Java Class for a Web Service](#)
- [Creating a New Session Bean](#)
- [Adding a New Method to an EJB](#)
- [Adding a Business Method to an EJB](#)
- [Adding a Post-Construct Method to an EJB 3.0 Session Bean](#)
- [Adding a Pre-Destroy Method to an EJB 3.0 Session Bean](#)
- [Adding a Timeout Method to an EJB 3.0 Session Bean](#)
- [Creating an EJB 3.0 Interceptor Reference](#)
- [Viewing the Source Code of an Enterprise Java Bean (EJB)](#)
- [Modifying an Enterprise Java Bean (EJB)](#)
- [Deleting a Method from an EJB](#)
Building a Package of Enterprise Java Beans (EJBs)

This section describes how to build a package of EJBs for later deployment to an application server.

To create a package in the Modeling Perspective

1. Double click on the process node to open the default class diagram.
2. Choose the Package tool from the palette.
3. Place the package in the diagram.
4. Place your EJBs in the package.

Related Concepts

- Enterprise Java Bean (EJB) Applications Overview

Related Tasks

- Creating an Enterprise Java Bean (EJB) Project
- Creating a Java Class for a Web Service
Create an EJB Modeling Project with XDoclet Annotations

The Create an EJB Modeling Project with XDoclet Annotations wizard converts an existing Web Tools Platform (WTP) EJB project to an EJB modeling project using XDoclet annotations.

Warning: The WTP EJB project must exist in the current Workspace and XDoclet annotation support must be installed and configured to work with the Workbench.

To create an EJB modeling project with XDoclet Annotations

1. Select File ➤ New ➤ Project to invoke the New Project wizard.
2. In the Select a Wizard window navigate to the EJB folder and select EJB Modeling Project from an XDoclet Annotated WTP Project, and click Next.
3. A list of WTP EJB projects in the current Workspace is displayed.
   
   Note: Only WTP EJB projects (not EJB modeling projects) are displayed.

4. Activate the checkbox next to the desired EJB project and click Finish.

The WTP EJB project is converted to an EJB modeling project and EJB diagrams are created based on EJB source and XDoclet annotations in the WTP EJB project.

Related Concepts
   
   JBuilder Project Migration Overview

Related Tasks
   
   Setting Import Properties
   Building an Imported Project
   Enabling XDoclet

Related Reference
   
   Creating Enterprise Beans with XDoclet Annotation Support
Creating a Bean-Managed-Persistence (BMP) Entity Bean

This section describes how to create a new BMP entity bean.

To create a new BMP entity bean in the Modeling Perspective:

1. Open the Modeling Perspective.
2. Bring up the model for your EJB project.
3. Select the BMP entity bean tool.
4. Place the BMP entity bean in the model.

To create a new BMP entity bean in the Code Editor:

1. Create a new Java file for your BMP entity bean.
2. Code the BMP entity bean by hand.
3. Add annotations.
4. Add the new BMP entity bean source file to your project.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts
- Enterprise Java Beans (EJB) Overview
- Entity Bean Overview

Related Tasks
- Creating a Java Class for a Web Service
- Importing Entity Beans from a Database
Creating a Container-Managed-Persistence (CMP) Entity Bean

This section describes how to create a new container-managed-persistence (CMP) entity bean.

To create a new CMP entity bean in the Modeling Perspective:

1. Open the modeling perspective.
2. Bring up the model for your EJB project.
3. Select the **CMP entity bean** tool.
4. Place the CMP entity bean in the model.

To create a new CMP entity bean in the Code Editor:

1. Create a new Java file for your CMP entity bean.
2. Code the CMP entity bean by hand.
3. Add annotations.
4. Add the new CMP entity bean source file to your project.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- [Enterprise Java Beans (EJB) Overview](#)
- [Entity Bean Overview](#)

Related Tasks

- [Creating a Java Class for a Web Service](#)
- [Importing Entity Beans from a Database](#)
Creating a Message Bean

This section describes how to create a new message bean using either the Modeling Perspective or the Code Editor.

To create a new message bean in the Modeling Perspective:

1. Open the Modeling Perspective.
2. Bring up the model for your EJB project.
3. Select the **Message Bean** tool.
4. Place the message bean in the model.
5. Define the message destination and message destination link for the message bean.

To create a new message bean in the Code Editor:

1. Create a new Java file for your message bean.
2. Code the message bean by hand.
3. Add annotations.
4. Add the new message bean source file to your project.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts
- Enterprise Java Beans (EJB) Overview
- Message Bean Overview

Related Tasks
- Creating a Java Class for a Web Service
- Creating a Message Destination for a Message Bean
- Creating a Message Destination Link for a Message Bean
Creating a Message Destination for a Message Bean

This section describes how to create a message destination for a message bean.

To create a new message bean destination in the Modeling Perspective:

1. Open the Modeling Perspective.
2. Bring up the model for your EJB project.
3. Select the Message Bean Destination tool.
4. Place the message bean destination in the model.

Related Concepts
- Enterprise Java Beans (EJB) Overview
- Message Bean Overview

Related Tasks
- Creating a Message Bean
- Creating a Message Destination Link for a Message Bean
Creating a Message Destination Link for a Message Bean

This section describes how to create a message destination link for a message bean.

To create a new message bean destination link in the Modeling Perspective:

1. Open the Modeling Perspective.
2. Bring up the model for your EJB project.
3. Verify the existence of the message bean and message bean destination.
4. Select the Message Bean Destination Link tool.
5. Link the message bean to the message bean destination.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- Message Bean Overview

Related Tasks

- Creating a Message Bean
- Creating a Message Destination for a Message Bean
Creating a New Enterprise Java Bean (EJB) Project

This section describes how to create a new Enterprise Java Bean project.

To create a new EJB project:

1. Select File ➤ New ➤ Project.
2. Select EJB and then select the type of EJB project to create.
3. Enter a name for your new project.
4. Select a target runtime and project configuration for the project.
5. Specify the Java build settings.
6. Click the Finish button.

Related Concepts

- Enterprise Java Bean (EJB) Applications Overview

Related Tasks

- Creating an Enterprise Java Bean (EJB) Project
Creating a New Session Bean

This section describes how to create a new session bean using either the Modeling Perspective or the Code Editor.

To create a new session bean in the Modeling Perspective:

1. Open the Modeling Perspective.
2. Bring up the model for your EJB project.
3. Select the Session Bean tool.
4. Place the session bean in the model.

To create a new session bean in the Code Editor:

1. Create a new Java file for your session bean.
2. Code the session bean by hand.
3. Add annotations.
4. Add the new session bean source file to your project.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- Session Bean Overview

Related Tasks

- Creating a Java Class for a Web Service
Creating a One-Way Relationship Between Entity Beans

This section describes how to create a one-way relationship between entity beans. The relationship needs to match the relationship between tables in the underlying database.

To create a one-way relationship between entity beans in the Modeling Perspective:
1. Open the class diagram for the entity beans.
2. Select the EJB Relationship (Unidirectional) tool from the palette.
3. Select the source entity bean.
4. Select the target entity bean.

To create a one-way relationship between entity beans using the Code Editor:
1. Open the source code for the entity beans.
2. Add the new relationship directly to the source code.
3. Save your changes.

Related Concepts
- Enterprise Java Beans (EJB) Overview
- Entity Bean Overview

Related Tasks
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Creating a Relationship Between Entity Beans
- Creating the Primary Key for an Entity Bean
- Adding a Primary Key Join Field to an Entity Bean
Creating a Relationship Between Entity Beans

This section describes how to create a relationship between entity beans. The relationship needs to match the relationship between tables in the underlying database.

To create a relationship between entity beans in the Modeling Perspective:

1. Open the class diagram for the entity beans.
2. Select the **EJB Relationship** tool from the palette.
3. Select the source entity bean.
4. Select the target entity bean.

To create a relationship between entity beans using the Code Editor:

1. Open the source code for the entity bean.
2. Add the new relationship directly to the source code.
3. Save your changes.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- Entity Bean Overview

Related Tasks

- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Creating a One-Way Relationship Between Entity Beans
- Creating a Relationship With Primary Key Mapping Between Entity Beans
- Creating the Primary Key for an Entity Bean
- Adding a Primary Key Join Field to an Entity Bean
Creating a Relationship With Primary Key Mapping Between Entity Beans

This section describes how to create a relationship with primary key mapping between entity beans. To use primary key mapping, the source and target beans must have the same primary key field name. The relationship also needs to match the relationship between tables in the underlying database.

To create a relationship with primary key mapping between entity beans in the Modeling Perspective:

1. Open the class diagram for the entity beans.
2. Select the **EJB Relation With PK Mapping** tool from the palette.
3. Select the source entity bean.
4. Select the target entity bean.

To create a relationship with primary key mapping between entity beans in different packages in the Modeling Perspective:

1. Open the class diagram for the source entity bean.
2. Select the **EJB Relation With PK Mapping** tool from the palette.
3. Select the source entity bean.
4. Click on any whitespace in the diagram.
5. Select the target entity bean from the list.

To create a relationship with primary key mapping between entity beans using the Code Editor:

1. Open the source code for the entity beans.
2. Add the new relationship directly to the source code.
3. Save your changes.

Related Concepts

- [Enterprise Java Beans (EJB) Overview](#)
- [Entity Bean Overview](#)

Related Tasks

- [Viewing the Source Code of an Enterprise Java Bean (EJB)](#)
- [Modifying an Enterprise Java Bean (EJB)](#)
- [Creating a Relationship Between Entity Beans](#)
- [Creating a One-Way Relationship Between Entity Beans](#)
- [Creating the Primary Key for an Entity Bean](#)
- [Adding a Primary Key Join Field to an Entity Bean](#)
Creating a Resource Reference

This section describes how to create a resource reference for an entity bean.

**Note:** For information on creating an injected resource reference, refer to the “Creating an Injected Resource Reference” link in the Related Information list at the bottom of this page.

To create a resource reference in the Modeling Perspective:

1. Open the class diagram for the entity bean.
2. Right click on the entity bean.
3. Select **New ➤ Resource Reference**.
4. Place the new resource reference in the diagram.

Related Concepts

- [Enterprise Java Beans (EJB) Overview](#)
- [EJB Environment and Resources Overview](#)

Related Tasks

- [Creating an Environment Entry](#)
- [Creating an Environment Resource Reference](#)
- [Viewing the Source Code of an Enterprise Java Bean (EJB)](#)
- [Modifying an Enterprise Java Bean (EJB)](#)
Creating a Run-As-Security Link

This section describes how to create a run-as-security link in an EJB project.

To create a run-as-security link in an EJB project in the Modeling Perspective:

1. Open the class diagram for the EJB project.
2. Verify the existence of the EJB and the security role that you want to connect.
3. Select the Run-As-Security Link tool from the palette.
4. Click on the EJB that needs a run-as security link.
5. Click on a security role to link the EJB to the security role.

To create a run-as-security link in an EJB project using the Code Editor:

1. Open the source code for project.
2. Add the run-as security link directly to the source code.
3. Add annotations.
4. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- EJB Security Roles Overview

Related Tasks

- Creating a Security Role
- Creating a Security Role Reference
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
Creating a Security Role

This section describes how to create a security role in an EJB project.

To create a security role in an EJB project in the Modeling Perspective:

1. Open the class diagram for the EJB project.
2. Select the Security Role tool from the palette.
3. Click twice on the new security role.
4. Enter the name of the security role.

To create a security role in an EJB project using the Code Editor:

1. Open source code for the project.
2. Add the security role directly to the source code.
3. Add annotations.
4. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- EJB Security Roles Overview

Related Tasks

- Creating a Run-As-Security Link
- Creating a Security Role Reference
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
Creating a Security Role Reference

This section describes how to create a security role reference in an EJB project.

To create a security role reference in an EJB project in the Modeling Perspective:

1. Open the class diagram for the EJB project.
2. Verify the existence of the source EJB and the target security role.
3. Select the **Security Role Reference** tool from the palette.
4. Click on the EJB that needs a security role reference.
5. Click on a security role to link the EJB to the security role.

To create a security role reference in an EJB project using the Code Editor:

1. Open source code for the project.
2. Add the security role reference directly to the source code.
3. Add annotations.
4. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- EJB Security Roles Overview

Related Tasks

- Creating a Security Role
- Creating a Run-As-Security Link
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
Creating an EJB 3.0 Application Exception Class

This section describes how to create a new EJB 3.0 application exception class using either the Modeling Perspective or the Code Editor.

To create a new EJB 3.0 application exception class in the Modeling Perspective:

1. Open the Modeling Perspective.
2. Bring up the model for your EJB project.
3. Select the Application Exception tool.
4. Place the application exception class in the model.

To create a new EJB 3.0 application exception class in the Code Editor:

1. Create a new Java file for your EJB 3.0 application exception class.
2. Code the EJB 3.0 application exception class by hand.
3. Add annotations.
4. Add the new EJB 3.0 application exception class source file to your project.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview

Related Tasks

- Creating a Java Class for a Web Service
- Creating an EJB 3.0 Mapped Superclass
- Creating an EJB 3.0 Embeddable Class
Creating an EJB 3.0 Embeddable Class

This section describes how to create a new EJB 3.0 embeddable class using either the Modeling Perspective or the Code Editor.

To create a new EJB 3.0 embeddable class in the Modeling Perspective:

1. Open the Modeling Perspective.
2. Bring up the model for your EJB project.
3. Select the Embeddable Class tool.
4. Place the embeddable class in the model.

To create a new EJB 3.0 embeddable class in the Code Editor:

1. Create a new Java file for your EJB 3.0 embeddable class.
2. Code the EJB 3.0 embeddable class by hand.
3. Add annotations.
4. Add the new EJB 3.0 embeddable class source file to your project.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

Enterprise Java Beans (EJB) Overview

Related Tasks

Creating a Java Class for a Web Service
Creating an EJB 3.0 Application Exception Class
Creating an EJB 3.0 Mapped Superclass
Creating an EJB 3.0 Embeddable Class Reference
Creating an EJB 3.0 Embeddable ID Class Reference
Creating an EJB 3.0 Embeddable Class Reference

This section describes how to create an EJB 3.0 embeddable class reference.

**Note:** This feature is only available for EJB 3.0 projects.

To create an EJB 3.0 embeddable class reference in the Modeling Perspective:

1. Open the class diagrams for the EJBs.
2. Select the **Embeddable Class Reference** tool from the palette.
3. Click on the source EJB.
4. Click on the target embeddable class.

**Note:** You can create an EJB 3.0 embeddable class reference from an EJB in one EJB package to an embedded class in a different package.

To create an EJB 3.0 embeddable class reference using the Code Editor:

1. Open the source code for the EJB.
2. Add the new EJB 3.0 embeddable class reference and Java EE 5.0 annotation directly to the source code.
3. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts
- Enterprise Java Beans (EJB) Overview
- EJB Environment and Resources Overview

Related Tasks
- Creating an EJB Reference
- Creating an EJB 3.0 Embeddable Class
- Creating an EJB 3.0 Embeddable ID Class Reference
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
Creating an EJB 3.0 Embeddable ID Class Reference

This section describes how to create an EJB 3.0 embeddable ID class reference.

**Note:** This feature is only available for EJB 3.0 projects.

To create an EJB 3.0 embeddable ID class reference in the Modeling Perspective:

1. Open the class diagrams for the EJBs.
2. Select the **Embeddable ID Class Reference** tool from the palette.
3. Click on the source EJB.
4. Click on the target embeddable class.

**Note:** You can create an EJB 3.0 embeddable ID class reference from an EJB in one EJB package to an embedded class in a different package.

To create an EJB 3.0 embeddable ID class reference to a class in a different package:

1. Open the class diagram for the source EJB.
2. Select the **Embeddable ID Class Reference** tool from the palette.
3. Click on the source EJB.
4. Click on any whitespace in the diagram.
5. Select the target embeddable class from the list.

To create an EJB 3.0 embeddable ID class reference using the Code Editor:

1. Open the source code for the EJB.
2. Add the new EJB 3.0 embeddable ID class reference and Java EE 5.0 annotation directly to the source code.
3. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- [Enterprise Java Beans (EJB) Overview](#)
- [EJB Environment and Resources Overview](#)

Related Tasks

- [Creating an EJB Reference](#)
- [Creating an EJB 3.0 Embeddable Class](#)
- [Creating an EJB 3.0 Embeddable Class Reference](#)
- [Viewing the Source Code of an Enterprise Java Bean (EJB)](#)
- [Modifying an Enterprise Java Bean (EJB)](#)
Creating an EJB 3.0 Entity Listener Reference

This section describes how to create an EJB 3.0 entity listener reference.

**Note:** This feature is only available for EJB 3.0 projects.

To create an EJB 3.0 entity listener reference in the Modeling Perspective:

1. Open the class diagram for the entity bean.
2. Select the **Entity Listener Reference** tool from the palette.
3. Click on the source entity bean.
4. Click on the target entity listener.

**Note:** You can create an EJB 3.0 interceptor reference from an EJB in one EJB package to an interceptor in a different package.

To create an EJB 3.0 entity listener reference from an EJB in one package to an interceptor in a different package:

1. Open the class diagram for the entity bean.
2. Select the **Entity Listener Reference** tool from the palette.
3. Click on the source entity bean.
4. Click any whitespace in the diagram.
5. Select the listener class in the dialog.

To create an EJB 3.0 entity listener reference using the Code Editor:

1. Open the source code for the entity bean.
2. Add the new EJB 3.0 entity listener reference and Java EE 5.0 annotation directly to the source code.
3. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

**Related Concepts**

- Enterprise Java Beans (EJB) Overview
- Entity Bean Overview

**Related Tasks**

- Creating a Java Class for a Web Service
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
Creating an EJB 3.0 Interceptor Reference

This section describes how to create an EJB 3.0 interceptor reference.

**Note:** This feature is only available for EJB 3.0 projects.

To create an EJB 3.0 interceptor reference in the Modeling Perspective:

1. Open the class diagrams for the EJBs.
2. Select the **Interceptor Reference** tool from the palette.
3. Click on the source EJB method.
4. Click on the target interceptor.

**Note:** You can create an EJB 3.0 interceptor reference from an EJB in one EJB package to an interceptor in a different package.

To create an EJB 3.0 interceptor reference using the Code Editor:

1. Open the source code for the EJB.
2. Add the new EJB 3.0 interceptor reference and Java EE 5.0 annotation directly to the source code.
3. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- [Enterprise JavaBeans (EJB) Overview](#)

Related Tasks

- [Adding an Interceptor Method to an EJB 3.0 Session Bean](#)
- [Viewing the Source Code of an Enterprise Java Bean (EJB)](#)
- [Modifying an Enterprise Java Bean (EJB)](#)
Creating an EJB 3.0 Mapped Superclass

This section describes how to create a new EJB 3.0 mapped superclass using either the Modeling Perspective or the Code Editor.

To create a new EJB 3.0 mapped superclass in the Modeling Perspective:

1. Open the Modeling Perspective.
2. Bring up the model for your EJB project.
3. Select the Mapped Superclass tool.
4. Place the mapped superclass in the model.

To create a new EJB 3.0 mapped superclass in the Code Editor:

1. Create a new Java file for your EJB 3.0 mapped superclass.
2. Code the EJB 3.0 mapped superclass by hand.
3. Add annotations.
4. Add the new EJB 3.0 mapped superclass source file to your project.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

Enterprise Java Beans (EJB) Overview

Related Tasks

Creating a Java Class for a Web Service
Creating an EJB 3.0 Application Exception Class
Creating an EJB 3.0 Embeddable Class
Creating an EJB Reference

This section describes how to create an EJB reference.

Note: For information on how to create an injected EJB reference, refer to the “Creating an Injected EJB Reference” link in the Related Information section at the end of this page.

To create an EJB reference in the Modeling Perspective:

1. Open the class diagrams for the EJBs.
2. Select the EJB Reference tool from the palette.
3. Click on the source EJB.
4. Click on the target EJB.

Note: You can create an EJB reference from an EJB in one EJB package to an EJB in a different package.

To create an EJB reference using the Code Editor:

1. Open the source code for the EJBs.
2. Add the new reference and Xdoclet annotation directly to the source code.
3. Save your changes.

Note: Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- EJB Environment and Resources Overview

Related Tasks

- Creating an Injected EJB Reference
- Creating an Environment Entry
- Creating an Environment Resource Reference
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
Creating an Enterprise Java Bean (EJB) Project

This section describes how to create an Enterprise Java Bean (EJB) project.

To create an EJB project:

1. Select File ➔ New ➔ Project.
2. Select EJB and then select the type of EJB modeling project to create.
3. Enter a name for your new project.
4. Select a target runtime for the project.
5. Select the Xdoclet or Java EE 5.0 configuration for the project.
6. Specify the Java build settings.
7. Click the Finish button.

Related Concepts

- Enterprise Java Bean (EJB) Applications Overview

Related Tasks

- Creating a New Enterprise Java Bean (EJB) Project
- Creating an Enterprise Java Bean (EJB) Modeling Project
- Create a Java Persistence API (JPA) Modeling Project
- Importing an Enterprise Java Bean (EJB) Modeling Project
Creating an Environment Entry
This section describes how to create an environment entry for an entity bean.

To create an environment entry in the Modeling Perspective:

1. Open the class diagram for the entity bean.
2. Right click on the entity bean.
4. Place the new environment entry in the diagram.

Related Concepts
- Enterprise Java Beans (EJB) Overview
- EJB Environment and Resources Overview

Related Tasks
- Creating an EJB Reference
- Creating an Environment Resource Reference
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
Creating an Environment Resource Reference

This section describes how to create an environment resource reference for an entity bean.

To create an environment resource reference in the Modeling Perspective:

1. Open the class diagram for the entity bean.
2. Right click on the entity bean.
3. Select **New ▶ Environment Resource Reference**.
4. Place the new environment resource reference in the diagram.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- EJB Environment and Resources Overview

Related Tasks

- Creating an EJB Reference
- Creating an Environment Entry
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
Creating an Injected EJB Reference

This section describes how to create an injected EJB reference.

**Note:** This feature is only available for EJB 3.0 projects.

For information on how to create a non-injected EJB reference, refer to the “Creating an EJB Reference” link in the Related Information section at the end of this page.

To create an injected EJB reference in the Modeling Perspective:

1. Open the class diagrams for the EJBs.
2. Select the **Injected EJB Reference** tool from the palette.
3. Click on the source session bean.
4. Click on the target EJB.

**Note:** You can create an injected EJB reference from an EJB in one EJB package to an EJB in a different package.

To create an injected EJB reference using the Code Editor:

1. Open the source code for the EJBs.
2. Add the new injected reference and Java EE 5.0 annotation directly to the source code.
3. Save your changes.

**Note:** Adding artifacts in model diagrams generates source code and annotations. When you add artifacts manually, you are responsible for creating both source code and annotations.

Related Concepts

- Enterprise Java Beans (EJB) Overview
- EJB Environment and Resources Overview

Related Tasks

- Creating an EJB Reference
- Creating an Environment Entry
- Creating an Environment Resource Reference
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
Creating the Primary Key for an Entity Bean

This section describes how to create the primary key for an entity bean. The primary key needs to match the primary key in the underlying database table.

To create a simple primary key field for an entity bean in the Modeling Perspective:

1. Open the class diagram for the entity bean.
2. Right click on the entity bean.
4. Select Simple PK Field.

To create a compound primary key for an entity bean in the Modeling Perspective:

1. Click on the EJB in the Property Editor.
2. Click on CMP Field.
3. Click on the Standard EJB Properties tab in the Property Editor.
4. Select the fields used in the compound primary key.

To create the primary key field for an entity bean using the Code Editor:

1. Open the source code for the Entity bean.
2. Add the new field directly to the source code.
3. Save your changes.

Related Concepts

Enterprise Java Beans (EJB) Overview
Entity Bean Overview

Related Tasks

Creating an Enterprise Java Bean (EJB) Project
Viewing the Source Code of an Enterprise Java Bean (EJB)
Modifying an Enterprise Java Bean (EJB)
Adding a Primary Key Join Field to an Entity Bean
Creating a One-Way Relationship Between Entity Beans
Creating a Relationship Between Entity Beans
Deleting a Field from an Enterprise Java Bean (EJB)

This section describes how to delete a field from an Enterprise Java Bean (EJB).

To delete a field from an EJB in the Modeling Perspective:

1. Open the diagram for the EJB.
2. Right click on the field to be deleted.
3. Select delete.
4. Confirm the deletion of the field.

To delete a field from an EJB using the Code Editor:

1. Open source code for the EJB.
2. Delete the field directly from the source code.
3. Save your changes.

Related Concepts

- Enterprise Java Beans (EJB) Overview

Related Tasks

- Creating a Java Class for a Web Service
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Adding a New Field to an Enterprise Java Bean (EJB)
Deleting a Method from an EJB

This section describes how to delete a method from an EJB.

To delete a method from an EJB in the Modeling Perspective:

1. Open the diagram for the EJB.
2. Right click on the method to be deleted.
3. Select delete.
4. Confirm the deletion of the method.

To delete a method from an EJB using the Code Editor:

1. Open source code for the EJB.
2. Delete the method from the source code.
3. Save your changes.

Related Concepts

- Enterprise Java Beans (EJB) Overview

Related Tasks

- Creating a Java Class for a Web Service
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Modifying an Enterprise Java Bean (EJB)
- Adding a New Method to an EJB
Enabling XDoclet

Many Java EE applications require XDoclet support. This section describes how to enable XDoclet support.

XDoclet 1.2.3 with support for JDK 5.0 ships with JBuilder 2007 and is available in the JBuilder 2007 Eclipse plugins directory.

To enable XDoclet support:

1. Select Window ▶ Preferences ▶ XDoclet.
2. In the Set XDoclet Runtime Preferences dialog, check the Enable XDoclet Builder box to enable XDoclet support. Specify the home directory in XDoclet Home field. Select the appropriate version in the Version dropdown menu.
3. Click Apply and click OK.
4. You may also need to select Window ▶ Preferences ▶ XDoclet ▶ ejbdoclet/webdoclet options.
5. In the ejbdoclet or webdoclet dialogs, check the applicable tasks and servers.
6. Click Restore Defaults to restore default settings or Apply to apply the designated settings. Click OK.

Related Concepts

Java EE Applications Overview
Creating a Java EE Project

Related Tasks

Setting Up a Runtime Server
Import a Java EE Project

To import a Java EE project into the IDE:

2. Select File ➤ New ➤ Project to invoke the New Project Wizard.
3. Browse to the EJB folder and select EJB Project.
4. Configure the following project preferences:
   - Project Name
   - Target Runtime
   - Configurations
   - Project Contents

5. For Project Contents, deactivate the Use Default checkbox and browse to the desired directory to select the Java EE project to be imported.
   - Click Next to invoke the Project Facets dialog.
6. Activate the checkbox next to Java Version 5.0 and click Finish.
   - The Java EE project is now open in the Navigation View.

Related Concepts
   - Java EE Applications Overview

Related Tasks
   - Developing Java EE Applications
   - Setting Up a Runtime Server
   - Running an Application on a Runtime Server
Importing Entity Beans from a Database

This section describes how to import database tables into an EJB 2.x or EJB 3.0 project as entity beans.

To import entity beans from a database server in an EJB 2.x modeling project:

1. Right click on the EJB modeling project in the Model Navigator.
2. Select **Import Entity Beans from Database...**
3. Select the database connection from the drop-down list. If your database connection is missing from the list, click **Add connections**... to add the database connection to the list.
4. Select the database schema for importation.
5. Specify the source folder and package into which to import the entity beans.
   
   **Note:** You can specify a new package into which to import the data.
6. Select the tables to be imported.
7. Click **Finish** to import the entity beans.

To import entity beans from a database server in an EJB 3.0 modeling project:

1. Right click on the EJB modeling project in the Model Navigator.
2. Select **Import Entities from Database...**
3. Select the database connection from the drop-down list. If your database connection is missing from the list, click **Add connections**... to add the database connection to the list.
4. Select the database schema for importation.
5. Specify the source folder and package into which to import the entity beans.
   
   **Note:** You can specify a new package into which to import the data.
6. Select the tables to be imported.
7. Click **Finish** to import the entity beans.

Related Concepts

- [Enterprise Java Beans (EJB) Overview](#)

Related Tasks

- [Creating a Java Class for a Web Service](#)
- [Viewing the Source Code of an Enterprise Java Bean (EJB)](#)
- [Modifying an Enterprise Java Bean (EJB)](#)
Modifying an Enterprise Java Bean (EJB)

This section describes how to modify an enterprise java bean (EJB) using the Code Editor.

To modify an EJB:

1. Open the EJB's source code.
2. Make your changes directly to the bean's source code.
3. Save your changes.

Related Concepts

- Enterprise Java Beans (EJB) Overview

Related Tasks

- Creating a Java Class for a Web Service
- Viewing the Source Code of an Enterprise Java Bean (EJB)
- Adding a New Field to an Enterprise Java Bean (EJB)
- Deleting a Field from an Enterprise Java Bean (EJB)
- Adding a New Method to an EJB
- Deleting a Method from an EJB
Removing an Enterprise Java Bean (EJB)
This section describes how to remove an Enterprise Java Bean (EJB).

To remove an EJB using the Package Explorer:
1. Open the package containing the EJB.
2. Click on the Java file containing the bean.
3. Press the **DELETE** key on your keyboard.
4. Confirm the deletion.

To remove an EJB in the Modeling Perspective:
1. Open the diagram containing the ERJB.
2. Click on the EJB.
3. Press the Delete key on your keyboard.
4. Confirm the deletion.

Related Concepts
- Enterprise Java Beans (EJB) Overview

Related Tasks
- Creating an Enterprise Java Bean (EJB) Project
- Creating a Java Class for a Web Service
- Modifying an Enterprise Java Bean (EJB)
Viewing the Source Code of an Enterprise Java Bean (EJB)

This section describes how to view the source code of an Enterprise Java Bean (EJB).

To view the source code of an EJB from the Modeling Perspective:

1. Open the Modeling Perspective.
2. Bring up the model for your EJB project.
3. Right click on the EJB.
4. Select Open to view the source code for the EJB.

To view the source code of an EJB from the Package Explorer:

1. Select the module in which the EJB resides.
2. Click on the package containing the EJB.
3. Double click on the EJB source file.

Related Concepts

Enterprise Java Beans (EJB) Overview

Related Tasks

Creating an Enterprise Java Bean (EJB) Project
Modifying an Enterprise Java Bean (EJB)
Web Services

The JBuilder 2007 web services features allow you to quickly design, deploy, run, and test a web service.

In This Section

- **Activating the Web Services Explorer for Existing Components**
  Describes how to activate the Web Service Designer for existing components.

- **Configuring Your Workspace**
  Describes how to configure your workspace for Apache Axis and Tomcat.

- **Creating a Client Project**
  Describes how to create a client project to test your web service.

- **Creating a Client Web Service from a URL WSDL**
  Describes how to create a web services client from a WSDL URL location.

- **Creating a Dynamic Web Project**
  Describes how to create a dynamic web project for your web service.

- **Creating a Java Class for a Web Service**
  Describes how to create a Java class for a web service.

- **Creating a New Web Service**
  Describes how to add a new web service to your existing project.

- **Creating a New WSDL Web Service in the Web Services Explorer**
  Describes how to add a new WSDL web service to your project.

- **Creating a Web Service from a Java Project with a WSDL**
  Describes how to create a client web service from a Java project containing a WSDL.

- **Designing a Bottom-Up Web Service Using the Apache Axis Runtime**
  Describes how to design a bottom-up web service using Apace Axis and Tomcat.

- **Designing a Top-Down Web Service Using the Apache Axis Runtime**
  Describes how to design a top-down web service using Apache Axis and Tomcat.

- **Exporting a Java Class to a Web Service**
  Describes how to export a class to a web service.

- **Opening the Web Services Explorer**
  Describes how to open the Web Services Explorer.

- **Running a Web Service**
  Describes how to run your web service.

