ptc

integrity™ lifecycle manager

Eclipse Software Development Platform Integration Guide

11.1

Copyright © 2017 PTC Inc. and/or Its Subsidiary Companies. All Rights Reserved.

User and training guides and related documentation from PTC Inc. and its subsidiary companies (collectively "PTC") are subject to the copyright laws of the United States and other countries and are provided under a license agreement that restricts copying, disclosure, and use of such documentation. PTC hereby grants to the licensed software user the right to make copies in printed form of this documentation if provided on software media, but only for internal/personal use and in accordance with the license agreement under which the applicable software is licensed. Any copy made shall include the PTC copyright notice and any other proprietary notice provided by PTC. Training materials may not be copied without the express written consent of PTC. This documentation may not be disclosed, transferred, modified, or reduced to any form, including electronic media, or transmitted or made publicly available by any means without the prior written consent of PTC and no authorization is granted to make copies for such purposes. Information described herein is furnished for general information only, is subject to change without notice, and should not be construed as a warranty or commitment by PTC. PTC assumes no responsibility or liability for any errors or inaccuracies that may appear in this document.

The software described in this document is provided under written license agreement, contains valuable trade secrets and proprietary information, and is protected by the copyright laws of the United States and other countries. It may not be copied or distributed in any form or medium, disclosed to third parties, or used in any manner not provided for in the software licenses agreement except with written prior approval from PTC.

UNAUTHORIZED USE OF SOFTWARE OR ITS DOCUMENTATION CAN RESULT IN CIVIL DAMAGES AND CRIMINAL PROSECUTION.

PTC regards software piracy as the crime it is, and we view offenders accordingly. We do not tolerate the piracy of PTC software products, and we pursue (both civilly and criminally) those who do so using all legal means available, including public and private surveillance resources. As part of these efforts, PTC uses data monitoring and scouring technologies to obtain and transmit data on users of illegal copies of our software. This data collection is not performed on users of legally licensed software from PTC and its authorized distributors. If you are using an illegal copy of our software and do not consent to the collection and transmission of such data (including to the United States), cease using the illegal version, and contact PTC to obtain a legally licensed copy.

Important Copyright, Trademark, Patent, and Licensing Information: See the About Box, or copyright notice, of your PTC software.

UNITED STATES GOVERNMENT RIGHTS

PTC software products and software documentation are “commercial items” as that term is defined at 48 C.F. R. 2.101. Pursuant to Federal Acquisition Regulation (FAR) 12.212 (a)-(b) (Computer Software) (MAY 2014) for civilian agencies or the Defense Federal Acquisition Regulation Supplement (DFARS) at 227.7202-1(a) (Policy) and 227.7202-3 (a) (Rights in commercial computer software or commercial computer software documentation) (FEB 2014) for the Department of Defense, PTC software products and software documentation are provided to the U.S. Government under the PTC commercial license agreement. Use, duplication or disclosure by the U.S. Government is subject solely to the terms and conditions set forth in the applicable PTC software license agreement.

PTC Inc., 140 Kendrick Street, Needham, MA 02494 USA

Contents

1

About This Guide

You can integrate Integrity Lifecycle Manager with the Eclipse Platform. This allows you to access Integrity Lifecycle Manager version control commands that are based on the Eclipse open source initiative.

Note

- Integration with the Eclipse platform works only on Windows and Linux operating systems supported for the Integrity Lifecycle Manager client. For more information on supported operating systems, go to the PTC Integrity eSupport portal. The address is http://www.ptc.com/support/integrity.htm.
- As of Integrity Lifecycle Manager 11.0, the default installation directory of the Integrity Lifecycle Manager client changed. This change affects integrations that were installed with Integrity Client 2009 or earlier. For more information, see the *Integrity Lifecycle Manager Help Center*.

Eclipse Team Support provides flexibility in designing and implementing support of the repository on the workbench. In turn, the Integrity Lifecycle Manager integration is easier to use and provides greater control.