- **Setting Service Properties in the Web Services Explorer**
  Describes how to set service properties in the Web Services Explorer.

- **Setting WSDL Properties in the Web Services Explorer**
  Describes how to set WSDL properties in the Web Services Explorer.

- **Testing the Web Service with the Client**
  Describes how to test your web service with the Axis Admin console and the client project.

- **Working in the Web Services Explorer**
  Describes steps for working in the Web Services Explorer.
Activating the Web Services Explorer for Existing Components

Tip: These steps assume correctly configured runtime and server parameters. Links to topics detailing these steps are listed in the Related Procedures section of this topic.

The Web Services Explorer creates a design surface for visually creating and implementing web services in an existing Java class or WSDL. Use the following steps to activate the Web Services Explorer for existing components:

1. Open the desired dynamic web project containing the Java class or WSDL component.
2. If the file is a Java class, right click the component, select Web Services and click Create Web Services from Model in the drop down menu.
   If the file is a WSDL, right click the file in the Package Explorer, select Web Services and click Select WSDL on Diagram in the submenu.
3. To edit the element properties switch to the Modeling perspective:
   Select Window ▶ Open Perspective ▶ Modeling.
   Tip: Another way to display the Properties editor view for web services elements is to click on the element in the Web Services Diagram and select Window ▶ Show View ▶ Properties.
4. The Properties view is now open on the workbench.

Related Concepts
- Web Services Overview
- Runtime Servers

Related Tasks
- Setting Up a Runtime Server
- Opening the Web Services Explorer
- Working in the Web Services Explorer
- Setting Service Properties in the Web Services Explorer
- Setting WSDL Properties in the Web Services Explorer
Configuring Your Workspace

To build a bottom-up web service and run it in JBuilder 2007, you first need to configure the Apache Tomcat server and JRE. The Eclipse Web Tools Project (WTP) uses Apache Axis 1.2 for the web service runtime.

To add JDK 5.0 as the JRE

1. In Eclipse, open the Installed JREs page of the Preferences dialog box (Windows ➤ Preferences ➤ Java ➤ Installed JREs). You use this page to add Java runtime environments.
2. Click Add to display the Add JRE dialog box, where you add a JRE.
3. Leave the JRE type set to Standard VM.
4. In the JRE Name dialog box enter an identifying name for the JRE, such as JDK 5.0.
5. Choose the location of the JRE home folder in the JRE Home Directory field. Use the Browse button to browse to the location of a JDK 1.4.

   Note: This must be a full JDK, not just the JRE.

6. Enter any default VM arguments in the Default VM Arguments field.
7. Select the Use Default System Libraries option to use the default libraries.
8. Click OK when you're done.
9. Select the new JDK as the default in the Installed JREs list.

   This JRE will now be available in New Server Runtime dialog box where you configure Tomcat.

To setup Tomcat 5.5 as the server runtime

2. Extract the compressed files to a local folder.
3. In Eclipse, open the Installed Server Runtimes Environment page of the Preferences dialog box (Windows ➤ Preferences ➤ Server ➤ Installed Runtimes). You use this page to configure server runtimes.
4. Click Add to display the New Server Runtime dialog box, where you add a server.
5. Open the Apache node and select Apache Tomcat 5.5. Click Next.
6. Click the Browse button next to the Tomcat Installation Directory field to browse to the Tomcat 5.5 local folder.
7. Choose a JDK from the JRE drop-down list.

   Note: Choose a 1.4 version of the JDK.

8. Click Finish when you are done.

   Apache Tomcat 5.5 is added to the Installed Server Runtimes list.

9. Select Tomcat 5.5 as the default and click OK to save the server runtime configuration.

   The selected runtime is used when you create new projects.
Related Concepts

Web Services Overview

Related Tasks

Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Designing a Top-Down Web Service Using the Apache Axis Runtime
Creating a Client Project

To test your web service, you can create a web client or a Java utility client.

To create a web client project

1. Open the WebContent node of your project.
2. Right-click the WSDL document that was created when you ran the web service.
3. Select Web Services ➤ Create Client Project.
   The Create Client Project wizard opens.
4. Verify the server. You can click the Edit button to change the selected server.
5. In the Client Project Type drop-down list, make sure that Dynamic Web Project is selected.
6. Change the default name of the client project in the Client Project field, if needed.
7. Click Finish to create the client project.

A new dynamic web project, that hosts the client project, is created. Generated files are placed in the / Generated Source/ folder of the client project. A JUnit test file is created. Do not change this test case directly. To update the test case, update the JUnit subclass that is written to the client project /src/ folder. If you must change the test case and want to save your changes, you can set the WSDL Test Case Overwrite property to false.

Related Concepts
   Web Services Overview

Related Tasks
   Designing a Bottom-Up Web Service Using the Apache Axis Runtime
   Setting WSDL Properties in the Web Services Explorer
Creating a Client Web Service from a URL WSDL

With a dynamic web project in place, you can create a client web service from a WSDL at a URL location.

To create a dynamic web client project from a WSDL URL location

1. Right-click the dynamic web project node and choose New ► Other ► Web Services ► Web Service Client From URL.
   The Add Web Service From URL dialog box is opened.

2. Verify the server runtime. If it is incorrect, click the Edit button to select the correct runtime.

3. Choose Dynamic Web Project from the Client Project Type drop-down list.

4. Enter the name of the client project in the Client Project field. The name defaults to URLClient.

5. Enter the WSDL location in the WSDL Location field. The path must point to a URL location. The filename must end in .wsdl.

6. Click Finish when you're done.

A new client project is created. Generated files are placed in the project's Generated_Source folder. A JUnit test file is created. Do not change this test case directly. To update the test case, update the JUnit subclass that is written to the client project src folder. If you must change the test case, you can set the WSDL Test Case Overwrite property to false.

Related Concepts
   Web Services Overview

Related Tasks
   Designing a Top-Down Web Service Using the Apache Axis Runtime
Creating a Dynamic Web Project

A web service is hosted in a dynamic web project.

To create a dynamic web project

1. Create a new project (File ▶ New ▶ Project).
   The New Project wizard is displayed.
2. Open the Web node in the New Project wizard, and choose Dynamic Web Project. Click Next.
   The New Dynamic Web Project wizard is displayed.
3. Enter the project name in the Project Name field.
4. To place the project in the default workspace, select Use Defaults. To place the project in a different workspace, turn off Use Defaults and click the Browse button to browse to the workspace.
5. Apache Tomcat 5.5, the default server, is displayed in the Target Runtime drop down list. If it is not selected, select it from the list.
   Note: Do not select Add Project to EAR.
6. Click Finish to create the project.
   The new project is created in the Dynamic Web Projects node of the Project Explorer.

Related Concepts

Web Services Overview
Creating a New Web Service

Related Tasks

Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Designing a Top-Down Web Service Using the Apache Axis Runtime
Creating a Java Class for a Web Service

To create a bottom-up web service, your dynamic web project needs to contain a Java class in the project `/src/` folder.

**To create a Java class for a web service**

1. Open the Project Explorer (Window ➤ Show View ➤ Project Explorer) and open the Dynamic Web Projects node.
2. Right-click the project node and choose New ➤ Other ➤ Class.
   The New Java Class wizard is displayed.
3. Enter the name of the class in the Name field. You can leave all other fields at the default settings.
   **Note:** Eclipse does not recommend that you use the default package. Enter a package name in the Package field.
   The new class is opened in the source code editor.
4. Click Finish when you are done.

Add methods that can be exported to a web service. Save the class.

**Related Concepts**

Web Services Overview

**Related Tasks**

Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Creating a New Web Service

A single dynamic web project can contain multiple Java web services.

To add a Java web service to your project

1. Open a dynamic web project, by selecting the project from the Project Explorer window at the left of the J2EE perspective (Window Open Perspective Other J2EE).
2. Open the Web Services Explorer
3. Open the Web Services palette.
4. Click the Java web services icon.

A Java web service representation is displayed on the design surface. Open the Properties view (Window Show View Properties) to set service properties. If the runtime and server are already configured, the service is immediately runnable.

Related Concepts

Web Services Overview

Related Tasks

Working in the Web Services Explorer
Setting Service Properties in the Web Services Explorer
Creating a New WSDL Web Service in the Web Services Explorer

A single dynamic web project can contain multiple WSDL web services.

To add a WSDL web service to your project

1. Open a dynamic web project.
2. Open the Web Services Explorer.
3. Open the Web Services palette.
4. Click the WSDL web services icon.

A WSDL web service representation is displayed on the design surface. Open the Properties view (Window ► Show View ► Properties) to set WSDL properties. If the runtime and server are already configured, the client is immediately runnable.

Related Concepts

Web Services Overview

Related Tasks

Working in the Web Services Explorer
Setting WSDL Properties in the Web Services Explorer
Creating a Web Service from a Java Project with a WSDL

You use the Convert Into Web Services Client Project wizard to convert a WSDL in a Java project into a client web service.

**Note:** The WSDL does not have to be contained in a dynamic web project.

**To create a web service from a Java project containing a WSDL**

1. Right-click the WSDL file in the Java project and choose Web Services ➤ Convert Into Client Project. The Convert Into Web Services Client Project wizard is displayed.
2. Verify the server runtime. If it is incorrect, click the Edit button to select the correct runtime.
3. In the Client Project Type drop-down list, choose the type of client project you want to create, either Dynamic Web Project or Java Utility Project.
4. Click Finish when you're done.

A new client project is created. Generated files are placed in the project's Generated_Source folder. A JUnit test file is created. Do not change this test case directly. To update the test case, update the JUnit subclass that is written to the client project src folder. If you must change the test case, you can set the WSDL Test Case Overwrite property to false.

If the client project is a Java project, generated files are also placed in the /Generated_Source/ folder and a JUnit test file is also created. However, because there is no WebContent node in the project, the WSDL file is placed in the root of the /src/ folder. The META-INF folder is also placed in the /src/ folder.

**Related Concepts**

- [Web Services Overview](#)

**Related Tasks**

- [Designing a Top-Down Web Service Using the Apache Axis Runtime](#)
Designing a Bottom-Up Web Service Using the Apache Axis Runtime

A bottom-up web service is a web service that is designed from a Java class. This procedure outlines the steps for creating a bottom-up web service using Apache Axis and Tomcat.

- Apache Axis is an open source implementation of Simple Object Access Protocol (SOAP), an XML-based protocol for exchanging information.
- Apache Jakarta Tomcat provides a servlet container for your web service.

To design a web service from a Java class using Axis and Tomcat

1. Configure your workspace.
   Configuring Your Workspace
2. Create a dynamic web project.
   Creating a Dynamic Web Project
3. Create a Java class for the web service.
   Creating a Java Class for a Web Service
4. Export the class to a web service.
   Exporting a Java Class to a Web Service
5. Set service properties.
   Setting Service Properties in the Web Services Explorer
6. Run your web service.
   Running a Web Service
7. Create a client project to test your web service.
   Creating a Client Project
8. Set WSDL properties.
   Setting WSDL Properties in the Web Services Explorer
9. Test your web service.
   Testing the Web Service with the Client

Related Concepts

- Web Services Overview
- Web Services Explorer Overview

Related Tasks

- Working in the Web Services Explorer
Configuring Your Workspace

To build a bottom-up web service and run it in JBuilder 2007, you first need to configure the Apache Tomcat server and JRE. The Eclipse Web Tools Project (WTP) uses Apache Axis 1.2 for the web service runtime.

To add JDK 5.0 as the JRE

1. In Eclipse, open the Installed JREs page of the Preferences dialog box (Windows ➤ Preferences ➤ Java ➤ Installed JREs). You use this page to add Java runtime environments.
2. Click Add to display the Add JRE dialog box, where you add a JRE.
3. Leave the JRE type set to Standard VM.
4. In the JRE Name dialog box enter an identifying name for the JRE, such as JDK 5.0.
5. Choose the location of the JRE home folder in the JRE Home Directory field. Use the Browse button to browse to the location of a JDK 1.4.
   
   **Note:** This must be a full JDK, not just the JRE.
6. Enter any default VM arguments in the Default VM Arguments field.
7. Select the Use Default System Libraries option to use the default libraries.
8. Click OK when you’re done.
9. Select the new JDK as the default in the Installed JREs list.
   
   This JRE will now be available in New Server Runtime dialog box where you configure Tomcat.

To setup Tomcat 5.5 as the server runtime

2. Extract the compressed files to a local folder.
3. In Eclipse, open the Installed Server Runtimes Environment page of the Preferences dialog box (Windows ➤ Preferences ➤ Server ➤ Installed Runtimes). You use this page to configure server runtimes.
4. Click Add to display the New Server Runtime dialog box, where you add a server.
5. Open the Apache node and select Apache Tomcat 5.5. Click Next.
6. Click the Browse button next to the Tomcat Installation Directory field to browse to the Tomcat 5.5 local folder.
7. Choose a JDK from the JRE drop-down list.
   
   **Note:** Choose a 1.4 version of the JDK.
8. Click Finish when you are done.
   
   Apache Tomcat 5.5 is added to the Installed Server Runtimes list.
9. Select Tomcat 5.5 as the default and click OK to save the server runtime configuration.
   
   The selected runtime is used when you create new projects.
Related Concepts

Web Services Overview

Related Tasks

Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Designing a Top-Down Web Service Using the Apache Axis Runtime
Creating a Dynamic Web Project

A web service is hosted in a dynamic web project.

To create a dynamic web project

1. Create a new project (File ➤ New ➤ Project). The New Project wizard is displayed.
2. Open the Web node in the New Project wizard, and choose Dynamic Web Project. Click Next. The New Dynamic Web Project wizard is displayed.
3. Enter the project name in the Project Name field.
4. To place the project in the default workspace, select Use Defaults. To place the project in a different workspace, turn off Use Defaults and click the Browse button to browse to the workspace.
5. Apache Tomcat 5.5, the default server, is displayed in the Target Runtime drop down list. If it is not selected, select it from the list.

   **Note:** Do not select Add Project to EAR.

6. Click Finish to create the project.

   The new project is created in the Dynamic Web Projects node of the Project Explorer.

Related Concepts

- Web Services Overview
- Creating a New Web Service

Related Tasks

- Designing a Bottom-Up Web Service Using the Apache Axis Runtime
- Designing a Top-Down Web Service Using the Apache Axis Runtime
Creating a Java Class for a Web Service

To create a bottom-up web service, your dynamic web project needs to contain a Java class in the project /src/ folder.

To create a Java class for a web service

1. Open the Project Explorer (Window ➤ Show View ➤ Project Explorer) and open the Dynamic Web Projects node.
2. Right-click the project node and choose New ➤ Other ➤ Class.
   The New Java Class wizard is displayed.
3. Enter the name of the class in the Name field. You can leave all other fields at the default settings.
   **Note:** Eclipse does not recommend that you use the default package. Enter a package name in the Package field.
   The new class is opened in the source code editor.
4. Click Finish when you are done.

Add methods that can be exported to a web service. Save the class.

Related Concepts

- Web Services Overview

Related Tasks

- Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Exporting a Java Class to a Web Service

Exporting a Java class to a web service opens the Web Services Explorer and makes the service immediately runnable.

To export a Java class to a web service

1. Expand the project src node so that you can see the class you just created.
2. Right-click the class.
3. Choose Web Services ➤ Create Web Services Model.

The Web Services Explorer opens and creates a service representation. The methods in the class are exposed as a web service. You can set properties to modify the service or WSDL file.

The Opening Diagram Progress dialog box is displayed. The Web Services Explorer is opened and a service representation is created. The methods in the class are exposed. A WSDL file is created in the WebContent node. You can set properties in the Web Services Explorer to modify the service.

Related Concepts

Web Services Overview

Related Tasks

Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Setting Service Properties in the Web Services Explorer

When you create a service in the Web Services Explorer, a service representation is created. You can open the Properties view to set service properties that control the Java2WSDL builder. Default property values are created based on the selected server and toolkit.

The generated WSDL contains both interface and implementation WSDL constructs.

To set server properties:

1. If the class representation is not displayed in the Web Services Explorer, right-click the Java class you want to export to a web service and choose Web Services ➤ Create Web Services Model.
2. Open the Properties view (Window ➤ Show View ➤ Properties ➤ Server properties).
   
   You can set the following properties:

   - **Binding name**: Fully-qualified name of client-side stub class that acts as a proxy for a remote web service.
   - **Deploy scope**: Defines how instances of the service are created. **Request** selects one instance per request. **Application** shares one instance among all requests. **Session** selects one instance per authenticated session.
   - **Extra classes**: Extra server classes.
   - **Location URL**: URL of the service.
   - **Port Type name**: Name to assign to the portType element in the generated WSDL file.
   - **Namespace options**: Namespace options.
   - **Service name**: A service interface that defines a get method for each port listed in the service element of the WSDL.
   - **Service style**: The binding style in the WSDL document. **rpc** assigns Remote Procedure Call as the binding style. This is the default. **document** assigns document as the binding style. Document services don't use encoding. **wrapped** assigns wrapped as the binding style. Wrapped services are a specialized form of document services, which unwrap document style data to individual parameters.
   - **SOAP action**: Assigns a SOAP action for the operation in the WSDL. **DEFAULT** causes the soap action to be set according to the operation's meta data. **OPERATION** assigns the operation name as the SOAP action for the operation in the WSDL. **NONE** does not assign a SOAP action. This allows the action to be provided in the operation descriptor at runtime.
   - **Type mapping version**: The type mapping version. Apache Axis 1.2 uses this setting internally to set up the default type mapping and the SOAP encoding type mappings. **1.1** chooses the default type mapping and no SOAP encoding. **1.2** chooses the default type mapping and SOAP encoding. **1.3** chooses the JAX-RPC 1.1 type mapping and SOAP encoding.
   - **Use**: The use of the service and the WSDL document. **literal** specifies that the XML Schema define the representation of the XML for the request. **encoded** specifies that SOAP encoding be specified in the generated WSDL.

Changes are applied to the WSDL file at the next build.

To set web service properties:

1. If the class representation is not displayed in the Web Services Explorer, right-click the Java class you want to export to a web service and choose Web Services ➤ Create Web Services Model.
2. Open the Properties view (Window ➤ Show View ➤ Properties ➤ Web service properties).
   
   You can set the following properties:
- **Allowed methods**: Methods to expose in the service and the WSDL.
- **Class or interface**: Name of the class to be exported as a web service.
- **Disallowed methods**: Methods to exclude from the service and the WSDL.
- **Display name**: Name of service to be displayed.
- **Enabled**: Checked if this web service is enabled.
- **Exclude package/class from tree**: The classes to exclude from the search tree when exporting data types and methods for the web service.
- **Implementation class**: Name of interface implementation class.
- **Include inherited methods**: Check to include inherited methods.
- **Service port**: Port number of this service.

**To set WSDL properties:**

1. If the class representation is not displayed in the Web Services Explorer, right-click the Java class you want to export to a web service and choose **Web Services ▶ Create Web Services Model**.
2. Open the **Properties** view (Window ▶ Show View ▶ Properties ▶ WSDL Properties).
   - You can set the following properties:
     - **Implementation namespace**: Source namespace for the implementation WSDL.
     - **Implementation WSDL file**: File name of the implementation WSDL.
     - **Import schema**: Schema to be imported.
     - **Include WSDL file**: WSDL file to be included.
     - **Location import URL**: URL of the service.
     - **Output**: Name of the input WSDL file. The output WSDL file contains all data from the input WSDL file plus any new constructs.
     - **Target Namespace**: Target namespace for the implementation WSDL.

**Related Concepts**
- [Web Services Overview](#)
- [Apache Axis Toolkit](#)

**Related Tasks**
- [Designing a Bottom-Up Web Service Using the Apache Axis Runtime](#)
- [Working in the Web Services Explorer](#)

**Related Reference**
- [Java2 WSDL Reference](#)
Running a Web Service

When you open your web service in the Web Services Explorer, the service is runnable.

To run a web service

1. Choose Run ➤ Run.
   The Run dialog box is displayed.

2. Expand the Web Service node in the Configurations list. Choose the name of your project.
   On the Run page, the web module is selected and the Launch URI field is set to the name of the runnable Axis servlet.

3. Click Run.
   The Run On Server dialog box displayed, where you select a server instance to run the web service on.

4. In the Select Server Type list, make sure Tomcat 4.1 Server is selected.

5. Select the Set Server As Project Default option so you will not be asked again to select a server for this project.

6. Click Finish.

The Servers view is opened. The Console view is also opened and displays Tomcat startup messages. The Web browser opens and shows the available services, including the service exposed in your project. You can select the WSDL link to view the WSDL document generated by Axis.

Note: You use the WSDL document to generate the client project.

Related Concepts

Web Services Overview

Related Tasks

Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Designing a Top-Down Web Service Using the Apache Axis Runtime
Creating a Client Project

To test your web service, you can create a web client or a Java utility client.

**To create a web client project**

1. Open the **WebContent** node of your project.
2. Right-click the WSDL document that was created when you ran the web service.
3. Select **Web Services ➤ Create Client Project**.
   - The **Create Client Project** wizard opens.
4. Verify the server. You can click the **Edit** button to change the selected server.
5. In the **Client Project Type** drop-down list, make sure that **Dynamic Web Project** is selected.
6. Change the default name of the client project in the **Client Project** field, if needed.
7. Click **Finish** to create the client project.

A new dynamic web project, that hosts the client project, is created. Generated files are placed in the `/Generated_Source/` folder of the client project. A **JUnit** test file is created. Do not change this test case directly. To update the test case, update the **JUnit** subclass that is written to the client project `/src/` folder. If you must change the test case and want to save your changes, you can set the WSDL **Test Case Overwrite** property to **false**.

**Related Concepts**

- [Web Services Overview](#)

**Related Tasks**

- [Designing a Bottom-Up Web Service Using the Apache Axis Runtime](#)
- [Setting WSDL Properties in the Web Services Explorer](#)
Setting WSDL Properties in the Web Services Explorer

When you create a service in the Web Services Explorer, a service representation is created. Open the Properties view to set properties for the WSDL2Java builder. Default property values are created based on the selected server and toolkit.

To set WSDL properties

1. If the WSDL representation is not displayed in the Web Services Explorer, right-click the WSDL you want to create a client project from and choose Web Services ▶ Create Client Project.

2. Open the Properties view (Window ▶ Show View ▶ Properties).

You can set the following properties:

- **All**: Set to true to generate code for all elements, even un-referenced ones. By default, WSDL2Java only generates code for those elements in the WSDL file that are referenced.
- **Debug**: Set to true to print debug information (the WSDL2Java symbol table).
- **Deploy Scope**: Defines how instances of the service are created. Select Request to select one instance per request. Select Application to share one instance among all requests. Select Session to select one instance per authenticated session.
- **HelperGen**: Set to true to generate all type mapping in separate helper classes.
- **No Imports** Set to true to ignore the import statements in the WSDL and the schema associated with the WSDL. Uses the immediate WSDL document
- **Output**: The root directory for all generated files.
- **OverwriteTypes**: Set to true to overwrite existing bean types of the same name with new Java source.
- **Package For All**: Set to true to write all generated files to same package (set with the Package Name property).
- **Package Name**: The package name for generated files.
- **Server Side**: Set to true to generate the server-side bindings for the web service.
- **Skeleton Deploy**: Set to true to generate the optional skeleton class to encapsulate an implementation for the server.
- **Test Case**: Set to true to generate a JUnit test case the first time you build the project. Any changes you make to the test case will never be overwritten when building, unless you set the Test Case Overwrite property.
- **Test Case Overwrite**: Set to true to overwrite the existing JUnit test case each time you build the project.
- **Timeout**: Timeout in seconds. The default is 0. Set to -1 to disable.
- **Typemapping Version**: The type mapping version. Apache Axis 1.2 uses this setting internally to set up the default type mapping and the SOAP encoding type mappings. Choose 1.1 to choose the default type mapping and no SOAP encoding. Choose 1.2 to choose the default type mapping and SOAP encoding. Choose 1.3 to choose the JAX-RPC 1.1 type mapping and SOAP encoding.
- **URL**: The location of the input WSDL file.
- **Verbose**: Set to true to display output from builder.
- **Wrapped**: Set to true to unwrap data to individual parameters. The WSDL must have wrapped specified as the Style property for this option to work.

Changes are applied to the WSDL file at the next build.
Related Concepts

Web Services Overview

Related Tasks

Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Designing a Top-Down Web Service Using the Apache Axis Runtime
Working in the Web Services Explorer

Related Reference

WSDL2Java Reference
Testing the Web Service with the Client

When you deploy your web service to the web services server, the Axis Admin console is displayed, where you validate your web service.

To run the client project

1. Choose Run ➤ Run.
   The Run dialog box is displayed.
2. Expand the Web Client node in the Configurations list and choose the client project.
   On the Run page, the web module is selected and the Launch URI field is set to launch the test JSP.
3. Click Run.
   The Run On Server dialog box displayed, where you select the server instance for your client project.
4. In the Select Server Type list, make sure Tomcat 4.1 Server is selected.
5. Select the Set Server As Project Default option so you will not be asked again to select a server for this project.
6. Click Finish.

The Servers view is opened. The Console view is also opened and displays Tomcat startup messages. The Web browser opens and shows the web client test project. Test a method by choosing it from the list on the left and clicking the Invoke button.

Related Concepts

- Web Services Overview

Related Tasks

- Designing a Bottom-Up Web Service Using the Apache Axis Runtime
- Designing a Top-Down Web Service Using the Apache Axis Runtime
Designing a Top-Down Web Service Using the Apache Axis Runtime

A top-down web service is a web service that is designed from a WSDL document. This procedure outlines the steps for creating a top-down service using Apache Axis and Tomcat.

- Apache Axis is an open source implementation of Simple Object Access Protocol (SOAP), an XML-based protocol for exchanging information.
- Apache Jakarta Tomcat provides a servlet container for your web service.

To design a web service from a WSDL document using Axis

1. Configure your workspace.
   Configuring Your Workspace

2. Create a dynamic web project.
   Creating a Dynamic Web Project

3. Create a client web service from a WSDL identified by its URL address.
   Note: You can also create a client web service from a Java project containing a WSDL. (The Java project does not have to be a dynamic web project.)
   Creating a Client Web Service from a URL WSDL

4. Set WSDL properties.
   Setting WSDL Properties in the Web Services Explorer

5. Run your web service.
   Running a Web Service

6. Test your web service.
   Testing the Web Service with the Client

Related Concepts
- Web Services Overview
- Web Services Explorer Overview

Related Tasks
- Designing a Bottom-Up Web Service Using the Apache Axis Runtime
- Creating a Web Service from a Java Project with a WSDL
Configuring Your Workspace

To build a bottom-up web service and run it in JBuilder 2007, you first need to configure the Apache Tomcat server and JRE. The Eclipse Web Tools Project (WTP) uses Apache Axis 1.2 for the web service runtime.

To add JDK 5.0 as the JRE

1. In Eclipse, open the Installed JREs page of the Preferences dialog box (Windows ➤ Preferences ➤ Java ➤ Installed JREs). You use this page to add Java runtime environments.
2. Click Add to display the Add JRE dialog box, where you add a JRE.
3. Leave the JRE type set to Standard VM.
4. In the JRE Name dialog box enter an identifying name for the JRE, such as JDK 5.0.
5. Choose the location of the JRE home folder in the JRE Home Directory field. Use the Browse button to browse to the location of a JDK 1.4.
   
   **Note:** This must be a full JDK, not just the JRE.
6. Enter any default VM arguments in the Default VM Arguments field.
7. Select the Use Default System Libraries option to use the default libraries.
8. Click OK when you're done.
9. Select the new JDK as the default in the Installed JREs list.

This JRE will now be available in New Server Runtime dialog box where you configure Tomcat.

To setup Tomcat 5.5 as the server runtime

2. Extract the compressed files to a local folder.
3. In Eclipse, open the Installed Server Runtimes Environment page of the Preferences dialog box (Windows ➤ Preferences ➤ Server ➤ Installed Runtimes). You use this page to configure server runtimes.
4. Click Add to display the New Server Runtime dialog box, where you add a server.
5. Open the Apache node and select Apache Tomcat 5.5. Click Next.
6. Click the Browse button next to the Tomcat Installation Directory field to browse to the Tomcat 5.5 local folder.
7. Choose a JDK from the JRE drop-down list.

   **Note:** Choose a 1.4 version of the JDK.
8. Click Finish when you are done.

   Apache Tomcat 5.5 is added to the Installed Server Runtimes list.
9. Select Tomcat 5.5 as the default and click OK to save the server runtime configuration.

   The selected runtime is used when you create new projects.
Related Concepts

- Web Services Overview

Related Tasks

- Designing a Bottom-Up Web Service Using the Apache Axis Runtime
- Designing a Top-Down Web Service Using the Apache Axis Runtime
Creating a Dynamic Web Project

A web service is hosted in a dynamic web project.

To create a dynamic web project

1. Create a new project (File ► New ► Project).
   The New Project wizard is displayed.
2. Open the Web node in the New Project wizard, and choose Dynamic Web Project. Click Next.
   The New Dynamic Web Project wizard is displayed.
3. Enter the project name in the Project Name field.
4. To place the project in the default workspace, select Use Defaults. To place the project in a different workspace, turn off Use Defaults and click the Browse button to browse to the workspace.
5. Apache Tomcat 5.5, the default server, is displayed in the Target Runtime drop down list. If it is not selected, select it from the list.
   **Note:** Do not select Add Project to EAR.
6. Click Finish to create the project.
   The new project is created in the Dynamic Web Projects node of the Project Explorer.

Related Concepts

- Web Services Overview
- Creating a New Web Service

Related Tasks

- Designing a Bottom-Up Web Service Using the Apache Axis Runtime
- Designing a Top-Down Web Service Using the Apache Axis Runtime
Creating a Client Web Service from a URL WSDL

With a dynamic web project in place, you can create a client web service from a WSDL at a URL location.

To create a dynamic web client project from a WSDL URL location

1. Right-click the dynamic web project node and choose New ➤ Other ➤ Web Services ➤ Web Service Client From URL.
   
   The Add Web Service From URL dialog box is opened.

2. Verify the server runtime. If incorrect, click the Edit button to select the correct runtime.

3. Choose Dynamic Web Project from the Client Project Type drop-down list.

4. Enter the name of the client project in the Client Project field. The name defaults to URLClient.

5. Enter the WSDL location in the WSDL Location field. The path must point to a URL location. The filename must end in .wsdl.

6. Click Finish when you're done.

A new client project is created. Generated files are placed in the project's Generated Source folder. A JUnit test file is created. Do not change this test case directly. To update the test case, update the JUnit subclass that is written to the client project src folder. If you must change the test case, you can set the WSDL Test Case Overwrite property to false.

Related Concepts

Web Services Overview

Related Tasks

Designing a Top-Down Web Service Using the Apache Axis Runtime
Setting WSDL Properties in the Web Services Explorer

When you create a service in the Web Services Explorer, a service representation is created. Open the Properties view to set properties for the WSDL2Java builder. Default property values are created based on the selected server and toolkit.

To set WSDL properties

1. If the WSDL representation is not displayed in the Web Services Explorer, right-click the WSDL you want to create a client project from and choose Web Services ➤ Create Client Project.
2. Open the Properties view (Window ➤ Show View ➤ Properties).

You can set the following properties:

- **All**: Set to true to generate code for all elements, even un-referenced ones. By default, WSDL2Java only generates code for those elements in the WSDL file that are referenced.
- **Debug**: Set to true to print debug information (the WSDL2Java symbol table).
- **Deploy Scope**: Defines how instances of the service are created. Select Request to select one instance per request. Select Application to share one instance among all requests. Select Session to select one instance per authenticated session.
- **HelperGen**: Set to true to generate all type mapping in separate helper classes.
- **No Imports**: Set to true to ignore the import statements in the WSDL and the schema associated with the WSDL. Uses the immediate WSDL document.
- **Output**: The root directory for all generated files.
- **OverwriteTypes**: Set to true to overwrite existing bean types of the same name with new Java source.
- **Package For All**: Set to true to write all generated files to same package (set with the Package Name property).
- **Package Name**: The package name for generated files.
- **Server Side**: Set to true to generate the server-side bindings for the web service.
- **Skeleton Deploy**: Set to true to generate the optional skeleton class to encapsulate an implementation for the server.
- **Test Case**: Set to true to generate a JUnit test case the first time you build the project. Any changes you make to the test case will never be overwritten when building, unless you set the Test Case Overwrite property.
- **Test Case Overwrite**: Set to true to overwrite the existing JUnit test case each time you build the project.
- **Timeout**: Timeout in seconds. The default is 0. Set to -1 to disable.
- **Typemapping Version**: The type mapping version. Apache Axis 1.2 uses this setting internally to set up the default type mapping and the SOAP encoding type mappings. Choose 1.1 to choose the default type mapping and no SOAP encoding. Choose 1.2 to choose the default type mapping and SOAP encoding. Choose 1.3 to choose the JAX-RPC 1.1 type mapping and SOAP encoding.
- **URL**: The location of the input WSDL file.
- **Verbose**: Set to true to display output from builder.
- **Wrapped**: Set to true to unwrap data to individual parameters. The WSDL must have wrapped specified as the Style property for this option to work.

Changes are applied to the WSDL file at the next build.
Related Concepts

  Web Services Overview

Related Tasks

  Designing a Bottom-Up Web Service Using the Apache Axis Runtime
  Designing a Top-Down Web Service Using the Apache Axis Runtime
  Working in the Web Services Explorer

Related Reference

  WSDL2Java Reference
Running a Web Service

When you open your web service in the Web Services Explorer, the service is runnable.

To run a web service

1. Choose Run ➤ Run.
   The Run dialog box is displayed.

2. Expand the Web Service node in the Configurations list. Choose the name of your project.
   On the Run page, the web module is selected and the Launch URI field is set to the name of the runnable Axis servlet.

3. Click Run.
   The Run On Server dialog box displayed, where you select a server instance to run the web service on.

4. In the Select Server Type list, make sure Tomcat 4.1 Server is selected.

5. Select the Set Server As Project Default option so you will not be asked again to select a server for this project.

6. Click Finish.

The Servers view is opened. The Console view is also opened and displays Tomcat startup messages. The Web browser opens and shows the available services, including the service exposed in your project. You can select the WSDL link to view the WSDL document generated by Axis.

Note: You use the WSDL document to generate the client project.

Related Concepts

   Web Services Overview

Related Tasks

   Designing a Bottom-Up Web Service Using the Apache Axis Runtime
   Designing a Top-Down Web Service Using the Apache Axis Runtime
Testing the Web Service with the Client

When you deploy your web service to the web services server, the Axis Admin console is displayed, where you validate your web service.

To run the client project

1. Choose Run ▶ Run.
   
   The Run dialog box is displayed.

2. Expand the Web Client node in the Configurations list and choose the client project.
   
   On the Run page, the web module is selected and the Launch URI field is set to launch the test JSP.

3. Click Run.
   
   The Run On Server dialog box displayed, where you select the server instance for your client project.

4. In the Select Server Type list, make sure Tomcat 4.1 Server is selected.

5. Select the Set Server As Project Default option so you will not be asked again to select a server for this project.

6. Click Finish.

The Servers view is opened. The Console view is also opened and displays Tomcat startup messages. The Web browser opens and shows the web client test project. Test a method by choosing it from the list on the left and clicking the Invoke button.

Related Concepts

   Web Services Overview

Related Tasks

   Designing a Bottom-Up Web Service Using the Apache Axis Runtime
   Designing a Top-Down Web Service Using the Apache Axis Runtime
Exporting a Java Class to a Web Service

Exporting a Java class to a web service opens the Web Services Explorer and makes the service immediately runnable.

To export a Java class to a web service

1. Expand the project src node so that you can see the class you just created.
2. Right-click the class.
3. Choose Web Services ➤ Create Web Services Model.

The Web Services Explorer opens and creates a service representation. The methods in the class are exposed as a web service. You can set properties to modify the service or WSDL file.

The Opening Diagram Progress dialog box is displayed. The Web Services Explorer is opened and a service representation is created. The methods in the class are exposed. A WSDL file is created in the WebContent node. You can set properties in the Web Services Explorer to modify the service.

Related Concepts

Web Services Overview

Related Tasks

Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Opening the Web Services Explorer

To open the Web Services Explorer for a Java web service

1. Open the dynamic web project containing the Java class you want to export to a web service.
2. Right-click the class and choose Web Services ➤ Create Web Service Model.

The Web Services Explorer is opened. Open the Properties view (Window ➤ Show View ➤ Properties) to set service properties. If the runtime and server are already configured, the service is immediately runnable.

Related Concepts

Web Services Overview

Related Tasks

Working in the Web Services Explorer
Setting Service Properties in the Web Services Explorer
Running a Web Service

When you open your web service in the Web Services Explorer, the service is runnable.

To run a web service

1. Choose Run ➤ Run. The Run dialog box is displayed.
2. Expand the Web Service node in the Configurations list. Choose the name of your project. On the Run page, the web module is selected and the Launch URI field is set to the name of the runnable Axis servlet.
3. Click Run. The Run On Server dialog box is displayed, where you select a server instance to run the web service on.
4. In the Select Server Type list, make sure Tomcat 4.1 Server is selected.
5. Select the Set Server As Project Default option so you will not be asked again to select a server for this project.
6. Click Finish.

The Servers view is opened. The Console view is also opened and displays Tomcat startup messages. The Web browser opens and shows the available services, including the service exposed in your project. You can select the WSDL link to view the WSDL document generated by Axis.

Note: You use the WSDL document to generate the client project.

Related Concepts

Web Services Overview

Related Tasks

Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Designing a Top-Down Web Service Using the Apache Axis Runtime
Setting Service Properties in the Web Services Explorer

When you create a service in the Web Services Explorer, a service representation is created. You can open the Properties view to set service properties that control the Java2WSDL builder. Default property values are created based on the selected server and toolkit.

The generated WSDL contains both interface and implementation WSDL constructs.

To set server properties:

1. If the class representation is not displayed in the Web Services Explorer, right-click the Java class you want to export to a web service and choose Web Services ➤ Create Web Services Model.
2. Open the Properties view (Window ➤ Show View ➤ Properties ➤ Server properties).
   You can set the following properties:
   - **Binding name**: Fully-qualified name of client-side stub class that acts as a proxy for a remote web service.
   - **Deploy scope**: Defines how instances of the service are created. Request selects one instance per request. Application shares one instance among all requests. Session selects one instance per authenticated session.
   - **Extra classes**: Extra server classes.
   - **Location URL**: URL of the service.
   - **Port Type name**: Name to assign to the portType element in the generated WSDL file.
   - **Namespace options**: Namespace options.
   - **Service name**: A service interface that defines a get method for each port listed in the service element of the WSDL.
   - **Service style**: The binding style in the WSDL document. rpc assigns Remote Procedure Call as the binding style. This is the default. document assigns document as the binding style. Document services don't use encoding. wrapped assigns wrapped as the binding style. Wrapped services are a specialized form of document services, which unwrap document style data to individual parameters.
   - **SOAP action**: Assigns a SOAP action for the operation in the WSDL. DEFAULT causes the soap action to be set according to the operation's meta data. OPERATION assigns the operation name as the SOAP action for the operation in the WSDL. NONE does not assign a SOAP action. This allows the action to be provided in the operation descriptor at runtime.
   - **Type mapping version**: The type mapping version. Apache Axis 1.2 uses this setting internally to set up the default type mapping and the SOAP encoding type mappings. 1.1 chooses the default type mapping and no SOAP encoding. 1.2 chooses the default type mapping and SOAP encoding. 1.3 chooses the JAX-RPC 1.1 type mapping and SOAP encoding.
   - **Use**: The use of the service and the WSDL document. literal specifies that the XML Schema define the representation of the XML for the request. encoded specifies that SOAP encoding be specified in the generated WSDL.

Changes are applied to the WSDL file at the next build.

To set web service properties:

1. If the class representation is not displayed in the Web Services Explorer, right-click the Java class you want to export to a web service and choose Web Services ➤ Create Web Services Model.
2. Open the Properties view (Window ➤ Show View ➤ Properties ➤ Web service properties).
   You can set the following properties:
- **Allowed methods**: Methods to expose in the service and the WSDL.
- **Class or interface**: Name of the class to be exported as a web service.
- **Disallowed methods**: Methods to exclude from the service and the WSDL.
- **Display name**: Name of service to be displayed.
- **Enabled**: Checked if this web service is enabled.
- **Exclude package/class from tree**: The classes to exclude from the search tree when exporting data types and methods for the web service.
- **Implementation class**: Name of interface implementation class.
- **Include inherited methods**: Check to include inherited methods.
- **Service port**: Port number of this service.

**To set WSDL properties:**

1. If the class representation is not displayed in the Web Services Explorer, right-click the Java class you want to export to a web service and choose **Web Services ➤ Create Web Services Model**.
2. Open the **Properties** view (**Window ➤ Show View ➤ Properties ➤ WSDL Properties**).
   You can set the following properties:
   - **Implementation namespace**: Source namespace for the implementation WSDL.
   - **Implementation WSDL file**: File name of the implementation WSDL.
   - **Import schema**: Schema to be imported.
   - **Include WSDL file**: WSDL file to be included.
   - **Location import URL**: URL of the service.
   - **Output**: Name of the input WSDL file. The output WSDL file contains all data from the input WSDL file plus any new constructs.
   - **Target Namespace**: Target namespace for the implementation WSDL.

**Related Concepts**
- [Web Services Overview](#)
- [Apache Axis Toolkit](#)

**Related Tasks**
- [Designing a Bottom-Up Web Service Using the Apache Axis Runtime](#)
- [Working in the Web Services Explorer](#)

**Related Reference**
- [Java2 WSDL Reference](#)
Setting WSDL Properties in the Web Services Explorer

When you create a service in the Web Services Explorer, a service representation is created. Open the Properties view to set properties for the WSDL2Java builder. Default property values are created based on the selected server and toolkit.

To set WSDL properties

1. If the WSDL representation is not displayed in the Web Services Explorer, right-click the WSDL you want to create a client project from and choose Web Services ➤ Create Client Project.

2. Open the Properties view (Window ➤ Show View ➤ Properties).

You can set the following properties:

- **All**: Set to true to generate code for all elements, even un-referenced ones. By default, WSDL2Java only generates code for those elements in the WSDL file that are referenced.
- **Debug**: Set to true to print debug information (the WSDL2Java symbol table).
- **Deploy Scope**: Defines how instances of the service are created. Select Request to select one instance per request. Select Application to share one instance among all requests. Select Session to select one instance per authenticated session.
- **HelperGen**: Set to true to generate all type mapping in separate helper classes.
- **No Imports**: Set to true to ignore the import statements in the WSDL and the schema associated with the WSDL. Uses the immediate WSDL document.
- **Output**: The root directory for all generated files.
- **OverwriteTypes**: Set to true to overwrite existing bean types of the same name with new Java source.
- **Package For All**: Set to true to write all generated files to same package (set with the Package Name property).
- **Package Name**: The package name for generated files.
- **Server Side**: Set to true to generate the server-side bindings for the web service.
- **Skeleton Deploy**: Set to true to generate the optional skeleton class to encapsulate an implementation for the server.
- **Test Case**: Set to true to generate a JUnit test case the first time you build the project. Any changes you make to the test case will never be overwritten when building, unless you set the Test Case Overwrite property.
- **Test Case Overwrite**: Set to true to overwrite the existing JUnit test case each time you build the project.
- **Timeout**: Timeout in seconds. The default is 0. Set to -1 to disable.
- **Typemapping Version**: The type mapping version. Apache Axis 1.2 uses this setting internally to set up the default type mapping and the SOAP encoding type mappings. Choose 1.1 to choose the default type mapping and no SOAP encoding. Choose 1.2 to choose the default type mapping and SOAP encoding. Choose 1.3 to choose the JAX-RPC 1.1 type mapping and SOAP encoding.
- **URL**: The location of the input WSDL file.
- **Verbose**: Set to true to display output from builder.
- **Wrapped**: Set to true to unwrap data to individual parameters. The WSDL must have wrapped specified as the Style property for this option to work.

Changes are applied to the WSDL file at the next build.
Related Concepts

Web Services Overview

Related Tasks

Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Designing a Top-Down Web Service Using the Apache Axis Runtime
Working in the Web Services Explorer

Related Reference

WSDL2Java Reference
Testing the Web Service with the Client

When you deploy your web service to the web services server, the Axis Admin console is displayed, where you validate your web service.

To run the client project

1. Choose Run ▶ Run.
   The Run dialog box is displayed.
2. Expand the Web Client node in the Configurations list and choose the client project.
   On the Run page, the web module is selected and the Launch URI field is set to launch the test JSP.
3. Click Run.
   The Run On Server dialog box displayed, where you select the server instance for your client project.
4. In the Select Server Type list, make sure Tomcat 4.1 Server is selected.
5. Select the Set Server As Project Default option so you will not be asked again to select a server for this project.
6. Click Finish.

The Servers view is opened. The Console view is also opened and displays Tomcat startup messages. The Web browser opens and shows the web client test project. Test a method by choosing it from the list on the left and clicking the Invoke button.

Related Concepts

Web Services Overview

Related Tasks

Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Designing a Top-Down Web Service Using the Apache Axis Runtime
Working in the Web Services Explorer

To work in the Web Services Explorer

1. Open the Web Services Explorer.
   - Opening the Web Services Explorer
2. Activate the Web Services Explorer for existing components.
   - Activating the Web Services Explorer for Existing Components
3. Create a new web service.
   - Creating a New Web Service
4. Create a new web service.
   - Creating a New WSDL Web Service in the Web Services Explorer
5. Set service properties.
   - Setting Service Properties in the Web Services Explorer
6. Set WSDL options.
   - Setting WSDL Properties in the Web Services Explorer

Related Concepts

- Web Services Overview
- Web Services Explorer Overview
Opening the Web Services Explorer

To open the Web Services Explorer for a Java web service

1. Open the dynamic web project containing the Java class you want to export to a web service.
2. Right-click the class and choose Web Services ▶ Create Web Service Model.

The Web Services Explorer is opened. Open the Properties view (Window ▶ Show View ▶ Properties) to set service properties. If the runtime and server are already configured, the service is immediately runnable.

Related Concepts

Web Services Overview

Related Tasks

Working in the Web Services Explorer
Setting Service Properties in the Web Services Explorer
Activating the Web Services Explorer for Existing Components

Tip: These steps assume correctly configured runtime and server parameters. Links to topics detailing these steps are listed in the Related Procedures section of this topic.

The Web Services Explorer creates a design surface for visually creating and implementing web services in an existing Java class or WSDL. Use the following steps to activate the Web Services Explorer for existing components:

1. Open the desired dynamic web project containing the Java class or WSDL component.

2. If the file is a Java class, right click the component, select Web Services and click Create Web Services from Model in the drop down menu.
   If the file is a WSDL, right click the file in the Package Explorer, select Web Services and click Select WSDL on Diagram in the submenu.

3. To edit the element properties switch to the Modeling perspective:
   Select Window ➢ Open Perspective ➢ Modeling.
   Tip: Another way to display the Properties editor view for web services elements is to click on the element in the Web Services Diagram and select Window ➢ Show View ➢ Properties.

4. The Properties view is now open on the workbench.

Related Concepts
   Web Services Overview
   Runtime Servers

Related Tasks
   Setting Up a Runtime Server
   Opening the Web Services Explorer
   Working in the Web Services Explorer
   Setting Service Properties in the Web Services Explorer
   Setting WSDL Properties in the Web Services Explorer
Creating a New Web Service

A single dynamic web project can contain multiple Java web services.

To add a Java web service to your project

1. Open a dynamic web project, by selecting the project from the Project Explorer window at the left of the J2EE perspective (Window Open Perspective Other J2EE).
2. Open the Web Services Explorer.
3. Open the Web Services palette.
4. Click the Java web services icon.

A Java web service representation is displayed on the design surface. Open the Properties view (Window Show View Properties) to set service properties. If the runtime and server are already configured, the service is immediately runnable.

Related Concepts

Web Services Overview

Related Tasks

Working in the Web Services Explorer
Setting Service Properties in the Web Services Explorer
Creating a New WSDL Web Service in the Web Services Explorer

A single dynamic web project can contain multiple WSDL web services.

To add a WSDL web service to your project

1. Open a dynamic web project.
2. Open the Web Services Explorer.
3. Open the Web Services palette.
4. Click the WSDL web services icon.

A WSDL web service representation is displayed on the design surface. Open the Properties view (Window ► Show View ► Properties) to set WSDL properties. If the runtime and server are already configured, the client is immediately runnable.

Related Concepts

Web Services Overview

Related Tasks

Working in the Web Services Explorer
Setting WSDL Properties in the Web Services Explorer
Setting Service Properties in the Web Services Explorer

When you create a service in the Web Services Explorer, a service representation is created. You can open the Properties view to set service properties that control the Java2WSDL builder. Default property values are created based on the selected server and toolkit.

The generated WSDL contains both interface and implementation WSDL constructs.

To set server properties:

1. If the class representation is not displayed in the Web Services Explorer, right-click the Java class you want to export to a web service and choose Web Services ➤ Create Web Services Model.
2. Open the Properties view (Window ➤ Show View ➤ Properties ➤ Server properties).
   You can set the following properties:

   - **Binding name**: Fully-qualified name of client-side stub class that acts as a proxy for a remote web service.
   - **Deploy scope**: Defines how instances of the service are created. Request selects one instance per request. Application shares one instance among all requests. Session selects one instance per authenticated session.
   - **Extra classes**: Extra server classes.
   - **Location URL**: URL of the service.
   - **Port Type name**: Name to assign to the portType element in the generated WSDL file.
   - **Namespace options**: Namespace options.
   - **Service name**: A service interface that defines a get method for each port listed in the service element of the WSDL.
   - **Service style**: The binding style in the WSDL document. rpc assigns Remote Procedure Call as the binding style. This is the default. document assigns document as the binding style. Document services don't use encoding. wrapped assigns wrapped as the binding style. Wrapped services are a specialized form of document services, which unwrap document style data to individual parameters.
   - **SOAP action**: Assigns a SOAP action for the operation in the WSDL. DEFAULT causes the soap action to be set according to the operation's meta data. OPERATION assigns the operation name as the SOAP action for the operation in the WSDL. NONE does not assign a SOAP action. This allows the action to be provided in the operation descriptor at runtime.
   - **Type mapping version**: The type mapping version. Apache Axis 1.2 uses this setting internally to set up the default type mapping and the SOAP encoding type mappings. 1.1 chooses the default type mapping and no SOAP encoding. 1.2 chooses the default type mapping and SOAP encoding. 1.3 chooses the JAX-RPC 1.1 type mapping and SOAP encoding.
   - **Use**: The use of the service and the WSDL document. literal specifies that the XML Schema define the representation of the XML for the request. encoded specifies that SOAP encoding be specified in the generated WSDL.

Changes are applied to the WSDL file at the next build.

To set web service properties:

1. If the class representation is not displayed in the Web Services Explorer, right-click the Java class you want to export to a web service and choose Web Services ➤ Create Web Services Model.
2. Open the Properties view (Window ➤ Show View ➤ Properties ➤ Web service properties).
   You can set the following properties:
- **Allowed methods**: Methods to expose in the service and the WSDL.
- **Class or interface**: Name of the class to be exported as a web service.
- **Disallow methods**: Methods to exclude from the service and the WSDL.
- **Display name**: Name of service to be displayed.
- **Enabled**: Checked if this web service is enabled.
- **Exclude package/class from tree**: The classes to exclude from the search tree when exporting data types and methods for the web service.
- **Implementation class**: Name of interface implementation class.
- **Include inherited methods**: Check to include inherited methods.
- **Service port**: Port number of this service.

### To set WSDL properties:

1. If the class representation is not displayed in the Web Services Explorer, right-click the Java class you want to export to a web service and choose **Web Services ➤ Create Web Services Model**.
2. Open the **Properties** view (**Window ➤ Show View ➤ Properties ➤ WSDL Properties**).

   You can set the following properties:

   - **Implementation namespace**: Source namespace for the implementation WSDL.
   - **Implementation WSDL file**: File name of the implementation WSDL.
   - **Import schema**: Schema to be imported.
   - **Include WSDL file**: WSDL file to be included.
   - **Location import URL**: URL of the service.
   - **Output**: Name of the input WSDL file. The output WSDL file contains all data from the input WSDL file plus any new constructs.
   - **Target Namespace**: Target namespace for the implementation WSDL.

### Related Concepts

- [Web Services Overview](#)
- [Apache Axis Toolkit](#)

### Related Tasks

- [Designing a Bottom-Up Web Service Using the Apache Axis Runtime](#)
- [Working in the Web Services Explorer](#)

### Related Reference

- [Java2 WSDL Reference](#)
Setting WSDL Properties in the Web Services Explorer

When you create a service in the Web Services Explorer, a service representation is created. Open the Properties view to set properties for the WSDL2Java builder. Default property values are created based on the selected server and toolkit.

To set WSDL properties

1. If the WSDL representation is not displayed in the Web Services Explorer, right-click the WSDL you want to create a client project from and choose Web Services ➤ Create Client Project.
2. Open the Properties view (Window ➤ Show View ➤ Properties).

You can set the following properties:

- **All**: Set to true to generate code for all elements, even un-referenced ones. By default, WSDL2Java only generates code for those elements in the WSDL file that are referenced.
- **Debug**: Set to true to print debug information (the WSDL2Java symbol table).
- **Deploy Scope**: Defines how instances of the service are created. Select Request to select one instance per request. Select Application to share one instance among all requests. Select Session to select one instance per authenticated session.
- **HelperGen**: Set to true to generate all type mapping in separate helper classes.
- **No Imports** Set to true to ignore the import statements in the WSDL and the schema associated with the WSDL. Uses the immediate WSDL document
- **Output**: The root directory for all generated files.
- **OverwriteTypes**: Set to true to overwrite existing bean types of the same name with new Java source.
- **Package For All**: Set to true to write all generated files to same package (set with the Package Name property).
- **Package Name**: The package name for generated files.
- **Server Side**: Set to true to generate the server-side bindings for the web service.
- **Skeleton Deploy**: Set to true to generate the optional skeleton class to encapsulate an implementation for the server.
- **Test Case**: Set to true to generate a JUnit test case the first time you build the project. Any changes you make to the test case will never be overwritten when building, unless you set the Test Case Overwrite property.
- **Test Case Overwrite**: Set to true to overwrite the existing JUnit test case each time you build the project.
- **Timeout**: Timeout in seconds. The default is 0. Set to -1 to disable.
- **Typemapping Version**: The type mapping version. Apache Axis 1.2 uses this setting internally to set up the default type mapping and the SOAP encoding type mappings. Choose 1.1 to choose the default type mapping and no SOAP encoding. Choose 1.2 to choose the default type mapping and SOAP encoding. Choose 1.3 to choose the JAX-RPC 1.1 type mapping and SOAP encoding.
- **URL**: The location of the input WSDL file.
- **Verbose**: Set to true to display output from builder.
- **Wrapped**: Set to true to unwrap data to individual parameters. The WSDL must have wrapped specified as the Style property for this option to work.

Changes are applied to the WSDL file at the next build.
Related Concepts

Web Services Overview

Related Tasks

Designing a Bottom-Up Web Service Using the Apache Axis Runtime
Designing a Top-Down Web Service Using the Apache Axis Runtime
Working in the Web Services Explorer

Related Reference

WSDL2Java Reference
Web Applications

The Java EE platform provides a simple, unified standard for distributed applications through a component-based application model. Use the following links to learn how to create a Java web application with JBuilder 2007.

In This Section
- Creating a Web Application Project
  Describes steps to create a web application project in JBuilder 2007
- Enabling XDoclet
  Describes how to enable XDoclet.
Creating a Web Application Project

A web application includes dynamic web pages containing various types of markup language and generated by web components running in the web tier, and a web browser to render the pages received from the server. Use the following steps to get started creating a web application project in JBuilder 2007.

To create a new project

1. Select File ➤ File ➤ New ➤ Project.
2. Type Web in the Wizards text entry box (to shift focus to the Web folder).
3. Select the Web folder and click + to view the sub-folders.
4. Choose to create a Static or Dynamic web project and click Next.
5. Type a Project Name in the text entry field, allow the default Target Runtime and Configurations options and click Next.
6. Allow the default Project Facets and click Finish to complete setup.

Tip: To configure detailed web module parameters accept the default Project Facets and click Next configure the following:

For a Static web application set the desired Context Root and Web Content Folder name then click Finish.

For a Dynamic web application set the desired Context Root Content Directory and Java Source Directory then click Finish.

Related Concepts

Web Applications Overview
Java EE Applications Overview

Related Tasks

Setting Up a Runtime Server
Publishing a Java EE Application to a Server Runtime
Running an Application on a Runtime Server

Related Reference

Eclipse help topic “Server targeting for web applications”
Eclipse help topic “Web Projects”
Eclipse help topic “Creating a static web project”
Eclipse help topic “Dynamic web projects and applications”
Eclipse help topic “Web page design”
Enabling XDoclet

Many Java EE applications require XDoclet support. This section describes how to enable XDoclet support.

XDoclet 1.2.3 with support for JDK 5.0 ships with JBuilder 2007 and is available in the JBuilder 2007 Eclipse plugins directory.

To enable XDoclet support:

1. Select Window ▶ Preferences ▶ XDoclet.
2. In the Set XDoclet Runtime Preferences dialog, check the Enable XDoclet Builder box to enable XDoclet support. Specify the home directory in XDoclet Home field. Select the appropriate version in the Version dropdown menu.
3. Click Apply and click OK.
4. You may also need to select Window ▶ Preferences ▶ XDoclet ▶ ejbdoclet/webdoclet options.
5. In the ejbdoclet or webdoclet dialogs, check the applicable tasks and servers.
6. Click Restore Defaults to restore default settings or Apply to apply the designated settings. Click OK.

Related Concepts

Java EE Applications Overview
Creating a Java EE Project

Related Tasks

Setting Up a Runtime Server
Modeling Applications

Borland's Together modeling system allows you to create a visual model as you develop Java database applications. InterBase and JDataStore database systems are included as part of the development environment.

In This Section

- Create a Dynamic Web Java Persistence API (JPA) Modeling Project
  Steps to create a Dynamic Web JPA modeling project.
- Create a Java Persistence API (JPA) Modeling Project
  Steps to create a JPA modeling project.
- Create an EJB Modeling Project with XDoclet Annotations
  Create an EJB modeling with XDoclet Annotations.
- Creating a Java Modeling Project
  Describes how to create a Java Modeling project.
- Creating a Modeling Project
  Describes how to create a modeling project.
- Creating an Enterprise Java Bean (EJB) Modeling Project
  Describes how to create a new Enterprise Java Bean (EJB) Modeling project.
- EJB Modeling Project from a Java Project
  Steps to import an EJB modeling project from an existing Java modeling project.
- Importing a Java Project as a Java Modeling Project
  Import a Java project as a Java Modeling project.
- Importing a Modeling Project
  Describes how to import a modeling project.
- Importing an Enterprise Java Bean (EJB) Modeling Project
  Describes how to import an Enterprise Java Bean (EJB) Modeling project.
Create a Dynamic Web Java Persistence API (JPA) Modeling Project

Use the following steps to create a Dynamic Web JPA modeling project.

1 Select File ▶ New ▶ Project to invoke the New Project wizard.
2 In the Select a Wizard window, navigate to the JPA folder, select Dynamic Web JPA Modeling Project and click Next.

   Tip: Type JPA in the wizard text box to quickly navigate to the JPA folder.
3 Name the project and select the Hibernate or Toplink Persistence Manager. Activate the Add library to the class path checkbox, accept the default settings for the remaining parameters, and click Next.
4 Configure the following Persistence Unit settings:
   - Persistence Unit Name
   - Transaction Type
   - Database Type
   - Database Connection
   - Schema

   Tip: If an active connection is not already configured, click the Add Connection link to complete the task.

Click Finish.

Related Concepts
   - Java EE Applications Overview
   - Modeling Applications Overview
   - Runtime Servers

Related Tasks
   - Setting Up a Runtime Server

Related Reference
   - Hibernate Documentation
Create a Java Persistence API (JPA) Modeling Project

To create a Java Persistence API (JPA) modeling project.

1. Select File ▶ New ▶ Project to invoke the New Project wizard.
2. In the Select a Wizard window navigate to the JPA folder, select JPA Modeling Project and click Next.
   Tip: Type JPA in the wizard text box to quickly navigate to the JPA folder.
3. Name the project and select the Hibernate or Toplink Persistence Manager.
   Activate the Add library to the class path checkbox, accept the default settings for the remaining parameters, and click Next.
4. Configure the following Persistence Unit settings:
   - Persistence Unit Name
   - Transaction Type
   - Database Type
   - Database Connection
   - Schema
   Tip: If an active connection is not already configured, click the Add Connection link to complete the task.

Click Finish.

Related Concepts
- Modeling Applications Overview
- Java EE Applications Overview
- Runtime Servers

Related Tasks
- Setting Up a Runtime Server

Related Reference
- Hibernate Documentation
Create an EJB Modeling Project with XDoclet Annotations

The Create an EJB Modeling Project with XDoclet Annotations wizard converts an existing Web Tools Platform (WTP) EJB project to an EJB modeling project using XDoclet annotations.

Warning: The WTP EJB project must exist in the current Workspace and XDoclet annotation support must be installed and configured to work with the Workbench.

To create an EJB modeling project with XDoclet Annotations

1. Select File ➤ New ➤ Project to invoke the New Project wizard.
2. In the Select a Wizard window navigate to the EJB folder and select EJB Modeling Project from an XDoclet Annotated WTP Project, and click Next.
3. A list of WTP EJB projects in the current Workspace is displayed.
   
   Note: Only WTP EJB projects (not EJB modeling projects) are displayed.
4. Activate the checkbox next to the desired EJB project and click Finish.

The WTP EJB project is converted to an EJB modeling project and EJB diagrams are created based on EJB source and XDoclet annotations in the WTP EJB project.

Related Concepts

JBuilder Project Migration Overview

Related Tasks

Setting Import Properties
Building an Imported Project
Enabling XDoclet

Related Reference

Creating Enterprise Beans with XDoclet Annotation Support
Creating a Java Modeling Project

This section describes how to create a Java Modeling project.

To create a new Java modeling project in JBuilder 2007:

1. Select File ➤ New ➤ Project.
2. Select Modeling Java Modeling Project from the list.
3. Enter a name for your new project.
4. Specify the Java build settings.
5. Click the Finish button.

Related Concepts

- Modeling Applications Overview

Related Tasks

- Creating a Modeling Project
- Importing a Modeling Project
- Importing a Java Project as a Java Modeling Project
Creating a Modeling Project

This section describes how to create a modeling project.

To create an empty modeling project in JBuilder 2007:

1. Select File ➤ New ➤ Project.
2. Select the appropriate type of modeling project from the list.
3. Enter a name for your new project.
4. Click the Finish button.

Related Concepts

- Modeling Applications Overview

Related Tasks

- Creating a Java Modeling Project
- Creating an Enterprise Java Bean (EJB) Modeling Project
- Create a Java Persistence API (JPA) Modeling Project
- Importing a Modeling Project
Creating an Enterprise Java Bean (EJB) Modeling Project

This section describes how to create a new Enterprise Java Bean (EJB) Modeling project

To create a new EJB modeling project in JBuilder 2007:

1. Select File New Project.
2. Select EJB EJB Modeling Project from the list.
3. Enter a name for the new project.
4. Select a target runtime and project configuration for the project.
5. Click the Finish button.

Related Concepts
   Modeling Applications Overview

Related Tasks
   Creating a Modeling Project
   Importing a Modeling Project
   Importing an Enterprise Java Bean (EJB) Modeling Project
EJB Modeling Project from a Java Project

Use the **EJB Modeling Project from a Java Project** wizard to import existing EJB sources and XML descriptors from an Eclipse Java project. The XML descriptors can be converted to **EJB 2.x XDoclet** annotations or to **EJB 3.0** annotations.