The following sections assist you in using the integration:

2

Overview

The Integrity Lifecycle Manager integration with the Eclipse Platform allows you to access Integrity Lifecycle Manager version control commands. You can integrate with several products that are based on the Eclipse open source initiative.

The integration also allows you to access Integrity Lifecycle Manager version control commands through several open source and commercially available development products, built as Eclipse Platform plug-ins.

The integration includes software developed by the Eclipse Project. For more information on the Eclipse Project, browse to http://www.eclipse.org.

Supported Versions

The integration with Eclipse Platform and is designed to work any product built on the Open Source Eclipse Platform.

Note

For more information on supported versions of Eclipse Platform, go to the PTC Integrity eSupport portal. The address is http://www.ptc.com/support/integrity.htm.

Before You Start

Before you use the integration, note the following:

- This guide assumes that you know how to use Eclipse Software Development Platform products, Integrity Lifecycle Manager, and Implementer (if working with Implementer change packages). For more information about using a product, refer to the appropriate documentation from the product vendor.
- Java Development Kit (JDK) 1.5 or higher must be installed.
- Read Best Practices on page 61 for more detailed information about using the Eclipse integration.
- If theIntegrity Lifecycle Manager client is shut down and you run a command that requires Integrity Lifecycle Manager, the Integrity Lifecycle Manager client automatically initializes.

3

Configuring Eclipse Integration

Integrations based on the Eclipse Platform include solutions such as RAD and RSA. You enable the integration using the **Add Site** function. For more information, see Enabling the Integrity Lifecycle Manager Integration with Eclipse 3.4 on page 13 and Enabling the Integrity Lifecycle Manager Integration with Eclipse 3.5+ on page 14.

Note

To use a new version of the Eclipse integration, you must configure Eclipse to perform updates automatically. Or, you can force a manual update to pick up the new version of the integration. In general, you can configure updates by selecting **Help ▸ Check for Updates**. For more information on updating software, see the Eclipse product documentation.

As of Integrity Lifecycle Manager 11.0, the default location of the Integrity Lifecycle Manager integration installation changed. You must point Eclipse to the new location as the update site. For example on Windows, the new default location is:

```
<installdir>\Integrity\ILMClient11\integrations\IBM\eclipse_3.4\eclipse
```

When uninstalling a previous version of the Integrity Lifecycle Manager client, you must also remove the previous update site within Eclipse. Otherwise, an error occurs after installing the new integration: `No repository found`.

If you are working with Implementer, the Integrity Lifecycle Manager server must also be configured to allow remote API connections to the server. For more information, see the *Integrity Lifecycle Manager Help Center*.

After enabling the integration, you can also set general preferences and additional options related to change packages and file annotations.

This section discusses the following topics:

Enabling Access to Java API Libraries (Linux Only)

On the Linux platform, the Eclipse integration requires access to the Java API libraries.

Set the `LD_LIBRARY_PATH` environment variable to the following:

```
export LD_LIBRARY_PATH=<Integrity Client installdir>/lib/linux
```

where <*Integrity Client installdir*> is the path to the Integrity Lifecycle Manager client installation location.

Enabling the Integration

To enable the Integrity Lifecycle Manager integration, refer to the instructions for the version with which you are working. Eclipse version 3.4 integration instructions differ from Eclipse version 3.5+.

Enabling the Integrity Lifecycle Manager Integration with Eclipse 3.4

To enable the Integrity Lifecycle Manager integration with Eclipse 3.4:

1. In Eclipse, select **Help** ▸ **Software Updates**. The **Software Updates and Add-ons** window opens.
2. Under the **Available Software** tab, click **Add Site**. The **Add Site** window opens.
3. To select the directory where you placed your plug-ins as an extension location, click **Local** and navigate to the following directory:

   ```
   <Integrity Client installdir>/integrations/IBM/eclipse_3.4/eclipse
   ```

   where <*Integrity Client installdir*> is the path to the directory where you installed the Integrity Lifecycle Manager client.
4. To enable the integration, check the box next to the newly added integration and then click **Install**. Eclipse copies the selected plug-in to the `Eclipse Features` and `Plugins` directory.
5. Restart the Eclipse workbench to enable the integration.

> **Note**
>
> You can also use Eclipse drop-ins to add a plug-in. For more information on drop-ins, consult the Eclipse product documentation.

Enabling the Integrity Lifecycle Manager Integration with Eclipse 3.5+

To enable the Integrity Lifecycle Manager integration with Eclipse 3.5+:

> **Note**
>
> The **Classic Update** method is not recommended for installing this version of the Eclipse integration. Before installing the integration, you can confirm the setting for this option by selecting **Window** ▸ **Preferences**. Under **General** ▸ **Capabilities**, confirm that the **Classic Update** option is cleared.

1. In Eclipse, select **Help** ▸ **Install New Software**. The **Install** window opens.
2. Click **Add**. The **Add Site** window opens.
3. Click **Local**. The **Browse For Folder** window opens.
4. Browse to the following folder on the Integrity Lifecycle Manager client machine:

   `<Integrity Client installdir>/integrations/IBM/eclipse_3.4/eclipse`

   where <*Integrity Client installdir*> is the path to the directory where you installed the Integrity Lifecycle Manager client.

   When you click **OK**, the selected folder displays in the **Location** field.
5. In the **Name** field, enter a name for the integration, such as `Integrity`. Then, click **OK**.
6. In the **Work with** list, select the Integrity Lifecycle Manager integration location.

> **Tip**
>
> If the Integrity Lifecycle Manager integration location does not automatically appear in the **Work with** list, click the link for **Available Software Sites** and select the site that you just added. The Integrity Lifecycle Manager integration then becomes available for selection in the **Work with** list.

7. From the list, select the **Collaboration** checkbox. The **Integrity Eclipse Integration** checkbox is automatically selected.
8. Click **Next**. The **Install Details** window opens with the Integrity Lifecycle Manager integration in the list of items to be installed.
9. To install the integration, click **Finish**.

10. Follow the remaining steps of the wizard to accept content and the license agreement.
11. In the **Software Updates** window, click **Yes** to restart the Eclipse workbench and complete the installation. When Eclipse restarts, the **Integrity** menu is available for selection.

Setting Preferences

Integration-specific preferences allow you to specify settings for Integrity Lifecycle Manager commands. The Integrity Lifecycle Manager preferences become the default.

In addition, **File Content** and **Ignored Resources** preferences affect functions in your workspace related to the integration. First, select **Window ▸ Preferences**.

Under **Team ▸ File Content**, you can specify a file type and the type of content in that particular file (binary or ASCII content).

Under **Team ▸ Ignore Resources**, you can specify resource name patterns that you do not want to add to Integrity Lifecycle Manager version control. In addition, Integrity Lifecycle Manager ignores any file that is marked **Derived**. To see if a file is marked **Derived**, check the file's properties.

Click **Restore Defaults** at any time to clear the changes you made.

Setting Integrity Lifecycle Manager Preferences

To set Integrity Lifecycle Manager preferences:

1. In your workspace, select **Window ▸ Preferences**. The **Preferences** window opens.
2. Under **Team**, select **Integrity Source**. The **Integrity Source** panel displays.
3. Make the necessary changes to the preferences:

> **Note**
> Any modifications to change package preferences require that you restart your Eclipse workspace.

- **Updating non-locked files**
  - **Lock file** performs the Integrity Lifecycle Manager **Lock** command, allowing you to edit the file. Enable this option to implicitly check out a file while editing it. This option is enabled by default.

> **Note**
>
> If you are using non-exclusive locks, you can lock a file. However, other users are not prevented from locking the same file.