**Note:** The following steps assume a correctly configured web application server. For steps to install the **JBoss** web application server see **Related Procedures**.

**Tip:** A new EJB modeling project is created based on the source and descriptors from the Java project. The Java project can exist anywhere on the hard disk.

**Use the following steps to import an EJB modeling project from an existing Java project**

1. Place the XML descriptors in a folder named **META-INF** and make sure the folder is located in the project source directory.
2. Select **File ▶ New ▶ Project** to invoke the **New Project** wizard.
3. Navigate to the EJB folder, select **EJB Modeling Project from Java Project**, and click **Next**.
4. Select the desired Java project and click **Next**.
5. Name the new EJB modeling project.
6. Set the **Target Runtime** and click **Next**.

   **Warning:** Create a new runtime where an existing runtime is not already installed.

7. Set the EJB and Java versions for the converted project.
   - **XDoclet** annotations based on XML descriptors are generated for **EJB 2.1**.
   - **Java EE 5.0** annotations based on XML descriptors are generated for **EJB 3.0**.

   **Note:** The XML descriptors must be located in a folder named **META-INF**.

8. Click **Next**.
9. Accept or customize the remaining configuration settings, and click **Finish**.

**Related Concepts**

[JBUILDER Project Migration Overview](#)

**Related Tasks**

- [Setting Import Properties](#)
- [Building an Imported Project](#)
- Create an EJB Modeling Project with XDoclet Annotations
- [Setting Up a Runtime Server](#)
Importing a Java Project as a Java Modeling Project

Use these steps to import a Java project as a Java Modeling project.

To import a new Java modeling project from a Java project:

1. Select File ➤ New ➤ Project.
2. Select Modeling ➤ Java Modeling Projects from Java Projects from the list.
3. Specify the Java project to import.
4. Click the Finish button.

Related Concepts

- Modeling Applications Overview

Related Tasks

- Creating a Modeling Project
- Importing a Modeling Project
- Creating a Java Modeling Project
Importing a Modeling Project

This section describes how to import a modeling project.

To import a modeling project into JBuilder 2007:

1. Select File New Project.
2. Select the appropriate type of modeling project from the list.
3. Enter a name for the new project.
4. Click the Finish button.

Related Concepts

Modeling Applications Overview

Related Tasks

Importing a Java Project as a Java Modeling Project
Importing an Enterprise Java Bean (EJB) Modeling Project
Creating a Modeling Project
Importing an Enterprise Java Bean (EJB) Modeling Project

This section describes how to import an EJB modeling project from a Java project or from an Xdoclet-annotated WTP project.

To import an EJB modeling project from a Java project:

1. Select **File** ▶ **New** ▶ **Project**.
2. Select **EJB** ▶ **EJB Modeling Project from Java Project** from the list.
3. Enter a name for your new project.
4. Select the project to import.
5. Select a target runtime and project configuration for the project.
6. Click the **Finish** button.

To import an EJB modeling project from an Xdoclet-annotated WTP project:

1. Select **File** ▶ **New** ▶ **Project**.
2. Select **EJB** ▶ **EJB Modeling Project from Xdoclet annotated WTP project** from the list.
3. Enter a name for your new project.
4. Select the project to import.
5. Select a target runtime and project configuration for the project.
6. Click the **Finish** button.

Related Concepts

- [Modeling Applications Overview](#)

Related Tasks

- [Creating a Modeling Project](#)
- [Importing a Modeling Project](#)
Setting Up Database Connections

This section provides links to information about creating database applications with JBuilder 2007.

In This Section

- **Connecting to an InterBase Database**
  Describes how to create an InterBase connection from JBuilder 2007.

- **Connecting to JDataStore**
  Describes how to create a connection to JDataStore from JBuilder 2007.
Connecting to an InterBase Database

This topic describes how to create an InterBase connection from JBuilder 2007.

To connect to an InterBase database from JBuilder 2007:

1. In JBuilder 2007, open the Database Explorer view
2. Right click on Connection and select and select ‘New Connection’

    For detailed information about how to use InterBase, please see the InterBase documentation included with this product.

Related Concepts

Connecting to JDataStore
Connecting to JDataStore

This topic describes how to create a connection to JDataStore from JBuilder 2007. JDataStore 7 remote connections are supported in JBuilder 2007.

To connect to a JDataStore database in JBuilder 2007:

1. Launch the JDataStore server process (JDS_HOME\bin\jdsserver.exe)
2. In JBuilder 2007, open the Database Explorer view
3. Right click on Connection and select and select ‘New Connection’

Related Concepts

Connecting to an InterBase Database
ProjectAssist Procedures

This section describes how the ProjectAssist Administrator installs/assimilates and configures the ProjectAssist servers and repositories, and then adds projects and users.

In This Section

- Adding and Configuring Projects through ProjectAssist
  Describes how the ProjectAssist Administrator installs the ProjectAssist components on the server side, adds users and projects for TeamInsight members.

- Adding and Configuring TeamInsight Users
  Describes how the ProjectAssist Administrator installs the ProjectAssist components on the server side, adds users and projects for TeamInsight members.

- Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
  Describes how the ProjectAssist Administrator installs the ProjectAssist components on the server side, adds users and projects for TeamInsight members.

- Installing the ProjectAssist Components File
  Describes how the ProjectAssist Administrator installs the ProjectAssist component stack file on the server side.

- Installing the ProjectAssist Components File for JBuilder 2007 Enterprise Borland ALM Edition
  Describes how the ProjectAssist Administrator installs the ProjectAssist component stack file when including an installed StarTeam assimilation into the project stack.

- Mail Notification for TeamInsight Users
  Describes how the ProjectAssist Administrator installs the ProjectAssist components on the server side, adds users and projects for TeamInsight members.
Adding and Configuring Projects through ProjectAssist

(JBuilder 2007 Editions): Only the ProjectAssist Administrator can install the TeamInsight components and add team members and projects. After the TeamInsight component stack is installed on the ProjectAssist server, the Administrator can add projects and users for those projects.

To add projects in the ProjectAssist Configuration Editor

1. In the ProjectAssist Configuration Editor window, click on the Projects tab at the bottom of the window.

   **Note:** Only the ProjectAssist Administrator can add projects through the ProjectAssist Designer.

2. Any previously added projects will appear in the Project List. To create a project with the default values assigned in all the ProjectAssist server components, click Add. To clone the settings for any existing project, click the product name and then click Clone.

3. Complete the General, Project Content, Project Names and Paths, and other component-specific (such as Bugzilla, CVS, or StarTeam) information fields as appropriate, or accept the default values.

4. When done adding projects, you can add users to the project. See the following links for related information on adding and configuring TeamInsight users.

5. After all projects are added, click the Install Project Assist icon in the upper right of the workspace.

   **Note:** The ProjectAssist Administrator can add projects and users at any time after the initial JBuilder 2007 install.

Related Concepts

- ProjectAssist and TeamInsight Overview

Related Tasks

- Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
- Installing the ProjectAssist Components File
- Installing the ProjectAssist Components File for JBuilder 2007 Enterprise Borland ALM Edition
- Adding and Configuring TeamInsight Users
- Mail Notification for TeamInsight Users
- Configuring Your TeamInsight Client

Related Reference

- ProjectAssist Configuration Editor: Projects
Adding and Configuring TeamInsight Users
(JBuilder 2007 Editions): The ProjectAssist Administrator can install the ProjectAssist components on the server and add team members (users) and projects during the install or at a later time.

To add users in the ProjectAssist Designer

1 In the ProjectAssist Designer window, open the developer stack product file. Click the Users tab at the bottom of the window.

2 The Administrator created initially should already be in the User List. If you click the Administrator name, the General Information frame appears with user details and permissions for the Administrator.

3 To create a user with the default roles assigned to all the ProjectAssist components, click Add.

4 Replace the generic filler information with user-specific information. To change the user's role for any TeamInsight component, right-click on the component in the Roles list. The roles of Administrator, Developer, or No Access can be assigned for each user according to components. The default role assignment for all components (except MySQL) is Developer. The default for MySQL (used by the Bugzilla component) is No Access.

5 To create a user with the same assigned user roles as an already established user, click Clone.

6 To remove any user defined in steps 3–6, click Remove.

7 After all users are defined, click the Install ProjectAssist icon in the upper-right of the workspace to add these users to your project.

   **Note:** Users cannot be removed after they have been added with the Install ProjectAssist icon. Be sure the information is correct before installing.

8 If you receive the Passwords for Authorization dialog, complete the ProjectAssist Administrator password for each ProjectAssist components. Check the Use the same passwords box when appropriate to default all components to a single password. Click Install ProjectAssist.

9 The Preinstalled Component Scan dialog opens. Click the desired Scan Type. Click Finish if you checked Skip Scan or Next. A Disk Scan Paths screen opens. Click Scan to initiate a scan. A screen displays with the scan results.

Related Concepts

ProjectAssist and TeamInsight Overview

Related Tasks

Installing the ProjectAssist Components File
Adding and Configuring Projects through ProjectAssist
Mail Notification for TeamInsight Users
Configuring Your TeamInsight Client
Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)

(JBuilder 2007 Editions): Only the ProjectAssist Administrator can install the ProjectAssist components on the server and add team members and projects. ProjectAssist component installation can occur on local or remote servers. For example, all components can be installed on a local server. One or more components can be installed on one or more servers. Existing components on a remote server can be assimilated into the ProjectAssist installation. Note that the Liferay portlet can only be installed on a local server by the ProjectAssist Administrator.

**Note:** RECOMMENDED PRE-INSTALL SYSTEM CONFIGURATION

1. Windows XP Professional (TM) Service Pack 2 (or greater), or Windows Server 2003
2. 1 GB of system memory
3. Any virus software should be disabled during the ProjectAssist server install
4. Temporarily disable any firewall software.
5. Close any open applications prior to server install

Installing the JBuilder 2007 Software with ProjectAssist on a Server

1. Double click on the *jbinstall.exe* file to begin an installation process.
2. A JBuilder 2007 splash screen appears. It is followed by the product Introduction screen. Read the introductory text and click Next.
3. Read and accept the license agreement (select the radio button) and click Next.
4. Select Next for installation in the default folder.
5. Check the box on the ProjectAssist screen to install the ProjectAssist server on this machine. Click Next.
6. The Application Servers screen appears. This screen lists the runtime servers that are available for use in building web applications and web services. They are not related to your ProjectAssist server installation. The default is to not install these servers. Click Next.
7. Select Next to install JBuilder2007 with ProjectAssist to the default folder. You will be asked on the Choose Shortcut Folder screen for the locations of shortcut installs. Click Next to accept the default location.
8. Review the Pre-Installation Summary. Click Install.
9. Wait for the progress bar to indicate that the install is complete. Click Done to exit the installer. To open JBuilder 2007, go to Start ➤ All Programs ➤ JBuilder 2007 . . . ➤ JBuilder 2007.

Related Concepts

- ProjectAssist and TeamInsight Overview

Related Tasks

- Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
- Installing the ProjectAssist Components File
- Adding and Configuring Projects through ProjectAssist
- Adding and Configuring TeamInsight Users
- Mail Notification for TeamInsight Users
- Configuring Your TeamInsight Client
Installing the ProjectAssist Components File

(JBuilder 2007 Edition): Only the ProjectAssist Administrator should install the ProjectAssist server components and add team members and projects.

After you (ProjectAssist Administrator) have installed JBuilder 2007 with ProjectAssist enabled, you can install the ProjectAssist components stack on the server using either the JBuilder 2007 product interface or the ProjectAssist-only program interface.

Warning: Be sure to assign separate names to the workspaces if you will be using both the JBuilder 2007 and the ProjectAssist interfaces. You cannot use the default workspace for both.

To initiate a ProjectAssist stack file install

1. If you have not opened the JBuilder 2007 product, double click the JBuilder 2007 icon, or choose Start ► All Programs ► JBuilder 2007 ► JBuilder 2007.

2. In the Workspace Launcher, click OK to select the default Workspace, or specify a name. Note that the Workspace Launcher defaults to the workspace that was opened last. It is important that you create a separate workspace for the ProjectAssist stack, and that you pay attention to which workspace you are in when you log in to the ProjectAssist console.

3. Select the File ► New ► Other to open the New wizard search dialog.

4. In the New wizard search field, enter ProjectAssist.

5. Select ProjectAssist File and click Next.

6. You can now create your ProjectAssist stack file by following the steps in the To create a new ProjectAssist stack file subtask.

To initiate a ProjectAssist stack file install using ProjectAssist Designer Console


2. In the Workspace Launcher, click OK to select the default Workspace, or specify a name. Note that the Workspace Launcher defaults to the workspace that was opened last. It is important that you create a separate workspace for the ProjectAssist stack, and that you pay attention to which workspace you are in when you log in to the ProjectAssist console.


4. Navigate to the ProjectAssist file wizard in one of the following ways:

   - Select File ► New ► Other ► Team ► ProjectAssist File to open the Create ProjectAssist File window.
   - Select File ► New ► Other to open the New wizard search dialog. Type ProjectAssist in text search field. Select ProjectAssist File and click Next.

5. You can now create your ProjectAssist stack file by following the steps in the To create a new ProjectAssist stack file subtask.

To create a new ProjectAssist stack file

1. On the Create ProjectAssist file screen, accept the defaults or enter appropriate values in the following fields to define the file. Press Tab to move to the next field.
2 In the Initial Project area, choose the default values or enter other appropriate values in the Project Name and Descriptions fields.

3 In the Installation Directories area, choose the default values or enter other appropriate values in the ProjectAssist install directory and ProjectAssist data directory fields. Click Next.

4 The New ProjectAssist File: Select Stack Components dialog screen allows you to select the desired version control, continuous build, defect or change request tracking, and task provider software to configure in the ProjectAssist stack file.

The New ProjectAssist File: Select Stack Components dialog for JBuilder 2007 Enterprise allows the user to select:

- Version Control System has selections for CVS or Subversion. Only assimilation is allowed with CVS.
- Continuous Build System allows the installation or assimilation of Continuum.
- Defect Tracker System has a selection for Bugzilla as the team defect and change request tracking system to report, track and repair defects in products
- Task Provider has a selection for XPlanner as the task provider system.

When ProjectAssist stack component selection is complete, click Next.

5 The New ProjectAssist File preinstallation screen lists the components that ProjectAssist scans for before installation. Select the type of scan you want to perform. The default selection is a thorough system scan (disk, system path, services and running processes). Click Next.

Tip: This screen appears again later in the install; when it appears, press Next.

6 The Choose disk scan paths screen appears. The default path selected is the system root directory. Press Next to scan this directory. Click Finish when the scan is complete.

7 Your ProjectAssist stack file is created. This file is created with the .pacx extension. The ProjectAssist Designer window appears with a folder view for the .pacx stack file at the top. This view also has tabs at the bottom of the window for Stacks, Users, Projects and Source.

8 Select the Stacks tab to configure your ProjectAssist components for a local server install or to choose to assimilate an existing local or remote install (refer to the following subtask). Click on the Install ProjectAssist icon to install the ProjectAssist component software. To add users and projects to your ProjectAssist stack file, see Related Procedures.

To install ProjectAssist components on a remote servers or to assimilate existing component installations

1 Click the Stacks tab at the bottom of the ProjectAssist Designer. The Stacks page appears with the ProjectAssist component software listed.

2 Click the Subversion, CVS, Continuum, Bugzilla or XPlanner component name to display the product information screen for that component. Each screen has an Installation Location field. Install on local machine is selected by default.

Refer to an existing installation (local or remote) is selected by default for CVS and cannot be changed.
Note: The ProjectAssist Liferay component is always installed on the local server.

3 To assimilate an existing component installation, click Refer to an existing installation (local or remote). For StarTeam and CVS, this is the default and cannot be changed. A Connection Information area appears on the component page of the Stacks tab to specify configuration information for the assimilation.

Note: The Stacks tab in the IDE shows only the selection made previously in the New ProjectAssist File-Select Stack Component dialog page.

- The CVS component page has the Host field for entering the address of the host server on which the CVS installation resides. Enter the port number in the PServer port field, or leave the default value of 2401. Enter the CVS path on the host machine that points to the CVS repository root in the Repository path field. Enter the name of the read/write user account and password in the Admin username and Password fields. There is also a Repository Web View (if any) field where you can optionally enter a URL address for a CVS web view that will appear as a tab in the TeamInsight Viewer.

- The Subversion page has the ProjectAssist Administrator Admin username field defaulting to the Administrator's name. Enter the existing component installation location in the URL field. Enter the Subversion password in the Password field. You can enable the ability to add users remotely to Subversion by checking the enable box in the Remote Subversion User Management area.

- The Continuum page has the ProjectAssist Administrator Admin username field defaulting to the Administrator's name. Enter the existing component installation location in the URL field. Enter the Continuum password in the Password field. Enter the port number in the Continuum RPC port field.

- The Bugzilla page has the ProjectAssist Administrator Email Address field defaulting to the Administrator's E-mail address. Enter the existing component installation location in the URL field. Enter the Bugzilla password in the Password field.

- The XPlanner page has the ProjectAssist Administrator Admin username field defaulting to the Administrator's name. Enter the existing component installation location in the URL field. Enter the XPlanner password in the Password field.

On any of the component pages, click the Test Connection button to test the connection to the assimilated component.

4 Click on the Install Developer Stack icon in the upper-right of the page to install the ProjectAssist component software. To add users and projects to your ProjectAssist stack file, see Related Tasks.

Related Concepts

ProjectAssist and TeamInsight Overview

Related Tasks

Adding and Configuring Projects through ProjectAssist
Adding and Configuring TeamInsight Users
Mail Notification for TeamInsight Users
Configuring Your TeamInsight Client
Installing the ProjectAssist Components File for JBuilder 2007 Enterprise Borland ALM Edition

(JBuilder 2007 Borland ALM Edition): Only the ProjectAssist Administrator should install the ProjectAssist server components and add team members and projects.

After the ProjectAssist Administrator has installed JBuilder 2007 with ProjectAssist enabled, you can install the ProjectAssist components stack on the server using either the JBuilder 2007 product interface or the ProjectAssist-only program interface.

**Warning:** Be sure to assign separate names to the workspaces if you will be using both the JBuilder 2007 and the ProjectAssist interfaces. You cannot use the default workspace for both.

To initiate a ProjectAssist stack file install with StarTeam assimilation

1. If you have not opened the JBuilder 2007 product, double click the JBuilder 2007 icon, or choose Start ▶ All Programs ▶ JBuilder 2007 ▶ JBuilder 2007.

2. In the Workspace Launcher, click OK to select the default Workspace, or specify a name. Note that the Workspace Launcher defaults to the workspace that was opened last. It is important that you create a separate workspace for the ProjectAssist stack, and that you pay attention to which workspace you are in when you log in to the ProjectAssist console.

3. Select the File ▶ New ▶ Other to open the New wizard search dialog.

4. In the New wizard search field, enter ProjectAssist.

5. Select ProjectAssist File and click Next.

6. You can now create your ProjectAssist stack file by following the steps in the To create a new ProjectAssist stack file subtask.

To initiate a ProjectAssist stack file install using ProjectAssist Designer Console

1. Double-click the ProjectAssist icon to install the ProjectAssist Designer Console.

2. In the Workspace Launcher, click OK to select the default Workspace, or specify a name. Note that the Workspace Launcher defaults to the workspace that was opened last. It is important that you create a separate workspace for the ProjectAssist stack, and that you pay attention to which workspace you are in when you log in to the ProjectAssist console.


4. Navigate to the ProjectAssist file wizard in one of the following ways:

   - Select File ▶ New ▶ Other ▶ Team ▶ ProjectAssist File to open the Create ProjectAssist File window.
   - Select File ▶ New ▶ Other . . . ▶ Other to open the New wizard search dialog. Type ProjectAssist in wizard search field. Select ProjectAssist File and click Next.

5. You can now create your ProjectAssist stack file by following the steps in the To create a new ProjectAssist stack file subtask.
To create a new ProjectAssist stack file

1 On the New ProjectAssist File-Create ProjectAssist file dialog, accept the defaults or enter appropriate values in the following fields to define the file. Press Tab to move to the next field.

- **File Name** is the ProjectAssist stack file (extension .pacx) name.
- **Administrator Name** is the ProjectAssist Administrator name.
- **Initials** is the ProjectAssist Administrator initials.
- **E-mail** is the E-mail address of the ProjectAssist Administrator.
- **ID** is the ProjectAssist Administrator alias ID.

2 In the Initial Project area, choose the default values or enter other appropriate values in the **Project Name** and **Descriptions** fields.

3 In the Installation Directories area, choose the default values or enter other appropriate values in the **ProjectAssist install directory** and **ProjectAssist data directory** fields. Click Next.

4 The New ProjectAssist File-Select Stack Components dialog allows you to select the desired version control, continuous build, defect or change request tracking, and task provider software to configure in the ProjectAssist stack file.

The New ProjectAssist File: Select Stack Components dialog for the JBuilder 2007 Enterprise Borland ALM Edition allows the user to select:

- **Version Control System** has selections for CVS, StarTeam or Subversion. Only assimilation is allowed with CVS or StarTeam.
- **Continuous Build System** allows the installation or assimilation of Continuum.
- **Defect Tracker System** has selections for Bugzilla or StarTeam as the team defect and change request tracking system to report, track and repair defects in products.
- **Task Provider** has selections for StarTeam or XPlanner as the task provider system. Only assimilation is allowed with StarTeam.

When ProjectAssist stack component selection is complete, click Next.

5 The New ProjectAssist File preinstallation screen lists the components that ProjectAssist scans for before installation. Select the type of scan you want to perform. The default selection is a thorough system scan (disk, system path, services and running processes). Click Next.

   **Tip:** This screen appears again later in the install; when it appears, press Next.

6 The Choose disk scan paths screen appears. The default path selected is the system root directory. Press Next to scan this directory. Click Finish when the scan is complete.

7 Your ProjectAssist stack file is created. This file is created with the .pacx extension. The ProjectAssist Designer window appears with a folder view for the .pacx stack file at the top. This view also has tabs at the bottom of the window for Stacks, Users, Projects and Source.

8 Select the Stacks tab to configure your selected ProjectAssist components for a local server install, if possible, or to choose to assimilate an existing local or remote install (refer to the following subtask).

   **Note:** For StarTeam and CVS, only assimilation to an existing installation is permitted. The local server install option cannot be selected.

Click on the Install ProjectAssist icon to install the ProjectAssist component software. To add users and projects to your ProjectAssist stack file, see Related Procedures.
To install ProjectAssist components on a remote servers or to assimilate existing component installations

1 Click the Stacks tab at the bottom of the ProjectAssist Designer. The Stacks page appears with the ProjectAssist component software listed.

2 Click the stack component name to display the product information screen for that component. Each screen has an Installation Location field. Refer to an existing installation (local or remote) is selected by default for StarTeam and CVS and cannot be changed.

   **Note:** The ProjectAssist Liferay component is always installed on the local server.

3 To assimilate an existing component installation, click Refer to an existing installation (local or remote). For StarTeam and CVS, this is the default and cannot be changed. A Connection Information area appears on the component page on the Stacks tab to specify configuration information for the assimilation.

   **Note:** The Stacks tab in the IDE shows only the selection made previously in the New ProjectAssist File-Select Stack Component dialog page.

   - The StarTeam component page has the Server address field for entering the address of the server on which the StarTeam installation resides. The TCP/IP endpoint field defaults to 49201. Enter the StarTeam administrator's name and password in the Admin username and Password fields.

   - The CVS component page has the Host field for entering the address of the host server on which the CVS installation resides. Enter the port number in the PServer port field, or leave the default value of 2401. Enter the CVS path on the host machine that points to the CVS repository root in the Repository path field. Enter the name of the read/write user account and password in the Admin username and Password fields. There is also a Repository Web View (if any) field where you can optionally enter a URL address for a CVS web view that will appear as a tab in the TeamInsight Viewer.

   - The Subversion component page has the ProjectAssist Administrator Admin username field defaulting to the Administrator's name. Enter the existing component installation location in the URL field. Enter the Subversion password in the Password field. You can enable the ability to add users remotely to Subversion by checking the enable box in the Remote Subversion User Management area.

   - The Continuum component page has the ProjectAssist Administrator Admin username field defaulting to the Administrator's name. Enter the existing component installation location in the URL field. Enter the Continuum password in the Password field. Enter the port number in the Continuum RPC port field.

   - The Bugzilla component page has the ProjectAssist Administrator Email Address field defaulting to the Administrator's E-mail address. Enter the existing component installation location in the URL field. Enter the Bugzilla password in the Password field.

   - The XPlanner component page has the ProjectAssist Administrator Admin username field defaulting to the Administrator's name. Enter the existing component installation location in the URL field. Enter the XPlanner password in the Password field.

On any of the component pages, click the Test Connection button to test the connection to the assimilated component.

4 Click on the Install Developer Stack icon in the upper-right of the page to install the ProjectAssist component software. To add users and projects to your ProjectAssist stack file, see Related Tasks.
Related Concepts

- ProjectAssist and TeamInsight Overview

Related Tasks

- Adding and Configuring Projects through ProjectAssist
- Adding and Configuring TeamInsight Users
- Mail Notification for TeamInsight Users
- Configuring Your TeamInsight Client
Mail Notification for TeamInsight Users

(JBuilder 2007 Editions): The ProjectAssist Administrator installs the TeamInsight components and add team members and projects on the ProjectAssist server. After users are added, the Administrator can optionally notify them of their addition to a ProjectAssist server installation. Users can then configure their TeamInsight component access on their client machine.

To notify users by email notification

1. Once users have been added, the Administrator can notify the added users of their access to the TeamInsight views of the ProjectAssist components. Click the Send Mail Notifications icon in the upper right of the ProjectAssist Designer window. If mail has not been previous enabled, a dialog appears asking if you want to configure mail. Click Yes to enable mail.

2. The Mail Preferences window appears. Check Enable mail box to configure the mail notification system. Complete the following fields:
   - Sender name: enter the ProjectAssist Administrator's name.
   - Sender email address: enter the ProjectAssist Administrator's email address.
   - SMTP server address: enter the mail address for your Simple Mail Transport Protocol (SMTP) server.
   - As appropriate, check or uncheck the Use custom server port, Server requires authentication, and POP before SMTP required. If checked, enter additional information about the SMTP server port field, the authentication name and password, and the POP server host, name and password.
   - If you want to send a test message prior to the general user message, click the Send Text Message button.

3. Click Apply and then click OK to send the user notification message. A Mail Notification dialog appears. Click Yes to send to only previously unnotified users or No to send to all users (including previously notified users).

4. A copy of the default notification mail message appears with the list of notification recipients. You may send this message as is or change text content in the text box. Click OK to send this message.

   **Note:** The user mail notification message has the configuration file attached. The TeamInsight users must save and link to this file on their client machines to access the ProjectAssist components installed on the servers. They can also find the configuration information and file from the Liferay Portal page. See the following related information for more details.

**Related Concepts**

- ProjectAssist and TeamInsight Overview

**Related Tasks**

- Installing the ProjectAssist Components File
- Adding and Configuring Projects through ProjectAssist
- Adding and Configuring TeamInsight Users
- Configuring Your TeamInsight Client
- Opening the TeamInsight Viewer and the Liferay Portal
- Changing Your Passwords for the TeamInsight Tools

250
TeamInsight Procedures

This section describes how team members configure their client machines to enable the TeamInsight development tools. Team members use the TeamInsight tools and the TeamInsight Viewer in the IDE to create projects, assign tasks, monitor bugs, control source code versions, and integrate builds into the development process.

In This Section

**Adding Mylar Repositories for Bugzilla and XPlanner**
Describes how to add Mylar task repositories for Bugzilla and XPlanner, two of the TeamInsight tools.

**Adding Mylar Repositories for StarTeam Change Requests or Task Planning**
Describes how to add Mylar task repositories for a StarTeam installation assimilated through the ProjectAssist installation.

**Adding Team Members in XPlanner (Administrator Task)**
Describes how the ProjectAssist Administrator adds team members to a project in XPlanner, and describes the attributes that the Administrator can assign to team members.

**Administering the Liferay Portal**
Describes how the Liferay Administrator can customize the Liferay portal using the Home A1 Administrator page.

**Changing Your Passwords for the TeamInsight Tools**
Describes the location of the password change mechanisms in the Liferay project portal and the TeamInsight tools.

**Checking Out a Project, Making Changes, and Checking Your Changes Into the Repository**
Describes how to check out a local copy of the Subversion repository, update your working copy with changes from the repository, and commit your changes into the repository.

**Configuring Your TeamInsight Client**
Describes how team members configure their workstations as TeamInsight clients.

**Creating and Starting Project Iterations in XPlanner (Administrator Task)**
Describes how to create and start an iteration for a project in XPlanner.

**Creating or Generating Bug Reports in Bugzilla**
Describes how to log in, change the password and use TeamInsight Bugzilla to report, track and fix product software bugs.

**Logging in to TeamInsight Bugzilla**
Describes how to log in, change the password and use TeamInsight Bugzilla to report, track and fix product software bugs.

**Managing Bug Reports in Bugzilla**
Describes how to log in, change the password and use TeamInsight Bugzilla to report, track and fix product software bugs.

**Monitoring Iteration Metrics in XPlanner**
Describes how to produce metrics and charts to monitor projects in XPlanner.

**Moving or Continuing a Story or Task in XPlanner**
Describes how to move stories to a different iteration and how to move tasks to a different story.

**Opening the TeamInsight Viewer and the Liferay Portal**
Describes how team members can open the TeamInsight Viewer and their Liferay project portal.

**Planning a Product Feature: Creating a User Story in XPlanner**
Describes how to create a user story in XPlanner.
Planning Your Work: Creating Tasks in XPlanner
Describes how to create a user story in XPlanner, and describes the attributes that the Administrator must assign to each team member.

Querying Bugzilla for Bug Reports
Describes how to log in, change the password and use TeamInsight Bugzilla to report, track and fix product software bugs.

Tracking Your Time and Completing Tasks in XPlanner
Describes how to examine tasks, track your time devoted to tasks, and complete your tasks in XPlanner.

Using Continuum/Maven for Continuous Integration Builds
Describes how Maven/Continuum provides continuous builds.

Using the Subversion Viewer for Browsing the Project Repository
Describes how to use the Subversion Viewer to browse the Subversion repository. One of the TeamInsight tools is Sventon, a read-only repository browser.
Adding Mylar Repositories for Bugzilla and XPlanner

JBuilder 2007 enables you to include the Bugzilla repository bugs and XPlanner repository tasks in the Eclipse Task List view, and to use Mylar to define queries against those repositories. The Mylar plugin offers task-focused user capabilities for JIRA, Bugzilla, Trac, and XPlanner. After you activate a task, Mylar remembers the context of your subsequent work, such as the files associated with the active task. Later when you return to the task, the preserved context enables you to work more efficiently.

Using Mylar, you can:

- Connect to task- or bug-tracking repository
- Define a query against the repository so that bugs or tasks are represented as Mylar tasks in the development environment
- Define tasks related to the repository
- View task or bug reports locally or in an embedded browser
- Activate tasks and focus on the active task
- Save task context, including files and file hierarchy
- Work with tasks offline and resynchronize with the repository at a later time

Note: If you are using a source repository that supports Mylar (currently CVS and Subversion), and the Eclipse plugin for it, you can commit your source changes based upon a change context associated with a Mylar task. Mylar automatically creates a comment that includes the task description.

To use Mylar with a Bugzilla or XPlanner repository

1. Click Window ▶ Configure Mylar and select either the Bugzilla or the XPlanner repository.
2. On the Configure Mylar dialog box, enter your password for the repository you selected. Click OK.
   The Task List and the Task Repositories views open, displaying a repository entry, and your tasks or bugs query for the repository you selected.
3. On the Task Repositories view, you can verify your logon, if necessary, with the appropriate server by double-clicking the repository icon and clicking Validate Settings.
4. On the Task List view, right-click the repository icon to display the Mylar context menu. Either select Open to open the predefined query for the repository, or select New Query to create a new query for a selected repository.
5. If you chose to edit the existing query, on the Edit Repository Query dialog box, select the entities (bugs, tasks, user stories, or project iterations) for which you want to see information. Click Finish.
   If you select Tasks in the Grouping field, a single query node is created in the Task List view, with all the applicable tasks underneath it. If you select User Stories in the Grouping field, then you get a query node for each selected user story in the Task List view, and each of the query nodes has task children that are associated with the parent user story.
   Additionally, you can control the scope of the tasks that are created – if you select All in the Scope group, all tasks from selected XPlanner entities are added to the query results. If you select My, then only your own tasks are added to the query results.
   The Task List displays the selected tasks or bugs from the repository.
   If you choose to edit the Bugzilla query through the Mylar Connector view, you see a dialog with options similar to those you would get for a Bugzilla query through a web page view.
6. To open the task or bug in an editor window, double-click the item in the Task List.
If you are opening an XPlanner task, a detailed editor appears that allows you to change the task name, description, and estimated time. You can also switch to the Browser tab to see the native XPlanner task or the Bugzilla bug editing web page.

7 To activate a task, click the icon in the left-hand column of the Task List. To focus Mylar on the current task in the Task List, click the Focus on workweek button. To focus Mylar on the current task in the Package Explorer or the Outline View, click the Focus on workweek button.

8 To open a task or bug with a known ID, select Navigate ▶ Open Repository Task from the main Eclipse menu. Type in the ID of the task or bug you want information about in the dialog. You must select the repository associated with the task or bug ID. If you want the task/bug to get added to your Task List view, check the Add to Task List category field on the dialog and indicate the category where the task/bug should be added (Root is the default).

Related Concepts

- ProjectAssist and TeamInsight Overview
- Mylar Concepts

Related Tasks

- Configuring Your TeamInsight Client

Related Reference

- External Documentation for Mylar from Eclipse.org
- External Documentation about Mylar Connectors to Repositories
- External Article: Task-Focused Programming with Mylar
Adding Mylar Repositories for StarTeam Change Requests or Task Planning

JBuilder 2007 Enterprise BALM Edition) JBuilder 2007 enables you to add StarTeam repository change requests and StarTeam repository tasks to the Eclipse Task List view, and to use Mylar to define queries against those repositories. The Mylar plugin adds task-focused user capabilities to the IDE. After you activate a task, Mylar remembers the context of your subsequent work, such as the files associated with the active task. Later when you return to the task, the preserved context enables you to work more efficiently.

Using Mylar, you can:

- Connect to repositories for tasks or change requests
- Define a query against the repository so that change requests or tasks are represented as Mylar tasks in the development environment
- Define tasks related to the repository
- View task or change request reports locally or in an embedded browser
- Activate tasks and focus on the active task
- Save task context, including files and file hierarchy
- Work with tasks offline and resynchronize with the repository at a later time

To use Mylar with an assimilated StarTeam installation repository

1. Click Window ▶ Configure Mylar and select a project configuration that references a StarTeam repository.

2. On the Configure Mylar Repository for StarTeam :Change Requests/Tasks dialog box, enter your password for the repository you selected. Click Validate to validate the user name/password and click OK.

   The Task List and the Task Repositories views open, displaying a repository entry, and your tasks or change requests query for the repository you selected.

3. On the Task Repositories view, verify your logon, if necessary, with the appropriate server by double-clicking the repository icon. This opens the Task Repository Settings:StarTeam Repository Settings dialog. Click the Validate Settings button to validate the server and login settings for this repository.

4. A StarTeam repository can be defined to show tasks, change requests, or both entities through the Task Repository Settings:StarTeam Repository Settings dialog or the Add Task Repository:StarTeam Repository Settings dialog. Check the Change Request and/or the Task boxes in the StarTeam Repository Type area of the dialog page. Based upon this selection, either tasks and/or change requests will appear as selectable items in the New StarTeam Query dialog. If you select All my tasks and All my change requests in this dialog, you will see two queries created in the Task List, one for “my tasks” and one for “my change requests”.

5. On the Task List view, right-click the repository icon to display the Mylar context menu. Select New Query to create a new query for a selected repository. This opens the New Repository Query dialog. Click Next to go to New StarTeam Query dialog, which allows you to name the query, chose all or selected tasks and change requests, and the scope.

6. If you chose to edit the existing query, on the Edit Repository Query dialog box, select the entities you want to see (tasks, or change requests). Click Finish.

   If you select Tasks in the Type field, a single query node is created in the Task List view, with all the applicable tasks sublisted. If you select Change Requests in the Type field, a single query node is created in the Task List view, with all the applicable change requests sublisted. Selecting both generates two lists in the Task List view. (See item 4.)
Additionally, you can control the scope of the tasks that are created – if you select All in the **Scope** group, all tasks or change requests from selected StarTeam entities are added to the query results. If you select My, then only your own tasks or change requests are added to the query results.

The Task List displays the selected tasks or change requests from the repository.

7 To open a task or change request in an editor window, double-click the task/change request in the **Task List**. This opens a detailed **Edit Repository Query** dialog window that allows you to change the selected tasks or change requests in the query, and the type or scope of the query.

8 To activate a task or change request, click the icon in the left-hand column of the Task List. To focus Mylar on the current task in the Task List, click the **Focus on workweek** button. To focus Mylar on the current task in the Package Explorer or the Outline View, click the **Focus on workweek** button.

---

**Related Concepts**

- ProjectAssist and TeamInsight Overview
- Mylar Concepts

**Related Tasks**

- Adding Mylar Repositories for Bugzilla and XPlanner
- Configuring Your TeamInsight Client

**Related Reference**

- StarTeam Repository Settings
- New StarTeam Query
- External Documentation for Mylar from Eclipse.org
- External Documentation about Mylar Connectors to Repositories
- External Article: Task-Focused Programming with Mylar
Adding Team Members in XPlanner (Administrator Task)

Two administrators have overlapping functions in XPlanner:

The **ProjectAssist Administrator** adds users for each of the TeamInsight tools during installation. Similarly, when new users need to be added to XPlanner, the ProjectAssist Administrator uses the system console to add users.

The **XPlanner Administrator** can add people as team members for any specific project in XPlanner. However, if the XPlanner Administrator adds a new user to the list of People in XPlanner, the user is not thereby added to TeamInsight. This situation can cause problems with the TeamInsight project tools.

**Tip:** To maintain the consistency of the TeamInsight project, only the ProjectAssist Administrator should add users.

**To add a person in XPlanner (outside of TeamInsight)**

1. On the ProjectAssist system console, log on as Administrator.
2. Enter XPlanner by selecting **Window ▶ Open TeamAssist Viewer ▶ XPlanner**.
3. On the top (**XPlanner Projects**) page, click **People**.
4. On the **People** page, click **Add Person**.
5. On the **Create Profile** page:
   - Enter the user's name, such as Joe Bloggs.
   - Create a user ID – either a user number such as 183 or a user name such as jbloggs.
   - Enter the user's initials (such as jb), E-mail address, and phone number. XPlanner uses a person's initials to indicate the person associated with a given task or story. Phone number and E-mail address are contact information for the team.
   - The **Hide?** box controls whether the person is listed in the People list available to developers.
   - Enter and confirm a temporary password for the new user.
   - Select the appropriate role for the new user: None, Viewer, Editor, Admin.
   - Select the project
6. Click **Add**.

**To add a team member to a project in XPlanner**

1. Start XPlanner by selecting **Window ▶ Open TeamAssist Viewer ▶ XPlanner**.
2. On the **Project** page, click **People**.
3. On the **People on Project** page, do something else??
4. If the following statement is true, then this procedure is not necessary.

Team members listed in the **People** page are available to work on any project.
Related Concepts

XPlanner: Project and Team Management
Creating and Starting Project Iterations in XPlanner (Administrator Task)

Related Tasks

Planning a Product Feature: Creating a User Story in XPlanner
Planning Your Work: Creating Tasks in XPlanner
Administering the Liferay Portal

The Liferay Administrator can use the Admin tab in the Liferay portal to customize the portal to match the needs of the project team. The Admin tab is only displayed immediately after the initial logon by the Liferay Administrator.

There is also a Setup tab within a TeamInsight component’s portlet allows the Liferay Administrator to reconfigure that portlet. This can be useful for resetting passwords for portlets that were changed or to change other configuration settings that have been invalidated after the install.

Changes made through the Setup tab take effect immediately with no restart of the server needed. Some changes are localized to the portlet of a particular project (for example, the Build Status portlet setting that contains the ID used by Continuum to identify a particular project). However, most settings are common to all instances of a portlet regardless of the project (for example, the URL used to access the application).

Note: To configure the Liferay portal, you must be the Liferay Administrator. See your ProjectAssist Administrator for permissions.

To open the Liferay Administrator (Admin) page

1. Select Window ➤ OpenTeamInsight Viewer ➤ Liferay.
   The Liferay Portal opens, displaying the sign-in command in the upper right corner of the Liferay portal.
2. Click Sign-in to display the sign-in dialog.
3. Enter your user name (typically your email address) and your password. Then click Sign in.
   The Liferay portal displays the Admin page. This is the Administrator page, which is only available immediately after the Liferay Administrator logs on to Liferay.
4. On the Admin page, you can configure the Liferay portlet using the following tabs:
   - Server
   - Auto Deploy
   - Enterprise
   - Portlets
   - Users
   - Live Sessions
   - Default Groups and Roles
   - Reserved Users
   - Mail Host Names
   - Emails

To access the Liferay Administrator’s Setup tab

1. Select Window ➤ OpenTeamInsight Viewer ➤ Liferay.
   The Liferay Portal opens, displaying the sign-in command in the upper right corner of the Liferay portal.
2. Click Sign-in to display the sign-in dialog.
3. Enter your Administrator’s user name (typically your email address) and your password. Then click Sign in.
The Liferay portal displays the Admin page. Other tabs are shown for Sample Projects and Configurations. The Setup tab is in the user interface for the individual portlets that are provided through Liferay when you are logged on as the Liferay administrator.

4 Click on the Sample Projects or Configurations tab to see the individual portlets for the TeamInsight components. The Setup tab is on the individual portlet pages.

5 Change any field on the Setup tab as desired. Changes made through the Setup tab take effect immediately with no restart of the server needed.

Related Concepts

- ProjectAssist and TeamInsight Overview
- Liferay: The TeamInsight Project Portal

Related Tasks

- Adding and Configuring TeamInsight Users
- Configuring Your TeamInsight Client
- Changing Your Passwords for the TeamInsight Tools
Changing Your Passwords for the TeamInsight Tools

The first time you log on to the Liferay project portal and the TeamInsight tools, you use a universal, temporary password that you receive in E-mail from the ProjectAssist Administrator.

To change your password in each of the tools, navigate to the locations described in this topic.

To find the password change fields in the TeamInsight tools

1. Click **Window** ➤ **Open TeamInsight Viewer** ➤ **Open All**.
2. Navigate to the appropriate location in each of the TeamInsight tools:
   - **Bugzilla**: The Bugzilla home page contains both the change password and logon commands. You can search the Bugzilla database without logging on.
   - **Continuum**: Only the ProjectAssist Administrator can change passwords for Continuum.
   - **Liferay**: On the Liferay project portal, click **My Account**. Then click the **Password** tab.
   - **Subversion Viewer**: No password is required for the Subversion Viewer (read-only).
   - **Subversion Repository**: On the Liferay project portal, click the **Configuration** tab at the top of the page. (For a Subversion repository assimilated into JBuilder 2007, use the password mechanism of the original Subversion version control system.)
   - **XPlanner**: On any XPlanner page, click **Me**. On your personal profile page, click **Edit**.
3. Enter your new password in the appropriate location.

Related Concepts

- ProjectAssist and TeamInsight Overview
- Subversion: Source Code Repository

Related Tasks

- Opening the TeamInsight Viewer and the Liferay Portal
Checking Out a Project, Making Changes, and Checking Your Changes Into the Repository

Your typical work pattern with Subversion is **Edit-Update-Commit**. Start by checking out a local copy of the repository into your workspace. Edit the files in your workspace. Typically, you **update** your files (merge the changes from the repository into your working files). Finally, you **commit** your changes to the repository (check your files into the repository). An Edit-Update-Commit workflow is the way to synchronize the repository and your workspace.

**To check out a local copy of the project repository**

1. Click **Project ▶ Checkout Project** and select the repository.
   
   If the repository is not configured, select **Configure** and locate the TeamInsight.ticx file that defines your Subversion repository. Your ProjectAssist Administrator distributes the TeamInsight.ticx file.

2. The repository is displayed in the **Navigator** view.

3. Expand the tree structure to locate the files that you want to open. Double-click a file to open it in the **Editor**.

**To update your local working files**

1. In the **Navigator** view, right-click the file or files for which you want to update the local working copy.

2. Select **Update**.

   Subversion merges the changes from the repository into your selected files.

**To commit your local files to the repository**

1. In the **Navigator** view, right-click the file or files you want to commit to the repository.

2. Select **Commit**.

   Subversion checks your working copy into the repository, creating a new version.

For more information about using Subclipse (the Subversion plug-in for Eclipse), see the Subclipse online help inside the help for Eclipse.

**Tip:** To preview the changes between your local working copy and the files in the repository, you can perform a “diff” operation using the Subversion read-only browser. To examine differences between the local copy and the repository, use the **Synchronize** view.

**Related Concepts**

- Subversion: Source Code Repository

**Related Tasks**

- Using the Subversion Viewer for Browsing the Project Repository
Configuring Your TeamInsight Client

TeamInsight team members import a configuration file to their local workstations to configure and setup the TeamInsight tools. If the ProjectAssist Administrator changes the project configuration at a later time, team members must import a changed configuration file.

**Note:** Only the ProjectAssist Administrator can install the server-side software and distribute the configuration file to enable the TeamInsight tools. The TeamInsight tools can be used on the following platforms:

- Microsoft® Windows® XP Professional (SP2)
- Microsoft® Windows® Vista 32-bit
- Red Hat Enterprise Linux
- Macintosh® OS X

To configure the TeamInsight client and access the TeamInsight tools

1. After the ProjectAssist Administrator installs the server side, verify that the following commands are present:

   - Window ➤ Configure Mylar contains a Configure command and lists No installed configurations.
   - Window ➤ Open TeamInsight Viewer contains a Configure command and lists No installed configurations.
   - Project ➤ Checkout Project contains a Configure command and lists No installed configurations.

   If these commands are not present, your TeamInsight tools are not correctly installed. You might need to reinstall the software. See your Administrator for help.

2. Locate the **TeamInsight.ticx** file for your project and copy the file to your local system.

   Your Administrator sends you an email with an attached TeamInsight.ticx file. The email gives your user ID and temporary password, and it also includes the configuration file as an attachment.

   The TeamInsight.ticx file enables your access to all the TeamInsight tools, including the Subversion repository.

   **Note:** After you have configured TeamInsight for the first time, you can import a new TeamInsight.ticx file by opening the Configuration page at the top of the Liferay portal. The .ticx file available from the Configuration page provides a configuration specifically for the team member logged into the Liferay portal. Therefore, sending a copy of this configuration file to another team member might not be appropriate unless that team member has access to the same applications.

3. Click any one of the three Configure commands for the TeamInsight tools:

   - Window ➤ Configure Mylar ➤ Configure
   - Window ➤ Open TeamInsight Viewer ➤ Configure
   - Project ➤ Checkout Project ➤ Configure

4. On the **TeamInsight Configuration File** dialog box, navigate to the location of your project's .ticx file.

5. Click **Open**.

   JBuilder 2007 displays a message confirming that menu configurations for the three commands have been imported successfully. You are now ready to use the TeamInsight tools for software development.

   **Note:** If the confirmation message does not appear, or if an error is displayed, see your Administrator for help.
To verify that the TeamInsight client is correctly configured, click the following menus:

- **Window » Configure Mylar** lists Bugzilla, XPlanner, or StarTeam.
- **Window » Open TeamInsight Viewer** lists the TeamInsight web components that were selected during the ProjectAssist server installation (such as Bugzilla, Continuum, Liferay, Subversion Viewer, XPlanner, CVS, or Borland’s ALM StarTeam).
- **Project » Checkout Project** is present.

By following any of these menu paths, you should see the URLs of the servers for the various TeamInsight tools that were configured or assimilated (such as Bugzilla, XPlanner, Continuum, CVS, StarTeam, and so forth.)

**Note:** After installing and configuring the TeamInsight tools, open the TeamInsight Viewer, open each applicable TeamInsight tool, and change your temporary password in each of the tools.

### To Specify URL Favorite Links Inside TeamInsight Viewer

1. You can add URLs of your choice to the TeamInsight Viewer through the JBuilder 2007 **Window** menu.
2. Go to **Window » Open TeamInsight Viewer » Edit Favorite Links** to add any favorite URL links that will be accessible from inside your TeamInsight Viewer. A tab for each favorite link added appears at the bottom of the TeamInsight Viewer.
3. You can follow the **Window » Open TeamInsight Viewer » Edit Favorite Links** to edit or remove any favorite URLs from your TeamInsight Viewer.

### Delete Configuration through Window Menu Selection

1. You can delete a TeamInsight configuration through the JBuilder 2007 **Window** menu.
2. Go to **Window » Open TeamInsight Viewer » Delete Configuration** to remove an imported TeamInsight configuration.

### Related Concepts

- ProjectAssist and TeamInsight Overview
- Liferay: The TeamInsight Project Portal

### Related Tasks

- Opening the TeamInsight Viewer and the Liferay Portal
- Changing Your Passwords for the TeamInsight Tools
- Adding Mylar Repositories for Bugzilla and XPlanner
Creating and Starting Project Iterations in XPlanner (Administrator Task)

Any XPlanner user can create and start iterations for projects in XPlanner. Iterations are typically short, only a few weeks. Projects typically have only one iteration started at a time (the current iteration).

Warning: Do not create projects from inside XPlanner if you want the project to be connected to the TeamInsight tools. Only the ProjectAssist Administrator can create projects that share the TeamInsight tools.

To create an iteration in XPlanner

1. Enter XPlanner either by selecting Window ► Open TeamInsight Viewer ► XPlanner or by selecting Window ► Add Mylar Repository ► XPlanner.
2. On the Top (XPlanner Projects) page, click the appropriate project name.
3. On the Project page, click Create Iteration.
4. On the Create Iteration window, supply a Name for the iteration (such as Sprint 1 or Backlog), a Start Date, an End Date, and a Description of the iteration.
5. Click Create. To clear the fields on the Create Iteration window, click Reset.

To start an iteration in XPlanner

1. In XPlanner, navigate to the page of the specific iteration.
2. Click Start.

Team members listed in the People page are available to work on any project.

Related Concepts

XPlanner: Project and Team Management
Mylar Concepts

Related Tasks

Planning a Product Feature: Creating a User Story in XPlanner
Monitoring Iteration Metrics in XPlanner
Adding Mylar Repositories for Bugzilla and XPlanner
Creating or Generating Bug Reports in Bugzilla

The Bugzilla component of TeamInsight is loaded during the ProjectAssist server installation or when the Administrator adds a new project. The ProjectAssist Administrator initially creates the access for individual project team members to the TeamInsight Bugzilla tool and all users are assigned the same password, which should be changed by the user. Any user can file a Bugzilla report that can be viewed by the team.

To create a new bug report in Bugzilla

1. After you reach the Bugzilla Main Page window, click on the Enter a new bug report link or Actions ▶ New to generate a new bug/defect report.
2. Select the product to report the bug against in the Bugzilla Enter Bug page. Click on appropriate link to report the bug against that product:
3. Complete the requested information about your bug report. Refer to The Bugzilla Guide for further information on completing these fields. All the members of your TeamInsight group are listed in the bug notification message. You can assign this bug to the appropriate development and QA person. All the members of the team receive notification of the new bug.
4. Click Commit to commit the bug into the repository.

To generate a bug report from the error log

1. Bug reports can be created directly from the error log in Bugzilla. You may want to keep your error log open on the JBuilder 2007 main page. To open the error log on the main page, go to Window ▶ Show View ▶ Error Log,
2. With the error log open, you can right-click on any error and select Report as Bug.

Related Concepts

- Bugzilla: Defect Tracking System
- Mylar Concepts

Related Tasks

- Logging in to TeamInsight Bugzilla
- Querying Bugzilla for Bug Reports
- Managing Bug Reports in Bugzilla
- Adding Mylar Repositories for Bugzilla and XPlanner

Related Reference

- Bugzilla Resources and Documents
- The Bugzilla Guide
Logging in to TeamInsight Bugzilla

The Bugzilla component of TeamInsight is loaded during the ProjectAssist server installation or when the Administrator adds a new project. The ProjectAssist Administrator initially creates the access for individual project team members to the TeamInsight Bugzilla tool and all users are assigned the same password. You can search the Bugzilla database without logging on to Bugzilla.

To initially login to TeamInsight Bugzilla and change your password

1. Enter TeamInsight Bugzilla either by selecting Window ► Open TeamInsight Viewer ► Bugzilla or by selecting Window ► Add Mylar Repository ► Bugzilla. You can also select to load all TeamInsight components by selecting Open All in either one of these paths.
2. Select the Bugzilla TeamInsight Viewer by clicking on the Bugzilla tab at the bottom of the viewer.
3. Click on Actions ► Login to login in using your Administrator-assigned password.
4. After you reach the Bugzilla Main Page window, which shows the work flow for a bug report in Bugzilla, click on the Change password or user preferences to update to a more secure user password.
5. Enter the requested information to change your password in the Bugzilla User Preferences page on the Account Preferences tab. When done, click Submit Changes button,

Related Concepts
- Bugzilla: Defect Tracking System
- Mylar Concepts

Related Tasks
- Creating or Generating Bug Reports in Bugzilla
- Querying Bugzilla for Bug Reports
- Managing Bug Reports in Bugzilla
- Adding Mylar Repositories for Bugzilla and XPlanner

Related Reference
- Bugzilla Resources and Documents
- The Bugzilla Guide
Managing Bug Reports in Bugzilla

The Bugzilla component of TeamInsight allows the user to generate bug reports in views different from the standard bug report output. Reports can also be generated in graphical, tabular or chart views according to a variety of criteria.

To create bug report graphical and chart displays

1. Along with the standard bug list, Bugzilla can generate two additional views of the bugs. These view include reports and charts. Reports give different views of the current database state. Charts plot the changes in sets of bugs over a specified time.

2. After you reach the Bugzilla Main Page window, click on the Summary reports and charts link or Actions ▶ Reports, from either the main page or a bug list search result page, to generate an alternate bug report view or chart.

3. The Bugzilla Reporting and Charting Kitchen page opens. From this page, you can select 3 types of report views and 1 type of chart view.

4. If you are interested in generating report views, click on one of the following links:
   - **Search** takes you to the Advanced Search tab of the Bugzilla Query page. The generates the same report as a standard advanced search bug query.
   - **Tabular reports** generates tables of bugs counts. You choose one or more fields as your axes, and then refine the set of bugs by completing the remainder of the fields on the Bugzilla Generate Tabular Report form. Click on Generate Report to view the report. Once the report appears, you can switch between Bar, Line, Table and CSV displays by clicking on the appropriate line at the end of your report.
   - **Graphical reports** generates line graphs, bar and pie charts. You choose one or more fields as your axes, and then refine the set of bugs by completing the remainder of the fields on the Bugzilla Generate Graphical Report form. Click on Generate Report to view the report. Once the report appears, you can switch between Pie, Bar, Line, Table and CSV displays by clicking on the appropriate line at the end of your report.

5. Charts generate a view of the bug database state over time. If you are interested in generating chart views, click on the Old Charts link on the Bugzilla Reporting and Charting Kitchen page. The Bugzilla Bug Charts page opens. Select your product and one or more data sets that you want to chart. Click the Continue button and your resulting chart is displayed.

Related Concepts

- Bugzilla: Defect Tracking System

Related Tasks

- Logging in to TeamInsight Bugzilla
- Creating or Generating Bug Reports in Bugzilla
- Querying Bugzilla for Bug Reports

Related Reference

- Bugzilla Resources and Documents
- The Bugzilla Guide
Monitoring Iteration Metrics in XPlanner

Three XPlanner commands produce useful statistics about iterations: **Metrics**, **Charts**, and **Accuracy**.

To display statistics about an iteration

1. Enter XPlanner either by selecting **Window ➤ Open TeamInsight Viewer ➤ XPlanner** or by selecting **Window ➤ Add Mylar Repository ➤ XPlanner**.
2. Navigate to the iteration you want to monitor. This can be any iteration, started or not.
3. On the **Iteration** page, click one of the following:
   - **Metrics** to compare hours worked by team members, both solo and paired, as well as hours accepted by developers.
   - **Charts** to display graphs and pie charts. The graphs represent both iteration progress (hours completed over time) and iteration burn down (remaining hours over time). The pie charts represent progress by task and by hour.
   - **Accuracy** to display statistics about the accuracy of time estimates in the iteration.

Related Concepts
- [ProjectAssist and TeamInsight Overview](#)
- [XPlanner: Project and Team Management](#)
- [Mylar Concepts](#)

Related Tasks
- [Adding Mylar Repositories for Bugzilla and XPlanner](#)
- [Planning a Product Feature: Creating a User Story in XPlanner](#)
- [Adding Team Members in XPlanner (Administrator Task)](#)
- [Creating and Starting Project Iterations in XPlanner (Administrator Task)](#)
- [Planning Your Work: Creating Tasks in XPlanner](#)
- [Tracking Your Time and Completing Tasks in XPlanner](#)
- [Moving or Continuing a Story or Task in XPlanner](#)

Related Reference
- [XPlanner Documentation Available from XPlanner.org](#)
Moving or Continuing a Story or Task in XPlanner

If a story or task is not completed in the original iteration, you can either move the story to a different iteration or move the task to a different story.

To move or continue a story

1. Enter XPlanner either by selecting Window ➤ Open TeamInsight Viewer ➤ XPlanner or by selecting Window ➤ Add Mylar Repository ➤ XPlanner.
2. Navigate to the Iteration page.
3. Do either of the following:
   - Click the Move/Continue icon next to the story you want to move.
   - Click the ID of the story you want to move. Then on the Story page, click the Move/Continue command, located at the bottom of the screen.
4. On the Move/Continue Story page, click the drop-down list of iterations, and select the destination for the story.
5. Click Move or Continue to move the story to the selected iteration. (To cancel the move, click the browser's Back button.)

To move or continue a task

1. In XPlanner, navigate to the Story page.
2. Do either of the following:
   - Click the Move/Continue icon next to the task you want to move.
   - Select the ID of the task you want to move. Then on the Task page, click the Move/Continue command, located at the bottom of the screen.
3. On the Move/Continue Task page, click the drop-down list of stories, and select the destination for the task.
4. Click Move or Continue to move the task to the selected story. (To cancel the move, click your browser's Back button.)
Related Concepts

- ProjectAssist and TeamInsight Overview
- XPlanner: Project and Team Management
- Mylar Concepts

Related Tasks

- Adding Mylar Repositories for Bugzilla and XPlanner
- Adding Team Members in XPlanner (Administrator Task)
- Creating and Starting Project Iterations in XPlanner (Administrator Task)
- Planning a Product Feature: Creating a User Story in XPlanner
- Planning Your Work: Creating Tasks in XPlanner
- Tracking Your Time and Completing Tasks in XPlanner
- Monitoring Iteration Metrics in XPlanner

Related Reference

- XPlanner Documentation Available from XPlanner.org
Opening the TeamInsight Viewer and the Liferay Portal

After you configure your local workstation for TeamInsight, you can access the Liferay project portal. The Liferay portal is a TeamInsight client tool that displays summary statistics and reports from plugins such as Kosmos, XPlanner, Continuum, Bugzilla, and QALab. You can also open any one TeamInsight tool or all of the tools at once in the TeamInsight Viewer.

To open one or all of the TeamInsight client tools in the TeamInsight Viewer

1 Configure your workstation as a TeamInsight client.
2 Select Window → Open TeamInsight Viewer and do either of the following:
   - Click the name of the tool you want to open (CVS, Bugzilla, Continuum, Liferay, Subversion Viewer, StarTeam or XPlanner).
   - Click Open All to open all TeamInsight tools.

The TeamInsight Viewer opens and displays either the one tool you chose or a window with a tab for each of the tools. It will also contain tabs for any favorite URLs that you have added through the Windows → Open TeamInsight Viewer → Edit Favorites Links path. Depending on your recent logons, a TeamInsight tool might also display its logon window.

To open the Liferay project portal

1 Make sure your workstation is configured as a TeamInsight client. (Clicking Window → Open TeamInsight Viewer displays the URLs of the TeamInsight tools.)
2 Select Window → Open TeamInsight Viewer → Liferay.
3 If you are not logged in to use the TeamInsight tools, the Sign In window appears. Enter your user ID (typically your Email address) and your password, and click Sign In.

The Liferay portal displays portlets for any installed tools that provide project information and links as follows:

- Current status report from JBoss Labs Subversion repository monitor, including the most recent activity
- CVS repository information for project repositories
- Burn down chart and Current iteration details from XPlanner
- Build status from Continuum/Maven and a Project Health link for more information
- Bugzilla status (pages for bugs organized by Important, Newest, Severity, Assignee, and Trends)
- QALab Summary and QALab Classes giving results from the open-source Cobertura and PMD plugins
- StarTeam Task, Bugs and/or StarTeam version control repository information

The Liferay portal is a tabbed window that contains pages for all configured projects as well as a tabs labeled Configuration and Setup. The Configuration page contains:

- A portlet that links to the TeamInsight.ticx file for the current project
- The password change mechanism for a Subversion repository

The Setup tab within a TeamInsight component's portlet allows the Liferay Administrator to reconfigure that portlet. This can be useful for resetting passwords for portlets that were changed or to change other configuration settings that have been invalidated after the install. Changes made through the Setup tab take effect immediately with no restart of the server needed. Some changes are localized to the portlet of a particular project (for example, the Build Status portlet setting that contains the ID used by Continuum to identify a particular project). However,
most settings are common to all instances of a portlet regardless of the project. (for example, the URL used to access the application).

Related Concepts

- ProjectAssist and TeamInsight Overview
- Liferay: The TeamInsight Project Portal

Related Reference

- External Liferay Documentation
Planning a Product Feature: Creating a User Story in XPlanner

User stories describe features planned for a given project. The tasks inside a story represent the work required to complete the feature described in the story. Any user can create a user story and associated tasks in XPlanner. Each story has a Customer and a Tracker associated with the story. Typically, the Customer is the person who requires the feature represented in the story, and the Tracker is the person who is responsible for the completion of the story.

To create a user story in XPlanner

1. Enter XPlanner by selecting Window ➤ Open TeamInsight Viewer ➤ XPlanner or by selecting Window ➤ Add Mylar Repository ➤ XPlanner.
2. On the Top (XPlanner Projects) page, click the ID of your project, such as Sprint 3 or Backlog.
3. On the Project page, click the ID of the iteration where you want to add a story.
4. On the iteration page, click Create Story.
5. On the Define Story page, complete the fields as follows:
   - Name: Enter a descriptive name for the story, such as New Font Widget.
   - Duration: Enter the hours you have worked on this task.
   - Disposition: Select from Planned, Carried Over, or Added.
   - Customer: Enter the name of the person who requires or uses the product of the story.
   - Tracker: Enter the name of the person who is responsible for completing the story.
   - Status: Select from Draft, Defined, Estimated, Planned, Implemented, Verified, or Accepted.
   - Priority: Enter an arbitrary number indicating relative priority of this story.
   - Estimated Hours: Enter the number of hours you are estimating to complete the work for the story (such as 40 or 3.5).
   - Description: Enter a description of the purpose and end result of the feature represented in this story. For example, “Add a new widget to the application that allows user to select the font displayed on the screen.”