- **Make file writable** performs the Integrity Lifecycle Manager **Make Working File Writable** command, allowing you to edit the working file, but not preventing other users from locking the file. Enable this option to edit a file locked by another user when you have no intention of checking the file back in. If you intend to check the file in later on, you can lock the file or associate the modified working file with a change package. You then submit your changes.

- **Annotation of Subprojects**

  The Eclipse integration provides specific annotations for shared subprojects, including development path names for variant projects and checkpoint revision numbers for build projects. When changes are made to a project or subproject configuration from outside the Eclipse integration, those changes are dynamically displayed within the integration.

  Only projects and subprojects under source control are annotated. Ordinary folders and subfolders are not annotated. In addition, to correspond with behavior in the Integrity Lifecycle Manager client, children of shared subprojects are not marked as shared.

  The integration also provides annotation for the configuration details of shared and configured subprojects. For variant subprojects, the annotation shows development path names. For build subprojects, the annotation shows checkpoint revision numbers. Normal subprojects configured within a build or variant subproject do not show any annotation.

  To set the type of annotation required, click to select from the following options under **Annotation of Subprojects**:

  - **Display configuration at project level only.**
  - **Annotate shared subprojects. (Default)**
  - **Annotate all subprojects with configuration information.**

> **Note**
>
> To avoid displaying more information than is generally required, the default setting provides annotation for shared subprojects only. After setting your preference options, you must restart Eclipse to have the new annotations displayed in the integration.

- **Use Change Package**

  The **Use Change Package** preference specifies the use of a default change package when performing Integrity Lifecycle Manager commands that use a change package. This preference is enabled by default.

  When this preference is enabled, all changes must be submitted using a change package. You no longer have the option of submitting individual changes using the **Submit Changes** command. To ensure the correct behavior, use the **Use Change Package** preference only if the policy for `ChangePackagesEnabled` is set to true on the .

- **When performing Integrity commands that require a change package, prompt for an active change package** provides for prompting when performing any software configuration management operation that requires a change package, if there is no active change package context. You can only enable this preference if the **Use Change Package** preference is enabled. The default setting for this option is false (disabled)

  When this preference is enabled, you are prompted to select or create a change package if the following conditions are present:

  - Active change package tracking is enabled.
  - No active change package set exists.

  Assume that you do not select or create a change package when first prompted. You are prompted again when you attempt further file operations that normally require a change package. Such operations include additional edits or saving the file. You are also prompted when you create or drop subprojects, move or rename files, add or drop files, and run the Integrity Lifecycle Manager **Sharing** wizard to add files.

> **Note**
>
> To return the expected prompt messages when obtaining locks and change packages, ensure that change package preferences are configured consistently between the Integrity Lifecycle Manager server and Eclipse. In other words, if change packages are mandatory in Eclipse, ensure that they are also configured as mandatory on the Integrity Lifecycle Manager server. If change package policies are not consistent, prompting could not occur as expected when locking and changing members.

- **Drop Integrity Sandbox when Eclipse project is deleted** specifies that if your Eclipse project is deleted, the associated Sandbox is also dropped. The default setting for this preference is false (disabled).

4. To set additional preferences, click **Integrity Preferences**. The **Preferences Configuration** window opens.
5. Click **Apply** to save the changes.
6. Click **OK** to close the **Preferences Configuration** window.

Setting File Content Preferences

To set File Content preferences:

1. In your workspace, select **Window** ▸ **Preferences**. The **Preferences** window opens.
2. Under **Team**, select **File Content**. The **File Content** preferences display.
3. To add a file type, click **Add Extension**. To add a file using a file name, click **Add Name**. To delete a file type from the list, click **Remove**. To modify the file type content, click **Change**.
4. To save the changes, click **Apply**.
5. To close the **Preferences** window, click **OK**.