6. To create a story using the parameters you have entered, click Create. To reset the fields and start over, click Reset.

Related Concepts

- XPlanner: Project and Team Management
- Mylar Concepts

Related Tasks

- Planning Your Work: Creating Tasks in XPlanner
- Tracking Your Time and Completing Tasks in XPlanner
- Adding Mylar Repositories for Bugzilla and XPlanner
Planning Your Work: Creating Tasks in XPlanner

Any user can add tasks to a project in XPlanner. Each task has an **Acceptor** associated with the task. Typically, the Acceptor is the person who is assigned to complete the work in the task.

**To create a task in XPlanner**

1. Enter XPlanner either by selecting **Window ▶ Open TeamInsight Viewer ▶ XPlanner** or by selecting **Window ▶ Add Mylar Repository ▶ XPlanner**.
2. Navigate to your project and to a specific iteration in the project.
3. On the **Iteration** page, click the ID of a user story.
4. On the **Story** page, click **Create Task**.
5. On the **Define Task** page:
   - In the **Name** field, enter a name that summarizes the task. This is the only required field. Other fields can be easily changed later.
   - In the **Type** drop-down list, select the type of task (Feature, Defect, Debt, FTest, ATest, or Overhead).
   - In the **Disposition** drop-down list, select the disposition - Planned, Discovered, Added, or Carried Over).
   - Assign a person from the **People** list as **Acceptor**. Select the person who is to perform the task.
   - In the **Estimated Hours** field, enter the number of hours to finish the task.
   - In **Description**, enter a description of the task, including necessary details to complete the task.
6. Click **Create**. To clear the fields on the **Define Task** page, click **Reset**.

**Related Concepts**

- XPlanner: Project and Team Management
- Mylar Concepts

**Related Tasks**

- Planning a Product Feature: Creating a User Story in XPlanner
- Tracking Your Time and Completing Tasks in XPlanner
- Adding Mylar Repositories for Bugzilla and XPlanner
Querying Bugzilla for Bug Reports

The Bugzilla component of TeamInsight is loaded during the ProjectAssist server installation or when the Administrator adds a new project. The ProjectAssist Administrator initially creates the access for individual project team members to the TeamInsight Bugzilla tool and all users are assigned the same password.

To query Bugzilla for bug reports

1. After you reach the Bugzilla Main Page window, click on the Searching existing bug reports link or Actions Search to search for existing bug reports, comments or patches. The Bugzilla Query page opens. You can select either the Find a Specific Bug tab or the Advanced Search tab.

2. By selecting the Find a Specific Bug tab, you can find a specific bug by entering words that describe it. Bugzilla searches bug descriptions and comments for the specified words and returns a list of matching bugs sorted by relevance. Select the appropriate choice from the Status: and Product: drop-down lists and enter your word search criteria in the Words: field. Click on the Search button to initiate your query.

3. By selecting the Advanced Search tab, you can narrow the search criteria by specifying a number of fields or options. Bugzilla searches bug descriptions and comments for the specified words and returns a list of matching bugs sorted by relevance. Select the appropriate choice from the following page items:

   - **Summary**: includes a drop-down box to specify the type of string matching and an area to enter the search string text
   - **Product**: is a drop-down list for selecting the product to which the bug will be applied
   - **Component**: is a drop-down list for selecting the product component to which the bug is applicable
   - **Version**: is a drop-down list for selecting the product version
   - **A Comment**: is a drop-down list for selecting the search criterion and an area to enter the search string text
   - **The URL**: is a drop-down list for selecting the search criterion and an area to enter the search string text
   - **White Board**: is a drop-down list for selecting the search criterion and an area to enter the search string text
   - **Keywords**: is a drop-down list for selecting the search criterion and an area to enter the search string text
   - **Status**: is a drop-down list for selecting a search by bug status
   - **Resolution**: is a drop-down list for selecting a search by bug resolution
   - **Severity**: is a drop-down list for selecting a search by the bug severity
   - **Priority**: is a drop-down list for selecting a search by assigned bug priority
   - **Hardware**: is a drop-down list for selecting your computer hardware
   - **OS**: is a drop-down list for selecting your operating system
   - **Email and Numbering**: allows searching by email recipients or bug numbers according to the specified strings
   - **Bug Changes**: allows searching by a specified date range for any of the selected change types selected in the drop-down list
   - **Sort results by**: specifies the sort of for returned search values
   - **Advanced Searching Using Boolean Charts**: allows searching based on boolean values

4. Click on the Search button to initiate your query.

5. Once you have run a search, the Bugzilla Bug List page appears. You can save your search for by entering a name in the as field and clicking on the Remember search button. All saved searches are listed after the Saved Searches: field.
Related Concepts

- Bugzilla: Defect Tracking System
- Mylar Concepts

Related Tasks

- Managing Bug Reports in Bugzilla
- Logging in to TeamInsight Bugzilla
- Creating or Generating Bug Reports in Bugzilla
- Managing Bug Reports in Bugzilla
- Adding Mylar Repositories for Bugzilla and XPlanner

Related Reference

- Bugzilla Resources and Documents
- The Bugzilla Guide
Tracking Your Time and Completing Tasks in XPlanner

To examine your tasks in XPlanner

1. Enter XPlanner either by selecting Window ▶ Open TeamInsight Viewer ▶ XPlanner or by selecting Window ▶ Add Mylar Repository ▶ XPlanner

2. Do either of the following:
   - Click the Me command, available in the upper right corner of most pages in XPlanner, to display the Person page. Your Person page lists all your planned and completed tasks, as well as the user stories where you are the customer or tracker. On your Person page, you can edit the content of your tasks and record the time you have devoted to tasks. To delete or move tasks, however, you must first click on the task name to open the Task page.
   - Navigate to your project. Then from the Project page, navigate to the relevant iteration, to the user story, and finally the Task page. On the Task page, you can manage your tasks as described in the following procedure.

To manage tasks (Edit, Delete, Move/Continue, Edit Time, Export)

1. Navigate from the Project page to the Iteration page, to the Story page, and finally to the Task page.

2. On the Task page, you can perform several actions:
   - Edit opens the Edit Task window in which you can add or change details about the task.
   - Delete deletes the task from the story and project.
   - Move/Continue allows you to select the destination and then move the task to another story or iteration.
   - Edit Time displays Start Time and End Time fields, as well as Duration and Person fields. Enter time you have spent on the task by using either Duration or a combination of Start Time and End Time.
   - Export exports the task as a PDF or as a JRPDF.
   - History displays the current XPlanner hierarchy, from project to story, and task.
   - Print prints the task.

To enter and track time devoted to tasks

1. Navigate through XPlanner from the Project page to the Iteration page to the Story page and then to the Task page.

2. On the Task page, enter the time you have spent on the task by doing either of the following:
   - In Duration, enter the number of hours spent on the task, such as 32 or 2.5.
   - In Start Time and End Time, enter the time of day when you started and ended work on the task. Use the format YYYY-MM-DD HH:MM. Click Enter Time to automatically enter the current time in either of these fields.

3. If you return to the Story page, you will see the time decremented on the progress field of the task.

To complete a task in XPlanner

1. On the Task page, verify that all the hours spent on the task have been entered.
2 Click the **Complete Task** button.

3 Navigate to the **Story** page. The **Progress** field should be filled with a different color from that used for tasks still in progress.

**Related Concepts**

- [XPlanner: Project and Team Management](#)
- [Mylar Concepts](#)

**Related Tasks**

- [Adding Mylar Repositories for Bugzilla and XPlanner](#)
- [Planning a Product Feature: Creating a User Story in XPlanner](#)
- [Planning Your Work: Creating Tasks in XPlanner](#)
- [Moving or Continuing a Story or Task in XPlanner](#)
Using Continuum/Maven for Continuous Integration Builds

As part of the ProjectAssist install, Continuum is installed on a server or assimilated from a previously existing installation. Continuum allows for continuous builds during the software development cycle. By default, two build definitions are automatically configured by ProjectAssist when the Continuum component is installed. One build definition runs hourly and does a clean and install. The other build definition runs once a day. This daily build performs the more lengthy site generation, which includes running reports.

The Continuum administrator can add users, change user passwords and perform other administrative tasks. To most users, the continuous build process appears seamless. They only need to go to the Continuum server if they wanted to force an immediate build.

**Note:** Only the Continuum administrator can change user passwords. Users cannot change their own passwords in Continuum.

To schedule additional builds (administrator)

1. Go to the Continuum component from either the TeamInsight Viewer or through your web browser directly.
2. Login in with your Continuum administrator username and password.
3. Click **Submit** to authenticate your login.
4. The **Continuum Projects** page opens. The portal displays project information about all projects. More information about the project can be obtained by clicking on the project link. A list on the left-hand side of the page links to **Continuum** information, **Add Project** tasks, **Administration** tasks and a **Legend** displaying the meanings of the various icons.
5. In the **Administration** task section, click on **Schedules**. This brings up the **Schedules** page, which lists the schedules installed with the Continuum server component. You can edit these schedules by clicking on the edit icon on the right of the schedule. To add a new schedule, click **Add** and complete the requested information.

To perform other administrative tasks

1. From links on the **Continuum Projects** page, the Continuum administrator can edit the general configuration information on the **General Configuration** page, manage user groups rights and privileges on the **Group Management** page, and add/edit users and user passwords on the **User Management** page.
2. The **Continuum Projects** page has several **Add Projects** links that allow the administrator to add new projects according to project type (Maven 2.0, Maven 1.x, Ant and Shell). However, the ProjectAssist Continuum component currently supports only Maven 2.0 projects.

To force an immediate build

1. From the **Continuum Projects** page, a build can be forced immediately on any listed project.
2. With all current project listed, go to the icons on the right-hand side next to the project name.
3. Click the **Build Now** icon to generate an immediate build of the code. Refer to the **Legend** area on the left-side of the **Continuum Projects** page if you want to know the meanings of the various icons.

Refer to the following documentation links for more information on Continuum and Maven.
Related Concepts

- ProjectAssist and TeamInsight Overview
- Continuum/Maven: Continuous Build System
- Subversion: Source Code Repository
- CVS: Source Code Repository
- StarTeam: Source Code Repository, Change Request Tracking, and Task Provider

Related Tasks

- Checking Out a Project, Making Changes, and Checking Your Changes Into the Repository
- Using the Subversion Viewer for Browsing the Project Repository

Related Reference

- Continuum Online Resources and Documents
- Maven Online Resources and Documents
Using the Subversion Viewer for Browsing the Project Repository

TeamInsight provides the Sventon read-only browser for viewing the Subversion repository. This topic describes how to use the Subversion Viewer (Sventon) to:

- View the Subversion repository
- Download a file from the repository
- Flatten the directory
- Display the log of changes or the current file locks
- Diff a selected file to the previous version

To open the Subversion Viewer and browse the repository

1. Click Window ➤ Open TeamInsight View ➤ Subversion Viewer.
2. On the TeamInsight portal, navigate to the Subversion browser.
   The Subversion Viewer displays the directory containing the central repository for your development project.

Note: To check out a local copy of the repository into your workspace, click Project ➤ Checkout Project and select your project.

Note: To open the SVN Repository view for browsing the Subversion repository, click Window ➤ Open Perspective ➤ Other ➤ SVN Repository Exploring.

To download a file from the repository

1. On the Subversion Viewer, navigate through the tree structure and open the file you want to download.
2. On the Show file window, click Download.
3. On the File Download dialog box, click Save.
4. On the Save As dialog box, locate the directory to contain the copy of the file and click OK.

Note: Downloading a file from the Subversion Viewer does not place the file in your JBuilder 2007 workspace. To check out a local copy of the repository into your workspace, click Project ➤ Checkout Project.

To flatten the directory

1. On the Subversion Viewer, click Flatten dir. The viewer flattens the directory by displaying the repository as if all files were in one directory.
   
   Note: In a large project, flattening the directory can take time and files might be difficult to locate.

2. To return the browser view to its original nested status, click go! on Go to path.

To display the log of changes or the current file locks

1. On the Subversion Viewer, click Show log or Show locks.
   The viewer displays the log of changes to the repository or the list of current file locks.
2 To return to the browser view, click Show directory.

To diff files in the repository

1 On the Subversion Viewer, navigate through the tree structure and double-click the file for which you want to display historical differences.

2 On the Show file window, click Diff to previous.
   The viewer displays a table listing the differences between the current and the previous versions of the file.

3 To return to the browser view, click Show directory.

Related Concepts
   Subversion: Source Code Repository

Related Tasks
   Configuring Your TeamInsight Client
   Checking Out a Project, Making Changes, and Checking Your Changes Into the Repository
Peer to Peer Collaboration

The JBuilder 2007 peer to peer subsystem allows you to collaborate with peers on the same local area network (LAN) as you are. You can chat with peers and share data with peers. You can also share projects through a repository.

In This Section

Chatting with Peers
Describes how to chat with peers and view the chat log.

Enabling Peer to Peer Collaboration
Describes how to enable peer to peer collaboration and set your status.

Managing Contact Groups
Describes how to create and manage contact groups.

Opening a Peer to Peer Session
Describes how to open a session with a peer or group.

Sending Data To Peers
Describes how to send files, lines of text in external files, stack traces, or web links.

Setting Collaboration Preferences
Describes how to set preferences for peer to peer collaboration.

Sharing Team-Enabled Projects with Peers
Describes how to share projects with peers that are checked into a version control system.
Chatting with Peers

To chat with peers

1. Open the Peers view (Window ▶ Show View ▶ Other ▶ Peer to Peer ▶ Peers) and set your status to Available.
2. Double-click the name of the peer or contact group you want to chat with or use multiple selection of peers, right-click and select Open Session.
   The Collaboration pane is opened on the right of the Peers view. The connection is displayed in the chat area. The chat is recorded on your machine, if chat logging is enabled.
3. Type a message into the text field at the bottom of the Collaboration pane.
4. Press ENTER to send the message.
   The message is displayed in the chat area of the Collaboration pane, both on your machine and on the peer's machine(s).

To view and delete the chat log

1. In the Peers pane on the left of the Peers view, select the name of the peer with whom you have chatted.
   Note: Each member of the collaboration has a copy of the chat session in the member's individual log.
2. To view the chat log, right-click and choose View Chat Log.
   The chat log is displayed in the editor as a text file. It is UTF-8 encoded.
3. To delete the chat log, right-click and choose Delete Chat Log.
   The chat log is deleted for that peer.

You set the chat log file location on the Peer to Peer page of the Preferences dialog box (Window ▶ Preferences ▶ Peer to Peer).

Related Concepts
   Peer to Peer Collaboration

Related Tasks
   Setting Collaboration Preferences
   Enabling Peer to Peer Collaboration
   Opening a Peer to Peer Session
   Sharing Team-Enabled Projects with Peers
   Sending Data To Peers
   Managing Contact Groups

Related Reference
   Peers View
   New Contact Group
   Peer To Peer Preferences
   Send Stack Trace
   Send Web Link
   Send VCS Link
Enabling Peer to Peer Collaboration

To use the peer to peer subsystem, you need to enable it and set up your identity. As you work, you can change your status from Available to Away or Offline.

To enable collaboration and create your identity

1. Open the Peer to Peer page of the Preferences dialog box (Window ▶ Preferences ▶ Peer to Peer).
2. Check the Enable Peer To Peer Subsystem option.
3. Enter your user name in the Name field. This defaults to your user logon.

   Note: Your user name is displayed in the Peers pane on your peers' machines.

4. Enter an optional description in the Description field. This description can help identify you to peers.

   Note: The description is displayed in a tooltip in the Peers pane on your peers' machines.

5. Enter the name of an image file in the Image field. The image helps identify you to other peers in a collaboration session. You can use the Browse button to browse to the image file location.

   Note: The image is displayed in a tooltip in the Peers pane on your peers' machines. Any icon you use is automatically resized to 48 x 48 pixels. The image may be distorted if resized.

6. Click Apply and OK to apply and save your identity settings.

The peer to peer subsystem is enabled and the Peers view is opened, with your status set to Available. You will see any other peers that are available on your LAN. Peers should be able to see you as an available peer.

To set your status

1. Open the Status drop-down list. The drop-down list box is located at the top of the Peers pane on the left side of the Peers view.
2. Choose a status from the list.

   - Available — You are available for collaboration. Your name, description, status, selected image, and IP address are displayed in the Peers list on your peers' computers.
   - Offline — You are offline. This terminates the active session, terminates the LAN connection, and removes your name from the Peers list on your peers' computers.
   - Away — You are away from your desk. This status is displayed next to your name in the Peers list on your peers' computers.
Related Concepts

Peer to Peer Collaboration

Related Tasks

Setting Collaboration Preferences
Opening a Peer to Peer Session
Managing Contact Groups
Chatting with Peers
Sharing Team-Enabled Projects with Peers
Sending Data To Peers

Related Reference

Peers View
New Contact Group
Peer To Peer Preferences
Send Stack Trace
Send Web Link
Send VCS Link
Managing Contact Groups

A contact group is a group of peers. You manage contact groups in the Peers pane.

To add a contact group

1. Right-click the Peers pane and choose Add Contact Group. The New Contact Group dialog box is displayed.
2. Enter the name of the group in the Group Name field and click OK. The name of the group is added to the Contact Groups list in the Peers pane.

To add peers to a contact group

1. In the Available Local Peers list of the Peers pane, right-click the name of the peer you want to add to the group.
   Tip: You can select more than one peer at a time to add to a group.
2. Choose Add Peer(s) To Contact Group.
3. Choose the group from the drop-down menu. The peer is added to the selected group and displayed in the Contact Groups list in the Peers pane.

To remove a peer from a contact group

1. Open the group in the Contact Groups list in the Peers pane.
2. Right-click the name of the peer to remove from the group.
   Tip: You can select more than one peer at a time to remove from a group.
3. Choose Remove Peer(s) From Contact Group.

To remove a contact group

1. Right-click the name of the group you want to remove.
2. Choose Remove Contact Group(s).
   Tip: You can select more than one group at a time to remove.
Related Concepts

Peer to Peer Collaboration

Related Tasks

Setting Collaboration Preferences
Enabling Peer to Peer Collaboration
Opening a Peer to Peer Session
Chatting with Peers
Sending Data To Peers

Related Reference

Peers View
New Contact Group
Peer To Peer Preferences
Send Stack Trace
Send Web Link
Send VCS Link
Opening a Peer to Peer Session

To open a session with a peer or contact group

1. If the Peers view is not already open, open it with the **Window ➤ Show View ➤ Other ➤ Peer to Peer ➤ Peers** command.
2. Right-click a peer or contact group you want to collaborate with in the Peers pane.
3. Choose **Open Session** to open the session.

   **Note:** You can also double-click the peer's name to open a session. You can multi-select peers to open a session with more than one peer.

A tab is added to the Collaboration pane on the right side of the Peers view. The Collaboration pane displays the peer or peers with whom you are connected and the chat area. Once you start a chat or send a file, the Collaboration pane tab on the peers' machine opens and displays information.

**Tip:** You can also drag a file from the **Package Explorer** or the **Navigator** directly only to the peer or contact group name in the Peers pane. A chat session is opened if one is not already open.

To close a session with a peer or contact group

1. Click the X on the tab of the session you wish to close in the Collaboration pane.
2. A message indicating that you have left the session is displayed in the Collaboration pane on the peer's machine.

To close all sessions

1. Click the **Close All Collaboration Sessions** button (the X) on the Collaboration pane toolbar.
2. A message indicating that you have left the session is displayed in the Collaboration pane on all peer machines. All sessions are closed and the Collaboration pane on your machine is closed.
Related Concepts

Peer to Peer Collaboration

Related Tasks

Setting Collaboration Preferences
Enabling Peer to Peer Collaboration
Sharing Team-Enabled Projects with Peers
Managing Contact Groups
Chatting with Peers
Sending Data To Peers

Related Reference

Peers View
New Contact Group
Peer To Peer Preferences
Send Stack Trace
Send Web Link
Send VCS Link
Sending Data To Peers

You can send files, diagrams, web links, and stack traces to peers in a chat session.

To send a file or diagram in a chat session

1. Open a session with a peer or contact group.
2. Click the **Send Files To Peers In Collaboration** button on the **Collaboration** pane toolbar.
3. Browse to the file or diagram you want to send in the **Open** dialog box and click **Open**.

The file is sent to the peer(s) in the chat session. A message appears in the peer’s chat area and the file name is automatically downloaded to the folder specified in the **Workspace Folder** field in the **File Transfer** area on the **Peer to Peer** page of the **Preferences** dialog box (Window ➤ Preferences ➤ Peer to Peer).

**Tip:** Dragging text deletes it from the original file unless you hold down the **CTRL** key.

To send a line of text from an external file

1. Open a session with a peer or contact group.
2. Open the application and file that you want to send text from.
3. Select the text in the file and drag it to the chat area for an open session or drop it on the peer in the Peers pane.

**Tip:** Dragging text deletes it from the original file unless you hold down the **CTRL** key.

4. Press **ENTER** to send the line of text.

The text is sent to the peer(s) in the chat session as a message.

To send a web link in a chat session

1. Open a session with a peer or contact group.
2. Click the **Send Web Link To Peers In Collaboration** button on the **Collaboration** pane toolbar.
3. Enter the URL in the **Send Web Link** dialog box and click **OK**.

The URL is sent to the peer(s) in the chat session. A message appears in the chat area and the URL is displayed as a link.

4. Click the URL to open the link in a web browser. Your peer(s) can do the same.

To send a stack trace in a chat session

1. Copy the contents of a stack trace into the Clipboard.
2. Open a session with a peer or contact group.
3. Click the **Send Stack Trace To Peers In Collaboration** button on the **Collaboration** pane toolbar.
4 Paste the stack trace into the Stack Trace dialog box and choose Send.

   The stack trace is sent to the peer(s) in the chat session. A message appears in the chat area and the stack trace is displayed as a link.

5 Click the link to open the stack trace in the Console view using the Java Stack Trace Console. Your peer(s) can do the same.

Related Concepts

   Peer to Peer Collaboration

Related Tasks

   Setting Collaboration Preferences
   Enabling Peer to Peer Collaboration
   Opening a Peer to Peer Session
   Managing Contact Groups
   Chatting with Peers
   Sharing Team-Enabled Projects with Peers

Related Reference

   Peers View
   New Contact Group
   Peer To Peer Preferences
   Send Stack Trace
   Send Web Link
   Send VCS Link
Setting Collaboration Preferences

Most collaboration preferences are set on the Peer to Peer page of the Preferences dialog box. If you modify settings during a open session, you may be prompted to restart your connection to apply the changes.

To set filtering for multiple adapters

1. Open the Peer to Peer page of the Preferences dialog box (Window ➤ Preferences ➤ Peer to Peer).

   **Note:** You can also right-click the Collaboration pane and choose Preferences to display the Preferences dialog box.

2. Choose the adapter you want to use in the Filtering drop-down list, or choose NONE to not use an adapter.

3. Click Apply to apply the settings. Click OK if you're done.

To set chat preferences

1. Open the Peer to Peer page of the Preferences dialog box (Window ➤ Preferences ➤ Peer to Peer).

2. Select the Log Chat Messages option to turn on logging for chat messages.

   **Note:** The chat log is UTF-8 encoded.

3. Enter the name of the directory where you want messages saved in the Workspace Directory field.

4. Click the Incoming Message Color box to set the color for incoming messages.

5. Click the Outgoing Message Color box to set the color for outgoing messages.

6. Click the Status Message Color box to set the color for status messages.

7. Click Apply to apply the settings. Click OK if you're done.

To automatically transfer files into the workspace

1. Select the Automatic Receive Enabled option to automatically transfer files when you're chatting with peers.

   **Note:** If you turn this option off, you will need to click a link to the file when you receive it in order to download it into a directory. You must be in an active chat session.

2. Enter the name of the directory you want files automatically downloaded to in the Workspace Directory field.

3. Click Apply to apply the settings. Click OK if you're done.

To set audio feedback

1. Select the Audio Feedback Enabled option.

2. Adjust the volume slider.

3. Click Apply and OK to apply and save the preferences.
Related Concepts

Peer to Peer Collaboration

Related Tasks

Enabling Peer to Peer Collaboration
Opening a Peer to Peer Session
Managing Contact Groups
Chatting with Peers
Sharing Team-Enabled Projects with Peers
Sending Data To Peers

Related Reference

Peers View
New Contact Group
Peer To Peer Preferences
Send Stack Trace
Send Web Link
Send VCS Link
Sharing Team-Enabled Projects with Peers

Projects are shared through a repository. When projects are shared, the Navigator or Package Explorer displays the project repository and location.

To share projects with a peer

1. Check out the project from the repository.
2. Right-click the project and choose Send VCS Link to Peer.
   The Select Peer dialog box is displayed.
3. Choose the name of the peer you want to send the link to and click Select.
   The project is sent as a VCS link to the selected peer. The message Sending VCS link for project “<Project Name>” is displayed in your chat area.

Note: To send the link to more than one peer, you need to choose the Send VCS Link to Peer command for each peer.

To receive a link to a shared project

1. Click the VCS link you received in the chat area.
2. Log onto the server if you are not already logged on.
3. Navigate the VCS check out dialog boxes.
   The project is checked out locally. The Navigator or Package Explorer displays the project repository and location.

Note: JBuilder 2007 and JBuilder 2006 projects are not compatible for the project sharing feature.

Related Concepts

Peer to Peer Collaboration

Related Tasks

Setting Collaboration Preferences
Enabling Peer to Peer Collaboration
Managing Contact Groups
Chatting with Peers
Sending Data To Peers

Related Reference

Peers View
New Contact Group
Peer To Peer Preferences
Send Stack Trace
Send Web Link
Send VCS Link
IDE Reference
This section lists all of the dialog/wizards information provided through JBuilder 2007.

In This Section
  Project Import Dialogs
  This section lists dialogs/wizards information for project import provided through JBuilder 2007.
  Axis Web Service Dialogs Reference
  This section lists dialogs/wizards information provided through JBuilder 2007 for the Axis Web Service.
  Dynamic Web JPA Modeling Dialogs Reference
  This section lists dialogs/wizards information provided through JBuilder 2007 for dynamic web JPA Modeling applications.
  JPA Modeling Dialogs Reference
  This section lists dialogs/wizards information for creating new JPA Modeling projects provided through JBuilder 2007.
  New EJB Modeling Dialogs Reference
  This section lists dialogs/wizards information provided through JBuilder 2007.
  EJB Modeling Projects from XDoclet Dialogs Reference
  This section lists dialogs/wizards information for converting an EJB project to an EJB Modeling project through JBuilder 2007.
  ProjectAssist and TeamInsight Dialogs
  This section lists dialogs/wizards for the ProjectAssist and TeamInsight features provided through JBuilder 2007.
  Peer to Peer Dialogs Reference
  This section lists dialogs/wizards information provided for peer to peer interaction through JBuilder 2007.
Project Import Dialogs

This section lists the dialog/wizards information provided for project import through JBuilder 2007.

In This Section

  Project Import Wizard
  Import a project to the IDE
Project Import Wizard

File▸Import

Use this dialog box to import a project to JBuilder 2007.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>Choose the type of source files to be imported into JBuilder 2007</td>
</tr>
<tr>
<td>General</td>
<td>Click one of general source types to import and click Next</td>
</tr>
<tr>
<td>CVS</td>
<td>Click to import a CVS file and click Next</td>
</tr>
<tr>
<td>EJB</td>
<td>Click EJB Jar File and click Next</td>
</tr>
<tr>
<td>J2EE</td>
<td>Click one of the J2EE source types to import and click Next</td>
</tr>
<tr>
<td>Legacy JBuilder</td>
<td>Click one of the Legacy JBuilder source types to import and click Next</td>
</tr>
<tr>
<td>Modeling</td>
<td>Click Profile Plug-ins to import a modeling project and click Next</td>
</tr>
<tr>
<td>Mylar</td>
<td>Click Mylar Task Data to import a Mylar project and click Next</td>
</tr>
<tr>
<td>Plug-in Development</td>
<td>Click the type of plug-in development project to import and click Next</td>
</tr>
<tr>
<td>Profiling and Logging</td>
<td>Click the type of project to import and click Next</td>
</tr>
<tr>
<td>Team</td>
<td>Click Team Project Set and click Next</td>
</tr>
<tr>
<td>Test</td>
<td>Click a test type and click Next</td>
</tr>
<tr>
<td>Web</td>
<td>Click WAR File and click Next</td>
</tr>
<tr>
<td>Web Services</td>
<td>Click a web service type and click Next</td>
</tr>
<tr>
<td>Other</td>
<td>Select a source type and click Next</td>
</tr>
</tbody>
</table>

Related Concepts

- JBuilder Project Migration Overview
- Importing Legacy Projects

Related Tasks

- Setting Import Properties
- Running an Imported Project
**Axis Web Service Dialogs Reference**

This section lists all of the dialog/wizards information provided for the Axis Web Service through JBuilder 2007.

**In This Section**

- **New Dynamic Web Project: New Axis Web Service Project Wizard**
  Use the **New Dynamic Web Project: New Axis Web Service Project** dialog to create a dynamic web project with Axis Web Service Modeling Support.

- **New Dynamic Web Project: Project Facets**
  Use the **New Dynamic Web Project: Project Facets** dialog to change the facet (unit of functionality) for the web project.

- **New Dynamic Web Project: Web Module**
  Use the **New Dynamic Web Project: Web Module** page to configure Web module settings.
New Dynamic Web Project: New Axis Web Service Project Wizard

Use the **New Dynamic Web Project: New Axis Web Service Project** dialog to create a dynamic web project with Axis Web Service Modeling Support. This is a 3–page wizard.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project name</td>
<td>Specify the new project name.</td>
</tr>
<tr>
<td>Project contents</td>
<td>Specify the file system location (the place where the resources you create are stored.) When the <strong>Use default</strong> check box is selected, the project is created in the file system location where your workspace resides. To change the default file system location, clear the checkbox and locate the path using the <strong>Browse</strong> button.</td>
</tr>
<tr>
<td>Target Runtime</td>
<td>Select the server where you want to deploy the Web project in this field. If you want to define a new server location, click <strong>New</strong> to select a server runtime environment.</td>
</tr>
<tr>
<td>EAR Membership</td>
<td>Adds the Web Services project to an enterprise archive (EAR) file. A new or existing EAR project file must be associated with the new Web project to facilitate deployment. The EAR file contains artifacts necessary to build a Web service. Check the <strong>Add project to an EAR</strong> box to add the project to an EAR file. Specify an existing EAR Project Name in the <strong>EAR Project Name</strong> area or use the default value of <strong>Projectname</strong> EAR. You can also click <strong>New</strong> to take you to the <strong>New EAR Application Project</strong> wizard to define a new EAR application project. When your Web project is created, the new EAR file is also created with the specified name. The default name is the Web project name appended with the letters <strong>EAR</strong>.</td>
</tr>
</tbody>
</table>

Click **Next** to go to the **Project Facets** page. The **Project Facets** page of this wizard allow the selection of various project functionality.

Click **Finish** to create the specified Axis Web Services project.

**Related Concepts**
- Web Services Overview

**Related Tasks**
- Working in the Web Services Explorer
- Designing a Bottom-Up Web Service Using the Apache Axis Runtime
- Designing a Top-Down Web Service Using the Apache Axis Runtime

**Related Reference**
- New Dynamic Web Project: Project Facets
- New Dynamic Web Project: Web Module
The **New Dynamic Web Project: Project Facets** dialog is the second page of the **New Dynamic Web Project: New Axis Web Service Project Wizard** wizard. Use the **Project Facets** page to select the various functionalities for the project. You can select facets by the custom configuration or use preconfigured project types.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configurations</td>
<td>Select the configuration associated with the project. You can select pre-filled configurations or create custom configurations. Configurations can be saved or deleted using the appropriate button.</td>
</tr>
<tr>
<td>Project Facet</td>
<td>Select the check boxes next to the facets you want this project to have. Only valid facets for the project are listed. The list of runtimes selected for the project limits the facets shown in the list. Only the facets compatible with all selected target runtimes are shown. The currently selected facets and their version numbers limit the other facets shown in the list to compatible facets. You can find out more about the requirements and limitations for each facet by right-clicking the facet name and then clicking <strong>Show Constraints</strong>. To remove a facet, clear its check box. Not all facets can be removed.</td>
</tr>
<tr>
<td>Version</td>
<td>Choose a version number for the facet by clicking the current version number and selecting the version number you want from the drop-down list.</td>
</tr>
<tr>
<td>Show Runtimes</td>
<td>Click the <strong>Show Runtimes</strong> button and select the runtimes that you want the project to be compatible with if you want to limit the project compatibility with one or more runtimes.</td>
</tr>
</tbody>
</table>

Click **Next** to go to the **New Dynamic Web Project: Web Module** page. The **New Dynamic Web Project: Web Module** page of this wizard enables the deployment of the project as a dynamic web module.

Click **Finish** to create the specified Axis Web Services project.

**Related Concepts**
- Web Services Overview

**Related Tasks**
- Working in the Web Services Explorer
- Designing a Bottom-Up Web Service Using the Apache Axis Runtime
- Designing a Top-Down Web Service Using the Apache Axis Runtime

**Related Reference**
- New Dynamic Web Project: New Axis Web Service Project Wizard
- New Dynamic Web Project: Web Module
New Dynamic Web Project: Web Module


<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context Root</td>
<td>The context root is the Web application root. The Web application root is the top-level directory of your application when it is deployed to the Web server. The context root can be changed after you create a project by using the project Properties dialog, which is accessed from the project's pop-up menu. The context root is used by the links builder to ensure that your links remain ready to publish as you move and rename files inside your project.</td>
</tr>
<tr>
<td>Content Directory</td>
<td>Specifies the content directory of the project.</td>
</tr>
<tr>
<td>Java Source Directory</td>
<td>Specifies the Java source directory.</td>
</tr>
</tbody>
</table>

Click Finish to create the specified Axis Web Services project.

Related Concepts

- Web Services Overview

Related Tasks

- Working in the Web Services Explorer
- Designing a Bottom-Up Web Service Using the Apache Axis Runtime
- Designing a Top-Down Web Service Using the Apache Axis Runtime

Related Reference

- New Dynamic Web Project: New Axis Web Service Project Wizard
- New Dynamic Web Project: Project Facets
Dynamic Web JPA Modeling Dialogs Reference

This section lists the dialog/wizards for Dynamic Web JPA Modeling application development provided through JBuilder 2007.

In This Section

- **New Dynamic Web Project: New Dynamic Web JPA Modeling Project**
  Use the [New Dynamic Web Project: New Dynamic Web JPA Modeling Project](#) dialog to creates a new Dynamic web modeling project with Java Persistence API (JPA) support.

- **New Dynamic Web Project: Persistence unit settings page**
  Use the [New Dynamic Web Project: Persistence unit setting page](#) dialog to set the persistence settings for your new Dynamic web modeling project with Java Persistence API (JPA) support.

- **New Dynamic Web Project: Project Facets**
  Use the [New Dynamic Web Project: Project Facets](#) dialog to change the facet (unit of functionality) for the web project.

- **New Dynamic Web Project: Web Module**
  Use the [New Dynamic Web Project: Web Module](#) page to configure Web module settings for your dynamic Web JPA project.
New Dynamic Web Project: New Dynamic Web JPA Modeling Project

Use the New Dynamic Web Project: New Dynamic Web JPA Modeling Project wizard to create a dynamic web project with Java Persistence API (JPA) support. This is a 4–page wizard.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project name</td>
<td>Specify the new dynamic web JPA modeling project name.</td>
</tr>
<tr>
<td>Project contents</td>
<td>Specify the file system location (the place where the resources you create are stored.) When the Use default check box is selected, the project is created in the file system location where your workspace resides. To change the default file system location, clear the checkbox and locate the path using the Browse button.</td>
</tr>
<tr>
<td>Persistence Provider</td>
<td>Select Hibernate, Toplink, or Other as the persistence manager in the Manager Name field. Check the Add library to the class path box, if not checked.</td>
</tr>
<tr>
<td>Target Runtime</td>
<td>Select the server where you want to deploy the Web project in this field. If you want to define a new server location, click New to select a server runtime environment.</td>
</tr>
<tr>
<td>EAR Membership</td>
<td>Adds the Web Services project to an enterprise archive (EAR) file. A new or existing EAR project file must be associated with the new Web project to facilitate deployment. The EAR file contains artifacts necessary to build a Web service. Check the Add project to an EAR box to add the project to an EAR file. Specify an existing EAR Project Name in the EAR Project Name area or use the default value of Projectname EAR. You can also click New to take you to the New EAR Application Project wizard to define a new EAR application project. When your Web project is created, the new EAR file is also created with the specified name. The default name is the Web project name appended with the letters EAR.</td>
</tr>
</tbody>
</table>

Click Next to go to the Persistence unit settings page page of this wizard to specify the persistence settings. Click Finish to create the specified dynamic Web JPA modeling project.

Related Concepts

Java EE Applications Overview
Modeling Applications Overview
Runtime Servers

Related Tasks

Create a Dynamic Web Java Persistence API (JPA) Modeling Project
Setting Up a Runtime Server

Related Reference

New Dynamic Web Project: Persistence unit settings page
New Dynamic Web Project: Project Facets
New Dynamic Web Project: Web Module
Hibernate Documentation
TopLink Resources
Use the New Dynamic Web Project: Persistence unit settings page of the New Dynamic Web JPA Modeling Project wizard to specify your project persistence settings. This is page 2 of a 4–page wizard.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistence unit name</td>
<td>Specify the name of the persistence unit.</td>
</tr>
<tr>
<td>Transaction type</td>
<td>Choose the transaction type.</td>
</tr>
<tr>
<td>Database type</td>
<td>Choose the type of database. This field is limited by the Manager Name selected on the previous page of this wizard.</td>
</tr>
<tr>
<td>Database</td>
<td>Specifies further information about the selected database, including 'Database Connection and database Schema information. The database connection describes the method used to talk with the database server. The database schema describes the structure of the database. An active connection must exist to select a database schema. The database connection information is applicable only for Hibernate and Toplink. Database connection information for other persistence managers has to be specified manually. An active connection must exist to select a database schema. Click on the Add Connection link to setup a database connection. The Schema dropdown menu is automatically populated depending on the active database connection.</td>
</tr>
<tr>
<td>Add Connection or Reconnect</td>
<td>Adds a database connection or reconnects using an existing connection.</td>
</tr>
</tbody>
</table>

Click Next to go to the Project Facets page. The Project Facets page of this wizard allows the specification of project functionalities.

Click Finish to create the specified dynamic Web JPA modeling project.

Related Concepts

- Java EE Applications Overview
- Modeling Applications Overview
- Runtime Servers

Related Tasks

- Create a Dynamic Web Java Persistence API (JPA) Modeling Project
- Setting Up a Runtime Server

Related Reference

- New Dynamic Web Project: New Dynamic Web JPA Modeling Project
- New Dynamic Web Project: Project Facets
- New Dynamic Web Project: Web Module
- Hibernate Documentation
- TopLink Resources
New Dynamic Web Project: Project Facets

The **New Dynamic Web Project: Project Facets** dialog is the third page of the **New Dynamic Web Project: New Dynamic Web JPA Modeling Project** wizard. Use the **Project Facets** page to select the various functionalities for the project. You can select facets by the custom configuration or use preconfigured project types.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configurations</td>
<td>Select the configuration associated with the project. You can select prefilled configurations or create custom configurations. Configurations can be saved or deleted using the appropriate button.</td>
</tr>
<tr>
<td>Project Facet</td>
<td>Select the check boxes next to the facets you want this project to have. Only valid facets for the project are listed. The list of runtimes selected for the project limits the facets shown in the list. Only the facets compatible with all selected target runtimes are shown. The currently selected facets and their version numbers limit the other facets shown in the list to compatible facets. You can find out more about the requirements and limitations for each facet by right-clicking the facet name and then clicking <strong>Show Constraints</strong>. To remove a facet, clear its check box. Not all facets can be removed.</td>
</tr>
<tr>
<td>Version</td>
<td>Choose a version number for the facet by clicking the current version number and selecting the version number you want from the drop-down list.</td>
</tr>
<tr>
<td>Show Runtimes</td>
<td>Click the <strong>Show Runtimes</strong> button and select the runtimes that you want the project to be compatible with if you want to limit the project compatibility with one or more runtimes.</td>
</tr>
</tbody>
</table>

Click **Next** to go to the **New Dynamic Web Project: Web Module** page. The **New Dynamic Web Project: Web Module** page of this wizard enables the deployment of the project as a dynamic web module.

Click **Finish** to create the specified dynamic Web JPA modeling project.

**Related Concepts**
- Java EE Applications Overview
- Modeling Applications Overview
- Runtime Servers

**Related Tasks**
- Create a Dynamic Web Java Persistence API (JPA) Modeling Project

**Related Reference**
- New Dynamic Web Project: New Dynamic Web JPA Modeling Project
- New Dynamic Web Project: Persistence unit settings page
- New Dynamic Web Project: Web Module
- Hibernate Documentation
- TopLink Resources
New Dynamic Web Project: Web Module


<table>
<thead>
<tr>
<th>Context Root</th>
<th>The context root is the Web application root. The Web application root is the top-level directory of your application when it is deployed to the Web server. The context root can be changed after you create a project by using the project Properties dialog, which is accessed from the project's pop-up menu. The context root is used by the links builder to ensure that your links remain ready to publish as you move and rename files inside your project.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content Directory</td>
<td>Specifies the content directory of the project.</td>
</tr>
<tr>
<td>Java Source Directory</td>
<td>Specifies the Java source directory.</td>
</tr>
</tbody>
</table>

Click Finish to create the specified dynamic Web JPA modeling project.

Related Concepts

Java EE Applications Overview
Modeling Applications Overview
Runtime Servers

Related Tasks

Create a Dynamic Web Java Persistence API (JPA) Modeling Project
Setting Up a Runtime Server

Related Reference

New Dynamic Web Project: New Dynamic Web JPA Modeling Project
New Dynamic Web Project: Persistence unit settings page
New Dynamic Web Project: Project Facets
Hibernate Documentation
TopLink Resources
EJB Modeling Projects from XDoclet Dialogs Reference

This section lists the dialog/wizards information for converting an EJB project to an EJB Modeling project through JBuilder 2007.

In This Section

- **EJB Modeling Project from XDoclet annotated WTP project**

  Use the **EJB Modeling Project from XDoclet annotated WTP Project** wizard to convert an EJB XDoclet annotated WTP project to an EJB modeling project.
EJB Modeling Project from XDoclet annotated WTP project

Use the EJB modeling project from EJB project: EJB Modeling Project from XDoclet annotated WTP Project wizard to convert an EJB XDoclet annotated WTP project to an EJB modeling project.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Projects</td>
<td>Lists all available WTP EJB projects in the current workspace. Only WTP EJB projects without the modeling nature are displayed in this list. Click the Select All button or the Deselect All button to select all deselect all items in the list.</td>
</tr>
</tbody>
</table>

Click Finish to create the new EJB modeling project.

Related Concepts
- Java EE Applications Overview
- Modeling Applications Overview

Related Tasks
- Setting Up a Runtime Server
- Create an EJB Modeling Project with XDoclet Annotations

Related Reference
- Creating Enterprise Beans with XDoclet Annotation Support
New EJB Modeling Dialogs Reference

This section lists all of the dialog/wizards information provided through JBuilder 2007.

In This Section

EJB Modeling Project from Java Project
Use the EJB Modeling Project from Java Project wizard to create an EJB modeling project from an existing Java project.

EJB Modeling Project from Java Project: Create New EJB Project from Java Project
Use the EJB Modeling Project from Java Project wizard to create an EJB modeling project from an existing Java project. This is page 2 of a 3–page wizard.

EJB Modeling Project from Java Project: Project Facets
Use the EJB Modeling Project from Java Project wizard to create an EJB modeling project from an existing Java project. This is page 3 of a 3–page wizard.
EJB Modeling Project from Java Project

New ▶ Project ▶ EJB ▶ EJB Modeling Project from Java Project

Use the EJB Modeling Project from Java Project wizard to create an EJB modeling project from an existing Java project. This is a 3–page wizard.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Projects</td>
<td>Lists all available Java projects for conversion, including modeling projects, in the selected workspace. Use the Browse button to set the workspace directory. Click the Refresh button to refresh the list contents.</td>
</tr>
</tbody>
</table>

Click Next to go to the Creates new EJB Modeling Project from Java Project page. The Creates new EJB Modeling Project from Java Project page of this wizard specifies characteristics of the new EJB modeling project. Click Finish to create the specified EJB modeling project from a Java project.

Related Concepts

Java EE Applications Overview
Modeling Applications Overview
Enterprise Java Bean (EJB) Applications Overview

Related Tasks

EJB Modeling Project from a Java Project
Creating an Enterprise Java Bean (EJB) Modeling Project
Setting Up a Runtime Server

Related Reference

EJB Modeling Project from Java Project: Create New EJB Project from Java Project
EJB Modeling Project from Java Project: Project Facets
EJB Modeling Project from Java Project: Create New EJB Project from Java Project

Use the EJB Modeling Project from Java Project wizard to create an EJB modeling project from an existing Java project. This is a 3–page wizard. This is page 2 of a 3–page wizard.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project name</td>
<td>Specify the new project name.</td>
</tr>
<tr>
<td>Project contents</td>
<td>Specify the file system location (the place where the resources you create are stored.) When the Use default check box is selected, the project is created in the file system location where your workspace resides. To change the default file system location, clear the checkbox and locate the path using the Browse button.</td>
</tr>
<tr>
<td>Target Runtime</td>
<td>Select the server where you want to deploy the EJB modeling project. If you want to define a new server location, click New to select a server runtime environment.</td>
</tr>
<tr>
<td>EAR Membership</td>
<td>Adds the project to an enterprise archive (EAR) file. A new or existing EAR project file must be associated with the new Web project to facilitate deployment. The EAR file contains artifacts necessary to build an EJB modeling project. Check the Add project to an EAR box to add the project to an EAR file.</td>
</tr>
<tr>
<td></td>
<td>Specify an existing EAR Project Name in the EAR Project Name area or use the default value of Projectname EAR. You can also click New to take you to the New EAR Application Project wizard to define a new EAR application project. When your Web project is created, the new EAR file is also created with the specified name. The default name is the Web project name appended with the letters EAR.</td>
</tr>
<tr>
<td>UML Version</td>
<td>Select the version of the Unified Modeling Language (UML) standard that will be used to build this project. To switch off the autobuild option, check the Switch off autobuild option for the workspace box.</td>
</tr>
</tbody>
</table>

Click Next to go to the Project Facets page. The Project Facets page of this wizard allow the selection of various project functionality.

Click Finish to create the specified Axis Web Services project.

Related Concepts

Java EE Applications Overview
Modeling Applications Overview
Enterprise Java Bean (EJB) Applications Overview

Related Tasks

EJB Modeling Project from a Java Project
Creating an Enterprise Java Bean (EJB) Modeling Project
Setting Up a Runtime Server

Related Reference

EJB Modeling Project from Java Project
EJB Modeling Project from Java Project: Project Facets
Use the EJB Modeling Project from Java Project wizard to create an EJB modeling project from an existing Java project. This is a 3–page wizard. This is page 3 of a 3–page wizard.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configurations</td>
<td>Select the configuration associated with the project. You can select prefilled configurations or create custom configurations. Configurations can be saved or deleted using the appropriate button.</td>
</tr>
<tr>
<td>Project Facet</td>
<td>Select the check boxes next to the facets you want this project to have. Only valid facets for the project are listed. The list of runtimes selected for the project limits the facets shown in the list. Only the facets compatible with all selected target runtimes are shown. The currently selected facets and their version numbers limit the other facets shown in the list to compatible facets. You can find out more about the requirements and limitations for each facet by right-clicking the facet name and then clicking Show Constraints. To remove a facet, clear its check box. Not all facets can be removed.</td>
</tr>
<tr>
<td>Version</td>
<td>Choose a version number for the facet by clicking the current version number and selecting the version number you want from the drop-down list.</td>
</tr>
<tr>
<td>Show Runtimes</td>
<td>Click the Show Runtimes button and select the runtimes that you want the project to be compatible with if you want to limit the project compatibility with one or more runtimes.</td>
</tr>
</tbody>
</table>

Click Finish to create the EJB modeling project from a Java project.

Related Concepts

- Java EE Applications Overview
- Modeling Applications Overview
- Enterprise Java Bean (EJB) Applications Overview

Related Tasks

- EJB Modeling Project from a Java Project
- Creating an Enterprise Java Bean (EJB) Modeling Project
- Setting Up a Runtime Server

Related Reference

- EJB Modeling Project from Java Project
- EJB Modeling Project from Java Project: Create New EJB Project from Java Project
JPA Modeling Dialogs Reference

This section lists the dialog/wizards used to create new JPA modeling projects provided through JBuilder 2007.

In This Section

New JPA Modeling Project: Create a JPA modeling project
Use the New JPA Modeling Project: Create a JPA modeling project wizard to create a Java Modeling project with Java Persistence API (JPA) support.

New JPA Modeling Project: Persistence unit settings page
Use the 3-page New JPA Modeling Project: Persistence unit settings page wizard to create a modeling project with Java Persistence API (JPA) support.

New JPA Modeling Project: Java Settings
Use the 3-page New JPA Modeling Project: Java Settings wizard to create a modeling project with Java Persistence API (JPA) support.
New JPA Modeling Project: Create a JPA modeling project

Use the New JPA Modeling Project: Create a JPA modeling project dialog to create a Java Modeling project with Java Persistence API (JPA) support. This is a page 1 of a 3–page wizard.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project name</td>
<td>Specify the new JPA modeling project name.</td>
</tr>
<tr>
<td>Persistence manager</td>
<td>Select Hibernate, Toplink, or Other as the persistence manager in the Manager Name field.</td>
</tr>
<tr>
<td></td>
<td>Check the Add library to the class path box, if not checked.</td>
</tr>
<tr>
<td>Contents</td>
<td>When the Create new project in workspace button is selected, the project is created in the file system location where your workspace resides. When the Create project from existing source button is selected you can specify the file location of the existing source. Locate the path using the Browse button.</td>
</tr>
<tr>
<td>JRE</td>
<td>When the Use default JRE button is selected, the default Java runtime environment is used. When the Use a project specific JRE button is selected you can select a specific project-related JRE from the drop down box. Click on the Configure JRE link to obtain a list of installed JREs and to add, edit, copy, remove, or search for other Java runtime environments.</td>
</tr>
<tr>
<td>Project layout</td>
<td>When the Use project folder as root for sources and class files button is selected, the project folder is used as the root directory for all source and class files. When the Create separate source and output folders button is selected, folders are created for the source and output unique from the project folder. Click the Configure default link to specify the default build path entries used by wizards when new Java projects are created.</td>
</tr>
</tbody>
</table>

Click Next to go to the Persistence unit settings page page of this wizard to specify the persistence settings.

Click Finish to create the specified new JPA modeling project.

Related Concepts
- Java EE Applications Overview
- Modeling Applications Overview

Related Tasks
- Create a Java Persistence API (JPA) Modeling Project
- Setting Up a Runtime Server

Related Reference
- New JPA Modeling Project: Persistence unit settings page
- New JPA Modeling Project: Java Settings
- Hibernate Documentation
- TopLink Resources
Use the New JPA Modeling Project: Persistence unit settings page of the New JPA Modeling Project wizard to specify your project persistence settings. This is page 2 of a 3-page wizard.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Persistence unit name</td>
<td>Specify the name of the persistence unit.</td>
</tr>
<tr>
<td>Transaction type</td>
<td>Choose the transaction type.</td>
</tr>
<tr>
<td>Database type</td>
<td>Choose the type of database. This field is limited by the Manager Name selected on the previous page of this wizard.</td>
</tr>
<tr>
<td>Database</td>
<td>Specifies further information about the selected database, including Database Connection and database Schema information. The database connection describes the method used to talk with the database server. The database schema describes the structure of the database. An active connection must exist to select a database schema. The database connection information is applicable only for Hibernate and Toplink. Database connection information for other persistence managers has to be specified manually. An active connection must exist to select a database schema. Click on the Add Connection link to setup a database connection. The Schema dropdown menu is automatically populated depending on the active database connection. Add Connection or Reconnect Adds a database connection or reconnects using an existing connection.</td>
</tr>
</tbody>
</table>

Click Next to go to the Java Settings page. The Java Settings page of this wizard defines the Java build settings.

Click Finish to create the new JPA modeling project.

Related Concepts
- Java EE Applications Overview
- Modeling Applications Overview

Related Tasks
- Create a Java Persistence API (JPA) Modeling Project
- Setting Up a Runtime Server

Related Reference
- New JPA Modeling Project: Create a JPA modeling project
- New JPA Modeling Project: Java Settings
- Hibernate Documentation
- TopLink Resources
Use the New JPA Modeling Project: Java Settings of the New JPA Modeling Project wizard to specify your build path settings for your Java project. This is page 3 of a 3–page wizard.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>This tab is where you specify the source location (root) of packages containing .java files. These files are then translated to .class files by the compiler and written to the defined output folder.</td>
</tr>
<tr>
<td>Projects</td>
<td>Specifies the required projects on the build path for a new project. This also adds all the classpath entries marked as exported (Order and Export tab) for the required project. These projects are automatically added to the referenced project list. This list is used to determine the build order as a project is built only after all its reference projects have been built.</td>
</tr>
<tr>
<td>Libraries</td>
<td>Choose the libraries to add to the build path, including file system (external) JAR files, folders contain class files, workbench-managed (internal) JAR files and predefined libraries.</td>
</tr>
<tr>
<td>Order and Export</td>
<td>Allows you to move a selected build path entry up or down in the class path order list for the new project. Entries marked in the list with a check mark are exported to be visible to the projects requiring them. Entries can be selected or deselected for exportation; source folders are always exported.</td>
</tr>
<tr>
<td>Details</td>
<td>contains links to additional tasks. Click the Click on Create new source folder to add a new Java source folder to your project. Click on Link additional source to link to a folder in the file system to use as an additional source folder. Click on Configure inclusion and exclusion filters link to specify patterns for inclusion and exclusion filtering.</td>
</tr>
<tr>
<td>Allow output folders for source folders</td>
<td>check this box to permit output folders to be utilized as source folders. Click on Create new source folder to add a new Java source folder to your project.</td>
</tr>
<tr>
<td>Default output folder</td>
<td>use the default name or click Browse to locate a folder to use as the default output folder.</td>
</tr>
</tbody>
</table>

Click Finish to create the new JPA modeling project.

Related Concepts
- Java EE Applications Overview
- Modeling Applications Overview

Related Tasks
- Create a Java Persistence API (JPA) Modeling Project
- Setting Up a Runtime Server

Related Reference
- New JPA Modeling Project: Create a JPA modeling project
- New JPA Modeling Project: Persistence unit settings page
- Hibernate Documentation
- TopLink Resources
ProjectAssist and TeamInsight Dialogs

This section lists the dialog/wizards information provided through JBuilder 2007 for the ProjectAssist and TeamInsight features.

In This Section

- New ProjectAssist File Link
  Create a link to an existing ProjectAssist .pacx file.

- New ProjectAssist File
  Creates a new ProjectAssist stack file (.pacx).

- New ProjectAssist File: Select Stack Components
  Selects the stack components for a new ProjectAssist stack file (.pacx).

- New ProjectAssist file: Choose disk scan paths
  Use the Choose disk scan paths to specify directory, paths and files to scan for preexisting components prior to the stack file creation.

- TeamInsight Viewer
  Use the TeamInsight Viewer to browse the web pages of the TeamInsight tools.

- Edit Repository Query or New XPlanner Query
  Queries an XPlanner repository by specifying a requested list of tasks.

- TeamInsight User Mail Notification
  Through the User Notification window, the ProjectAssist Administrator specifies e-mail message text and users to notify of the TeamInsight component availability.

- ProjectAssist Mail Preferences
  Enables the ProjectAssist Administrator mail account and configures mail preferences.

- Passwords for Authorization
  Allows the ProjectAssist Administrator to create passwords.

- Preinstalled Component Scan: Choose scan type
  Use the Preinstalled Component Scan: Choose scan type dialog to select the type of scan to perform on the server for preinstalled ProjectAssist components.

- ProjectAssist Configuration Editor: Projects
  To reach the Stacks, Users, and Projects tabs at the bottom of the ProjectAssist configuration editor, click on the ProjectAssist stack file name (.pacx extension).

- ProjectAssist Configuration Editor: Stacks
  To reach the Stacks, Users, and Projects tabs at the bottom of the ProjectAssist configuration editor, click on the ProjectAssist stack file name (.pacx extension).

- ProjectAssist Configuration Editor: Users
  To reach the Stacks, Users, and Projects tabs at the bottom of the ProjectAssist configuration editor, click on the ProjectAssist stack file name (.pacx extension).

- Maven Project from Archetype
  Specifies a new project using the Maven archetype wizard.

- New StarTeam Query
  Allows the definition of a new query against a StarTeam task or change request repository.

- StarTeam Repository Settings
  Specifies the settings for the StarTeam Mylar repository for change requests and/or tasks.
**New ProjectAssist File Link**

Creates a link to a ProjectAssist file. Open either the full or the ProjectAssist Administrator version of JBuilder 2007. Select File ▶️ New ▶️ Other ▶️ Team ▶️ ProjectAssist File Link. This opens the New ProjectAssist File Link wizard to link to an existing ProjectAssist stack file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File location</td>
<td>Specify the location of the existing ProjectAssist component stack file (which will have a .pacx file extension).</td>
</tr>
<tr>
<td>Browse</td>
<td>Click this button to browse to the file location for the existing ProjectAssist stack file.</td>
</tr>
</tbody>
</table>

**Related Concepts**

- [ProjectAssist and TeamInsight Overview](#)

**Related Tasks**

- [Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)](#)
- [Adding and Configuring Projects through ProjectAssist](#)
- [Adding and Configuring TeamInsight Users](#)
- [Mail Notification for TeamInsight Users](#)
- [Configuring Your TeamInsight Client](#)
New ProjectAssist File

File ➤ New ➤ Other ➤ Team ➤ ProjectAssist File

To create a new project assist developer stack file, open either the full or the ProjectAssist Administrator version of JBuilder 2007 through the menu path File ➤ New ➤ Other ➤ Team ➤ ProjectAssist File. This opens the New ProjectAssist file wizard to create a new ProjectAssist stack file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File name</td>
<td>Specifies the ProjectAssist component stack file name (which will have a .pacx file extension).</td>
</tr>
<tr>
<td>Administrator name</td>
<td>Specifies the ProjectAssist server Administrator name.</td>
</tr>
<tr>
<td>Initials</td>
<td>Specifies the ProjectAssist Administrator initials.</td>
</tr>
<tr>
<td>E-mail</td>
<td>Specifies the e-mail address of the ProjectAssist Administrator.</td>
</tr>
<tr>
<td>ID</td>
<td>Specifies the ProjectAssist Administrator alias ID.</td>
</tr>
<tr>
<td>Project name</td>
<td>Specifies a project name for the initially generated project.</td>
</tr>
<tr>
<td>Description</td>
<td>Provides additional description of the initial project.</td>
</tr>
<tr>
<td>ProjectAssist install directory</td>
<td>Provides (default value) or specifies (user-provided) root directory and path for the ProjectAssist installation.</td>
</tr>
<tr>
<td>ProjectAssist data directory</td>
<td>Provides (default value) or specifies (user-provided) root directory and path for ProjectAssist data files.</td>
</tr>
</tbody>
</table>

Related Concepts

ProjectAssist and TeamInsight Overview

Related Tasks

Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
Installing the ProjectAssist Components File
Installing the ProjectAssist Components File for JBuilder 2007 Enterprise Borland ALM Edition
Adding and Configuring Projects through ProjectAssist
Adding and Configuring TeamInsight Users
Mail Notification for TeamInsight Users
Configuring Your TeamInsight Client
New ProjectAssist File: Select Stack Components

File ➤ New ➤ Other ➤ Team ➤ ProjectAssist File ➤ Next

To create a new project assist developer stack file, open either the full or the ProjectAssist Administrator version of JBuilder 2007. Select File ➤ New ➤ Other ➤ Team ➤ ProjectAssist File. This opens the New ProjectAssist file wizard to create a new ProjectAssist stack file. Fill in the appropriate information and select Next to move to the New ProjectAssist file: Select Stack Components screen.

Note: The selections on this page are determined by your JBuilder 2007 Enterprise Edition. StarTeam choices appear only if you have the JBuilder 2007 Enterprise BALM Edition. Two columns in the Select Stack Components screen are titled New or existing installation and Existing installation only. The software component choices you can install for each function are listed in the appropriate column.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version Control System</td>
<td>Select the version control system to include component stack file. The choices are CVS, StarTeam and/or Subversion (depending on whether you have the JBuilder 2007 Enterprise or JBuilder 2007 Enterprise BALM Edition).</td>
</tr>
<tr>
<td>Continuous Build System</td>
<td>Select the build system for continuous integration to include in the component stack file.</td>
</tr>
<tr>
<td>Defect Tracker</td>
<td>Select the defect tracking or change request system to include in the component stack file. The choices are Bugzilla and/or StarTeam (depending on whether you have the JBuilder 2007 Enterprise or JBuilder 2007 Enterprise BALM Edition).</td>
</tr>
<tr>
<td>Task Provider</td>
<td>Select the task provider system to include in the component stack file. The choices are and/or StarTeam and/or XPlanner (depending on whether you have the JBuilder 2007 Enterprise or JBuilder 2007 Enterprise BALM Edition).</td>
</tr>
</tbody>
</table>

Related Concepts

ProjectAssist and TeamInsight Overview

Related Tasks

Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
Installing the ProjectAssist Components File
Installing the ProjectAssist Components File for JBuilder 2007 Enterprise Borland ALM Edition
Adding and Configuring Projects through ProjectAssist
Adding and Configuring TeamInsight Users
Mail Notification for TeamInsight Users
Configuring Your TeamInsight Client
Maven Project from Archetype

To create a new project based on the Maven archetype, go to File » New » Project or Other » Maven » Maven Project from Archetype. This opens the Maven Project from Archetype wizard. Fill in the appropriate information and click Finish to create your new project.

Note: The ProjectAssist Administrator can also add a project based on the Maven archetype through the Project tabs of the ProjectAssist Configuration Editor.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Archetype Group Id</td>
<td>Select the Maven archetype group ID from the dropdown list.</td>
</tr>
<tr>
<td>Archetype Artifact Id</td>
<td>Select the Maven archetype artifact ID from the dropdown list.</td>
</tr>
<tr>
<td>Archetype Version</td>
<td>Select the Maven archetype version from the dropdown list.</td>
</tr>
<tr>
<td>Remote Repositories</td>
<td>Enter the URL of a remote repository in which to search for the specified Maven archetype.</td>
</tr>
<tr>
<td>Project Group Id</td>
<td>Enter the Maven Group ID for the project to be created.</td>
</tr>
<tr>
<td>Project Artifact Id</td>
<td>Enter the Maven artifact ID for the project to be created.</td>
</tr>
<tr>
<td>Project Folder</td>
<td>Enter the root project folder name for the new project. You can browse for a current folder by clicking on the Browse button.</td>
</tr>
<tr>
<td>Project Version</td>
<td>Select the product version from the dropdown menu.</td>
</tr>
<tr>
<td>Project Package</td>
<td>Specify the project package name for the project to be created.</td>
</tr>
</tbody>
</table>

Click Finish to create the specified project based on the Maven archetype model.

Related Concepts

ProjectAssist and TeamInsight Overview

Related Tasks

Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
Installing the ProjectAssist Components File
Installing the ProjectAssist Components File for JBuilder 2007 Enterprise Borland ALM Edition
Adding and Configuring Projects through ProjectAssist
Adding and Configuring TeamInsight Users
Mail Notification for TeamInsight Users
Configuring Your TeamInsight Client

Related Reference

ProjectAssist Configuration Editor: Projects
New ProjectAssist file: Choose disk scan paths

Use the Choose disk scan paths page of the New ProjectAssist File wizard to specify directory, paths and files to scan for preexisting components prior to the stack file creation.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A:\</td>
<td>Choose paths and folders in the A:\ directory to scan (all or individual folders and files) for preexisting components.</td>
</tr>
<tr>
<td>C:\</td>
<td>Choose paths and folders in the C:\ directory to scan (all or individual folders and files) for preexisting components. This is the default selection.</td>
</tr>
<tr>
<td>D:\</td>
<td>Choose paths and folders in the D:\ directory to scan (all or individual folders and files) for preexisting components.</td>
</tr>
</tbody>
</table>

Related Concepts

- ProjectAssist and TeamInsight Overview

Related Tasks

- Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
- Adding and Configuring Projects through ProjectAssist
- Adding and Configuring TeamInsight Users
- Mail Notification for TeamInsight Users
- Configuring Your TeamInsight Client
## TeamInsight Viewer

**Window ▶ Open TeamInsight Viewer ▶ <tool-name> or Open All**

The TeamInsight Viewer is a custom browser window that opens in the editor pane. You can open any one or all of the TeamInsight tools from the **Window** menu. The TeamInsight window contains its own navigation bar located at the top of the window, including an entry field for URLs.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigation Bar</td>
<td>Located at the top of the TeamInsight Viewer, the navigation bar contains several buttons, such as: Home, Back, Forward, Stop, Refresh, Go to URL, and Go to home location for application. The <strong>Go to home location for application</strong> (an icon of a ringed planet) is useful for returning to the configured application after you visit another URL in the TeamInsight Viewer.</td>
</tr>
<tr>
<td>Window</td>
<td>The window area on the TeamInsight Viewer is a tabbed window for displaying the web pages of TeamInsight tools. To scroll the TeamInsight Viewer, use the scroll bars at the right-hand side of the viewer. After you open the Liferay portal in the TeamInsight Viewer, you can click links in several of the portlets (such as Continuum and XPlanner) to display the main web page of the application server that generates the portlet.</td>
</tr>
<tr>
<td>Tabs</td>
<td>The TeamInsight Viewer window has a tab for each TeamInsight tool that you have opened.</td>
</tr>
</tbody>
</table>

**Related Concepts**

- [ProjectAssist and TeamInsight Overview](#)
- [Liferay: The TeamInsight Project Portal](#)

**Related Tasks**

- [Opening the TeamInsight Viewer and the Liferay Portal](#)
- [Configuring Your TeamInsight Client](#)
Edit Repository Query or New XPlanner Query

To edit an existing query (or enter all-new values): In the Task List, either double-click an XPlanner repository or right-click an XPlanner repository and select Open from the context menu. The Edit Repository Query dialog box is displayed.

To create a new query: In the Task List, right-click anywhere and select New Query.

Mylar displays a preliminary New Repository Query dialog box ("Add or modify repository query"). On the Mylar dialog box, you can select from the available XPlanner and Bugzilla repositories, as configured in ProjectAssist and displayed in the TeamInsight Viewer, or you can click Add Task Repository to connect to another repository. (Mylar connects to several types of repositories.) Then the New XPlanner Query dialog box is displayed.

Note: A tree structure of the projects, iterations, and stories in the XPlanner repository appears in both of these dialog boxes. You can select to find either stories or tasks, and you can search a project, an iteration, or a specific story. The repository query finds either all your current tasks or only those tasks that match the query, and lists the resulting tasks in the Task List.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query name</td>
<td>Enter a name to identify this query and its results.</td>
</tr>
<tr>
<td>All my current tasks</td>
<td>Finds all your current tasks within the selected XPlanner repository.</td>
</tr>
<tr>
<td>Selected tasks</td>
<td>Finds only the tasks that meet the values you have specified in the subfields.</td>
</tr>
<tr>
<td>Projects, Iterations, and User Stories</td>
<td>Either select the name of your project or expand the directory listing and select the correct iteration or user story that you want to search.</td>
</tr>
<tr>
<td>Grouping</td>
<td>Select the grouping you want: Tasks or User Stories.</td>
</tr>
<tr>
<td>Scope</td>
<td>Select either All (meaning all tasks in the selected project, iteration, or user story) or My (meaning only your tasks).</td>
</tr>
</tbody>
</table>

Related Concepts

Liferay: The TeamInsight Project Portal
XPlanner: Project and Team Management

Related Tasks

Opening the TeamInsight Viewer and the Liferay Portal
TeamInsight User Mail Notification

Use this dialog box to select TeamInsight users to notify and send a prepared (or edited) notification message.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject text</td>
<td>Fill in your Subject line text or use the default text provided.</td>
</tr>
<tr>
<td>Message body</td>
<td>Defaults to prefilled e-mail message text to send (in HTML format). You can edit this to your own message or accept the default text.</td>
</tr>
<tr>
<td>Users to be notified</td>
<td>Lists the users to be notified in this mail message. Add or remove users for notification by clicking on the <strong>Add</strong> or <strong>Remove</strong> button.</td>
</tr>
<tr>
<td>Configure Mail . . .</td>
<td>Allows you to configure your e-mail system to send this e-mail (if it has not already been done).</td>
</tr>
<tr>
<td>Send Notification</td>
<td>Attaches a configuration file for the TeamInsight user to configure the client machine for access to the ProjectAssist component servers. It then sends the notification message to specified recipients.</td>
</tr>
<tr>
<td>Cancel</td>
<td>Click this button to cancel sending the notification message.</td>
</tr>
</tbody>
</table>

**Note:** After each TeamInsight user additions, the ProjectAssist Administrator is asked if the notification message is to be sent. The Administrator then has access to this dialog window if a notification message is to be sent.

**Related Concepts**
- ProjectAssist and TeamInsight Overview

**Related Tasks**
- Mail Notification for TeamInsight Users
- Adding and Configuring TeamInsight Users
- Configuring Your TeamInsight Client
ProjectAssist Mail Preferences

Window ▶ Preferences ▶ Mail

Enabling mail can also be initiated through the Send Mail Notification icon on the ProjectAssist configuration designer.

Use the Preferences Mail dialog box to enable the ProjectAssist Administrator's mail account prior to sending e-mail notifications to users.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Mail</td>
<td>Click the check box to enable ProjectAssist e-mail settings.</td>
</tr>
<tr>
<td>Sender name</td>
<td>Enter the Administrator (sender) name in the text area.</td>
</tr>
<tr>
<td>Sender email address</td>
<td>Enter the Administrator (sender) e-mail address in the text area.</td>
</tr>
<tr>
<td>SMTP server address</td>
<td>Enter the address of the Simple Mail Transport Protocol (SMTP) server for the Administrator's e-mail.</td>
</tr>
<tr>
<td>Use custom SMTP port</td>
<td>Click the check box to use a custom SMTP port.</td>
</tr>
<tr>
<td>SMTP server port</td>
<td>If the Use custom SMTP port check box is selected, Enter the server information in the SMTP server port field.</td>
</tr>
<tr>
<td>Server requires authentication</td>
<td>Click the check box if the server requires user authentication</td>
</tr>
<tr>
<td>Name</td>
<td>If the Server requires authentication check box is selected, enter the user name for server authentication.</td>
</tr>
<tr>
<td>Password</td>
<td>If the Server requires authentication check box is selected, enter the password for server authentication.</td>
</tr>
<tr>
<td>POP before SMTP required</td>
<td>Check the POP before SMTP required to route through a POP server.</td>
</tr>
<tr>
<td>Host</td>
<td>If the POP before SMTP required check box is selected, enter the POP Host identifier in the text box.</td>
</tr>
<tr>
<td>Name</td>
<td>If the POP before SMTP required check box is selected, type the user Name in the text box.</td>
</tr>
<tr>
<td>Password</td>
<td>If the POP before SMTP required check box is selected, type the user password in the text box.</td>
</tr>
<tr>
<td>Send Test Message</td>
<td>Click Send Test Message to confirm the configured mail preferences via a test message.</td>
</tr>
<tr>
<td>Restore Defaults</td>
<td>Click the Restore Defaults button to restore the mail preferences setting to a default state.</td>
</tr>
<tr>
<td>Apply</td>
<td>Click the Apply button to enable your settings.</td>
</tr>
</tbody>
</table>

Related Concepts
- ProjectAssist and TeamInsight Overview

Related Tasks
- Mail Notification for TeamInsight Users
- Configuring Your TeamInsight Client
Passwords for Authorization

<file>.paccx  ▶  Install ProjectAssist icon

Specifies passwords for the Administrator for all the ProjectAssist server components. The Password for Authorization dialog pops up when you initially click on the Install ProjectAssist icon on any of the ProjectAssist Designer configuration editor windows (Stacks, Users, or Projects).

| Item                                           | Description                                                                 |
|                                               |                                                                           |
| Use the same passwords when appropriate        | Check this box if you want to use the same passwords for all components. All passwords text areas are filled when you type in the initial password. The default value is checked. |
| Administrator ID                              | for each component, this area is prefilled with the ProjectAssist Administrator's username for log in to that component. |
|                                               | For Bugzilla and Liferay, the Administrator ID is username@somewhere.com (an E-mail address). |
|                                               | For Continuum, Subversion and XPlanner, the Administrator ID is username. |
| Password                                       | Specifies the Administrator password for each ProjectAssist component. If the Use the same password when appropriate box is checked, the same password is used when you type the initial password into the text area. |
| Confirm password                              | Enter the Administrator's password for the component again to confirm. |
| Validate Passwords                            | Click to validate that all passwords entered are in the correct syntax. |
| Install                                        | Click to install the passwords and components. |

Related Concepts

ProjectAssist and TeamInsight Overview

Related Tasks

Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
Installing the ProjectAssist Components File
Adding and Configuring Projects through ProjectAssist
Mail Notification for TeamInsight Users
Configuring Your TeamInsight Client
Preinstalled Component Scan: Choose scan type

File ➤ New ➤ Other ➤ Team ➤ Project Assist File

Use the Preinstalled Component Scan: Choose scan type page of the New ProjectAssist File wizard to select the type of scan to perform for any preinstalled ProjectAssist components on the installation server.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Components to scan for</td>
<td>Lists all the components that are included in the preinstallation scan.</td>
</tr>
<tr>
<td>Skip system scan</td>
<td>Check this button to skip the system scan.</td>
</tr>
<tr>
<td>Minimal system scan (system path, services and running processes)</td>
<td>Check this button to perform a minimal scan of your system.</td>
</tr>
<tr>
<td>Thorough system scan (disk, system path, services and running processes)</td>
<td>Check this button to perform a complete scan of your system. This is the default value.</td>
</tr>
</tbody>
</table>

Related Concepts

ProjectAssist and TeamInsight Overview

Related Tasks

Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
Adding and Configuring Projects through ProjectAssist
Adding and Configuring TeamInsight Users
Mail Notification for TeamInsight Users
Configuring Your TeamInsight Client
ProjectAssist Configuration Editor: Projects

<file>.pacx  Projects tab

Click the Projects tab to navigate to the Projects configuration editor where you add and configure projects for your team. The Sample Project generated during the stack file creation should already be in the Projects list.

Note: The individual TeamInsight component fields in the Projects tab are determined by the component selected during the ProjectAssist stack file installation. The descriptions below are for all components; your tab will only have choice for your defined TeamInsight components.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Projects</td>
<td>List the names of projects to be added. The Sample Project generated during the stack file creation should already be in the list.</td>
</tr>
<tr>
<td>Clone</td>
<td>Click a project name in the Projects area. Click Clone to create a project with the same configuration information. Replace the generic information with any project specific information.</td>
</tr>
<tr>
<td>Add</td>
<td>Click Add to create a project with the default values assigned. Replace the generic filler information with user-specific information.</td>
</tr>
<tr>
<td>Remove</td>
<td>Select a project in the Projects area and click Remove to remove the project.</td>
</tr>
<tr>
<td>Name</td>
<td>Enter the new project name or use the default name.</td>
</tr>
<tr>
<td>Description</td>
<td>Enter a description of the project.</td>
</tr>
<tr>
<td>Project Content Provider</td>
<td>Select a project content provider from the dropdown list. The type of projects provided in the Project Content dropdown list is determined by the TeamInsight component tools installed. The following choices may be present:</td>
</tr>
<tr>
<td>Group Id</td>
<td>Enter the Maven group ID.</td>
</tr>
<tr>
<td>Artifact Id</td>
<td>Enter the Maven artifact ID.</td>
</tr>
<tr>
<td>Archetype group Id (Maven archetype project)</td>
<td>Select the Maven archetype group ID from the dropdown list.</td>
</tr>
<tr>
<td>Archetype Artifact Id (Maven archetype project)</td>
<td>Select the Maven archetype artifact ID from the dropdown list.</td>
</tr>
<tr>
<td>Archetype Version (Maven archetype project)</td>
<td>Select the Maven archetype version from the dropdown list.</td>
</tr>
</tbody>
</table>

Select **Maven2 Archetype Project** to generate a Maven2 project using a Maven archetype model for quick generation of a Maven project. If a Maven 2 Archetype Project is selected, the Project Content area includes fields to specify aspects of the Maven project you are trying to create.

Select **Maven2 Sample Project** to generate a Maven2 sample project with POM files.

Select **Existing Project Directory** to link to an existing project with a pom.xml file on a local directory.

Select **Project checked into version control** if you are assimilating an existing CVS or Subversion installation and this project has a pom.xml file.

Select **Project checked into Subversion and uploaded to Continuum** if you are assimilating both the Subversion and Continuum installation and you want to specify the name of a project on the Continuum server.
Remote Repositories (Maven archetype project)
Enter the URL of a remote repository in which to search for the specified Maven archetype.

Project Group Id (Maven archetype project)
Enter the Maven Group ID for the project to be created.

Project Artifact Id (Maven archetype project)
Enter the Maven artifact ID for the project to be created.

Project Version (Maven archetype project)
Select the product version from the dropdown menu.

Project Package (Maven archetype project)
Specify the project package name for the project to be created.

SVN Repository path
Enter the repository path information for Subversion.

Path in repository
Enter the CVS repository path field in the Path in repository field. This is verified against existing paths and module names. Therefore, on the host machine, the project is in "repository path" plus "path in repository" (for example, "/public/SampleProject").

CVS vendor tag
Enter the required Vendor tag information in the Vendor tag when importing a project. An attempt is made to derive the vendor tag from the administrator's email address, but you can change this field to any value. This field cannot be blank.

Bugzilla project (product) name
Enter the name for the related Bugzilla project file.

XPlanner project name
Enter the name for the related XPlanner project file.

Bugzilla project version
Enter the Bugzilla project version.

Bugzilla project component(s)
Enter the name(s) for the Bugzilla project component(s).

StarTeam Version Control
Enter the name of the StarTeam project in the Project field.
Enter the StarTeam view in the View field, or leave empty for the default view. Enter the StarTeam folder name in the Folder field, or leave empty for the default folder.

StarTeam Change Requests
Enter the name of the StarTeam project in the Project field.
Enter the StarTeam view in the View field, or leave empty for the default view. Enter the StarTeam folder name in the Folder field, or leave empty for the default folder.

StarTeam Tasks
Enter the name of the StarTeam project in the Project field.
Enter the StarTeam view in the View field, or leave empty for the default view. Enter the StarTeam folder name in the Folder field, or leave empty for the default folder.

Install Developer Stacks
Click on this icon in the upper-right of the page to install after all projects and users have been added.

Uninstall Developer Stacks
Uninstalls components.

Send Mail Notification
Enables mail (if not previously enabled) and sends a notification message to users that new projects have been added.

Related Concepts

ProjectAssist and TeamInsight Overview

Related Tasks

Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
Installing the ProjectAssist Components File
Installing the ProjectAssist Components File for JBuilder 2007 Enterprise Borland ALM Edition
Adding and Configuring Projects through ProjectAssist
Adding and Configuring TeamInsight Users
Mail Notification for TeamInsight Users
Configuring Your TeamInsight Client
ProjectAssist Configuration Editor: Stacks

<file>.pacx  ▶ Stacks tab

Use the Stacks configuration editor to configure the individual components installed in the ProjectAssist stack file installation. Click on the component name in the left-side list on this page to bring up configuration information for that component.

There are also other categories on the Stacks page such as Shared Components, Settings and so forth although you may not need to change the default settings in these categories.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Install on local machine</td>
<td>Check this box for any component (Subversion, Continuum, Bugzilla, or XPlanner) to install the component on the local ProjectAssist server. This box is grayed out (not selectable) for assimilate-only ProjectAssist components (CVS or StarTeam). This creates a General area (for all components) with information on the Name, Description (all components), Installation Directory (Subversion, Continuum, and Bugzilla) and Data Directory (Subversion only). Bugzilla also has an SMTP server field that specifies the Bugzilla mail server name and port. Continuum also shows fields specifying Continuum HTTP port, Continuum RPC port and Windows service name.</td>
</tr>
<tr>
<td>Refer to an existing installation (local or remote)</td>
<td>Check this box for any component (Subversion, Continuum, Bugzilla, or XPlanner) to assimilate an existing component install (local or remote). For CVS and StarTeam components, this box is checked by default and cannot be changed. On the Subversion page, enter the existing component installation location in the URL field. The Admin username field defaults to the Administrator's name. Enter the Subversion password in the Password field. You can enable the ability to add users remotely to Subversion by checking the Enable add users to remote server box. On the Continuum page, enter the existing component installation location in the URL field. Enter the port number in the Continuum RPC port field. The Admin username field defaults to the Administrator's name. Enter the Continuum password in the Password field. On the Bugzilla page, enter the existing component installation location in the URL field. The Email Address field defaults to the Administrator's E-mail address. Enter the Bugzilla password in the Password field. On the XPlanner page, enter the existing component installation location in the URL field. The ProjectAssist Administrator Admin username field defaults to the Administrator's name. Enter the XPlanner password in the Password field.</td>
</tr>
</tbody>
</table>

Tip: On each component configuration page, click Test Connection to validate the configuration information.
<table>
<thead>
<tr>
<th>Install Developer Stacks</th>
<th>Click on this icon in the upper-right of the page to install the stack components after configuration.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uninstall Developer Stacks</td>
<td>Click on this icon in the upper-right of the page to uninstall the stack components.</td>
</tr>
<tr>
<td>Send Mail Notification</td>
<td>Do not use this icon on the <strong>Stacks</strong> page.</td>
</tr>
</tbody>
</table>

**Related Concepts**

- ProjectAssist and TeamInsight Overview

**Related Tasks**

- Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
- Installing the ProjectAssist Components File
- Installing the ProjectAssist Components File for JBuilder 2007 Enterprise Borland ALM Edition
- Adding and Configuring Projects through ProjectAssist
- Adding and Configuring TeamInsight Users
- Mail Notification for TeamInsight Users
- Configuring Your TeamInsight Client
Click the Users tab to navigate to the Users configuration editor to add and configure users for your project and the components. The Administrator created initially should already be in the User List. When you click on the Administrator's name, the General Information frame appears with user details and permissions for the Administrator.

### Item Description

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User List</td>
<td>The Administrator created initially should already be in the User List. If you click the Administrator name, the General Information frame appears with user details and permissions for the Administrator. Names are added to this list as you clone or add more users.</td>
</tr>
<tr>
<td>Clone</td>
<td>Click a user name in the User List area. Click Clone to create a user with the same assigned user roles as the already defined user. Replace the generic filler information with user-specific information.</td>
</tr>
<tr>
<td></td>
<td>To change the user's role for any TeamInsight component, right-click on the component in the Roles list. The roles of Administrator, Developer, or No Access can be assigned for each user according to components.</td>
</tr>
<tr>
<td>Add</td>
<td>Click Add to create a user with the default roles assigned to all the ProjectAssist components. Replace the generic filler information with user-specific information. The default role assignment for all components (except MySQL) is Developer. The default for MySQL (used by the Bugzilla component) is No Access.</td>
</tr>
<tr>
<td></td>
<td>To change the user's role for any TeamInsight component, right-click on the component in the Roles list. The roles of Administrator, Developer, or No Access can be assigned for each user according to components.</td>
</tr>
<tr>
<td>Remove</td>
<td>Select a user in the User List area and click Remove to remove a user prior to the clicking the Install Developer Stacks icon.</td>
</tr>
<tr>
<td></td>
<td>Users cannot be removed after they have been added by clicking on the Install Developer Stacks icon. Be sure the information is correct before installing.</td>
</tr>
<tr>
<td>Install Developer Stacks</td>
<td>Click on this icon in the upper-right of the page to install the users after configuration to the component servers.</td>
</tr>
<tr>
<td></td>
<td>Users cannot be removed after they have been added with the Install Developer Stacks icon. Be sure the information is correct before clicking this icon.</td>
</tr>
<tr>
<td>Uninstall Developer Stacks</td>
<td>Is not applicable to the Users page.</td>
</tr>
<tr>
<td>Send Mail Notification</td>
<td>Enables mail and sends a notification message to users after they are added.</td>
</tr>
</tbody>
</table>

### Related Concepts
- ProjectAssist and TeamInsight Overview

### Related Tasks
- Installing JBuilder 2007 and the ProjectAssist Server (Administrator Task)
- Installing the ProjectAssist Components File
- Adding and Configuring Projects through ProjectAssist
- Adding and Configuring TeamInsight Users
- Mail Notification for TeamInsight Users
- Configuring Your TeamInsight Client
New StarTeam Query

Window ➤ Configure Mylar ➤ StarTeam project name ➤ change request or task view

(JBuilder 2007 Enterprise BALM Edition): JBuilder 2007 enables you to add the StarTeam repository change requests and StarTeam repository tasks to the Eclipse Task List view, and to use Mylar to define queries against those repositories.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Name</td>
<td>Enter a name for your query.</td>
</tr>
<tr>
<td>All my current tasks and change requests</td>
<td>Select this button if you wish to see both repository queries for change requests and tasks.</td>
</tr>
<tr>
<td>Selected tasks or change requests</td>
<td>Check this button to select tasks or change requests for specific project/views/folders.</td>
</tr>
<tr>
<td>Type</td>
<td>Select whether to include tasks or change requests, or both, in your query. If you select Tasks in the Type field, a single query node is created in the Task List view, with all the applicable tasks sublisted. If you select Change Requests in the Type field, a single query node is created in the Task List view, with all the applicable change requests sublisted. Selecting both generates two lists in the Task List view.</td>
</tr>
<tr>
<td>Scope</td>
<td>Specifies the scope of control over the type. If you select All in this group, all tasks or change requests from selected StarTeam entities are added to the query results. If you select My, then only your own tasks or change requests are added to the query results.</td>
</tr>
</tbody>
</table>

Related Concepts

- ProjectAssist and TeamInsight Overview
- Mylar Concepts

Related Tasks

- Adding Mylar Repositories for Bugzilla and XPlanner
- Configuring Your TeamInsight Client

Related Reference

- StarTeam Repository Settings
- External Documentation for Mylar from Eclipse.org
- External Documentation about Mylar Connectors to Repositories
- External Article: Task-Focused Programming with Mylar
StarTeam Repository Settings

Use the **StarTeam Repository Settings** dialog for Mylar-based repositories for StarTeam change requests and/or tasks.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server <a href="">address:port</a></td>
<td>Select the address of the StarTeam repository server in the dropdown list. The address is in the format address:port.</td>
</tr>
<tr>
<td>Label</td>
<td>Enter the label for the StarTeam repository.</td>
</tr>
<tr>
<td>User ID</td>
<td>Enter the User ID for the authorized StarTeam user.</td>
</tr>
<tr>
<td>Password</td>
<td>Enter the password for the User ID.</td>
</tr>
<tr>
<td>Default location</td>
<td>Enter or browse for the default location of the StarTeam repository. This is the location to be searched for all entities of this repository.</td>
</tr>
<tr>
<td>StarTeam Repository Type</td>
<td>Select the type of repository you would like to establish. If you select Tasks in the <strong>Type</strong> field, a single query node is created in the Task List view, with all the applicable tasks sublisted. If you select Change Requests in the <strong>Type</strong> field, a single query node is created in the <strong>Task List</strong> view, with all the applicable change requests sublisted. Selecting both generates two lists in the <strong>Task List</strong> view.</td>
</tr>
<tr>
<td>Character Encoding</td>
<td>Use the default character encoding of UTF-8 or click <strong>Other</strong> and select another encoding method from the dropdown list.</td>
</tr>
<tr>
<td>Validate Settings</td>
<td>Click the <strong>Validate Settings</strong> buttons to verify that all your settings are correct.</td>
</tr>
</tbody>
</table>

**Related Concepts**

- ProjectAssist and TeamInsight Overview
- Mylar Concepts

**Related Tasks**

- Adding Mylar Repositories for Bugzilla and XPlanner
- Configuring Your TeamInsight Client

**Related Reference**

- New StarTeam Query
- External Documentation for Mylar from Eclipse.org
- External Documentation about Mylar Connectors to Repositories
- External Article: Task-Focused Programming with Mylar
Peer to Peer Dialogs Reference

This section lists the dialog/wizards information for peer to peer interaction provided through JBuilder 2007.

In This Section

Peer To Peer Preferences
Sets preferences for peer to peer collaboration.

Peers View
Opens peer to peer sessions, manages chats, sends and receives files, web links and stack traces.

New Contact Group
Creates a new contact group.

Send Stack Trace
Sends a stack trace to a peer.

Send Web Link
Sends a web link to a peer during a collaboration session.

Send VCS Link
Sends a link to a peer for a project checked out from a Version Control System (VCS).
# Peer To Peer Preferences

Use this dialog box to set preferences for collaborating with peers.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Peer to Peer Subsystem</td>
<td>Enables the peer to peer features and opens the Peers view when you click the Apply button.</td>
</tr>
<tr>
<td>Name</td>
<td>The name you want to display to peers. This defaults to your user name.</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description that can help identify you to peers.</td>
</tr>
<tr>
<td>Image</td>
<td>An optional icon that helps identify you in a peer to peer collaboration session. The following file types are accepted: .GIF, .JPEG, and .PNG.</td>
</tr>
<tr>
<td>Browse</td>
<td>Displays the Open dialog box, where you browse to the location of an image to use for identification. Any icon you use is resized to 48 x 48 pixels. This may distort the image.</td>
</tr>
<tr>
<td>Filtering</td>
<td>The adapter to use. Select NONE if you have only one adapter or want to be prompted at peer to peer startup for the adapter.</td>
</tr>
<tr>
<td>Log Chat Messages</td>
<td>Enables logging of chat messages to a file.</td>
</tr>
<tr>
<td>Workspace Directory</td>
<td>The Eclipse workspace folder in which to save chat logs</td>
</tr>
<tr>
<td>Incoming Message Color</td>
<td>The color for incoming messages.</td>
</tr>
<tr>
<td>Outgoing Message Color</td>
<td>The color for outgoing messages.</td>
</tr>
<tr>
<td>Status Message Color</td>
<td>The color for status messages.</td>
</tr>
<tr>
<td>Automatic Receive Enabled</td>
<td>Enables automatic file transfer and allows a file sent from a peer to be automatically received, rather than downloaded manually.</td>
</tr>
<tr>
<td>Workspace Directory</td>
<td>The Eclipse workspace folder to save files to when automatic receive is enabled.</td>
</tr>
<tr>
<td>Audio Feedback Enabled</td>
<td>Enables audio feedback. There are different sounds for incoming messages and incoming status information.</td>
</tr>
<tr>
<td>Slider</td>
<td>Adjusts the audio feedback volume.</td>
</tr>
</tbody>
</table>

**Related Concepts**

- [Peer to Peer Collaboration](#)

**Related Tasks**

- [Setting Collaboration Preferences](#)
- [Opening a Peer to Peer Session](#)

**Related Reference**

- [Peers View](#)
- [New Contact Group](#)
- [Send Stack Trace](#)
- [Send Web Link](#)
- [Send VCS Link](#)
Peers View

Window ▶ Show View ▶ Other ▶ Peer to Peer ▶ Peers

The Peers view is where you discover peers, choose the peer(s) you want to chat with, create and manage contact groups, chat with peers, and send data to peers. The Peers view contains the Peers pane on the left and the Collaboration pane on the right.

Peers Pane

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Your current status: Available, Away, or Offline.</td>
</tr>
<tr>
<td>IP Address</td>
<td>Your IP address; used for identification. The IP address is shown when you are online.</td>
</tr>
<tr>
<td>Available Local Peers</td>
<td>The list of available peers.</td>
</tr>
<tr>
<td>Contact Groups</td>
<td>Your contact groups. Peers assigned to each contact group are also displayed.</td>
</tr>
</tbody>
</table>

Collaboration Pane

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peer in Session</td>
<td>The name(s) of the peer(s) in the chat session.</td>
</tr>
<tr>
<td>Chat Area</td>
<td>The chat.</td>
</tr>
<tr>
<td>Message Area</td>
<td>The message input area</td>
</tr>
<tr>
<td>Session Tab</td>
<td>The representation of the session. To close the session, click the X on the tab.</td>
</tr>
</tbody>
</table>

The Collaboration pane toolbar contains buttons for:

- Adding peer(s) to the chat session
- Sending a file to peer(s)
- Sending a web link to peer(s)
- Sending a stack trace to peer(s)
- Closing all chat sessions

Related Concepts

- Peer to Peer Collaboration

Related Tasks

- Enabling Peer to Peer Collaboration
- Opening a Peer to Peer Session
- Chatting with Peers
- Sending Data To Peers

Related Reference

- New Contact Group
- Peer To Peer Preferences
- Send Stack Trace
- Send Web Link
- Send VCS Link
New Contact Group

Use this dialog box to create a name for a contact group.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Name</td>
<td>Enter the contact group name.</td>
</tr>
</tbody>
</table>

Related Concepts

- Peer to Peer Collaboration

Related Tasks

- Managing Contact Groups

Related Reference

- Peers View
- Peer To Peer Preferences
- Send Stack Trace
- Send Web Link
- Send VCS Link
Send Stack Trace

Use this dialog box to send a stack trace to a peer.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stack Trace</td>
<td>The stack trace to send. Paste the stack trace from the Clipboard.</td>
</tr>
</tbody>
</table>

Related Concepts

Peer to Peer Collaboration

Related Tasks

Send Stack Trace

Related Reference

Peers View
New Contact Group
Peer To Peer Preferences
Send Stack Trace
Send Web Link
Send VCS Link
Send Web Link

Collaboration pane toolbar ➤ Send Web Link to Peers in Collaboration icon

Use this dialog box to specify a web link to send to a peer during a collaboration session.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web link</td>
<td>The URL of the web link to send. Click <strong>OK</strong> to send.</td>
</tr>
</tbody>
</table>

Related Concepts

- Peer to Peer Collaboration

Related Tasks

- Sending Data To Peers

Related Reference

- Peers View
- New Contact Group
- Peer To Peer Preferences
- Send Stack Trace
- Send VCS Link
Send VCS Link

VCS projectname (right-click) ➤ Send VCS Link to Peer

Projects are shared through a repository. When projects are shared, the Navigator or Package Explorer displays the project repository and location. You can send your peers a link to the VCS project repository by right-clicking on the project name and selecting Send VCS Link to Peer. This opens the Select Peers dialog.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Available peers</td>
<td>Lists all available peers to whom you can send the VCS link. Click Select to send. The project is sent as a VCS link to the selected peer. The message Sending VCS link for project “&lt;Project Name&gt;” is displayed in your chat area.</td>
</tr>
</tbody>
</table>

Related Concepts

Peer to Peer Collaboration

Related Tasks

Sharing Team-Enabled Projects with Peers
Sending Data To Peers

Related Reference

Peers View
New Contact Group
Peer To Peer Preferences
Peers View
New Contact Group
Peer To Peer Preferences
Send Stack Trace
Send Web Link