Setting Ignored Resources Preferences

To set **Ignored Resources** preference:

1. In your workspace, select **Window** ▸ **Preferences**. The **Preferences** window opens.
2. Under **Team**, select **Ignored Resources**. The **Ignored Resources** preferences display.
3. To add a pattern, click **Add Pattern**. To delete a pattern from the list, click **Remove**.

4. To save the changes, click **Apply**.
5. To close the **Preferences** window, click **OK**.

Deactivating the Integration

To deactivate the Integrity Lifecycle Manager integration with Eclipse platform, you remove the installed Integrity Lifecycle Manager software plugin.

To deactivate the Eclipse integration:

1. Select **Help** ▸ **About Eclipse SDK**.
2. In the **About Eclipse SDK** window, click **Installation Details**. The **Eclipse SDK Installation Details** window opens.
3. Under the **Installed Software** tab, select **Integrity Eclipse Integration** and click **Uninstall**. The **Uninstall** window opens with the Integrity Lifecycle Manager integration in the list of items to be uninstalled.
4. To uninstall the integration, click **Finish**.

> **Note**
>
> As part of uninstalling the integration, you want to remove the Integrity Lifecycle Manager integration update site. Select **Window** ▸ **Preferences**. Then, under **Install/Update**, select **Available Software Sites**. In the **Available Software Sites** panel, highlight the Integrity Lifecycle Manager plugin location, and then click **Remove**.

4

Using the Integration

This section provides information on how to use the Integrity Lifecycle Manager integration with Eclipse. The following topics are discussed:

- Working in Online or Offline Mode on page 22
- Setting Up an Integrated Workspace on page 23
- Working in an Integrated Workspace on page 32
- Integrity Lifecycle Manager Commands on page 41
- Refactoring on page 50
- Comparing Revisions on page 51
- Team Synchronizing on page 52

Working in Online or Offline Mode

When you start Eclipse, theIntegrity Lifecycle Manager client automatically attempts to establish a connection with the Integrity Lifecycle Manager server. If the client cannot establish a connection with the server, you can still work in Eclipse.

When working in Eclipse in online mode without a server connection, all Integrity Lifecycle Manager views and commands that affect the Integrity Lifecycle Manager repository stay enabled. If you perform a command that requires a server connection, the Integrity Lifecycle Manager client attempts to reconnect, prompting you to enter your credentials if required.

If you know that the server is not going to be available for a long period of time, you can switch to working in offline mode. For example, if you are working remotely without an Internet connection, you can work in offline mode.

In offline mode, you can perform any commands that do not require a server connection, such as editing a file. For this task, the file is made writable, but it is not locked. Once you switch to online mode, all disabled Integrity Lifecycle Manager commands and views become active again. You can then resynchronize your changes with the Integrity Lifecycle Manager repository.

Note

You are automatically switched to offline mode in the following situations:

- You cancel your server connection window without entering any credentials.
- The Integrity Lifecycle Manager client becomes unavailable.

Switching Between Online and Offline Modes

To switch between online and offline modes, select **Integrity ▸ Work Online/Offline**. You can also click the online  icon and offline  icon in the Integrity Lifecycle Manager trim.

Note

The Integrity Lifecycle Manager trim displays in the bottom-right corner of the workbench by default. However, you can drag the trim anywhere in the workbench.

When switching from offline to online mode, note the following:

- Switching from offline to online mode refreshes the status of the workspace, retrieving the latest decorators. Depending on how many files or Eclipse projects are visible, this process can take a long time. However, it occurs in the background and does not prevent you from working in Eclipse.
- After switching from offline to online mode, you cannot submit changes in the Synchronize view while the Integrity Lifecycle Manager status cache refreshes. Before you submit changes, check the status of the Eclipse progress bar and ensure that there is no Integrity Lifecycle Manager command activity.

Working in Offline Mode

When working in offline mode, note the following:

- Information available in your resource cache before you switch to offline mode determines whether file decorators update when you select files or make changes to them. Locks made by other users, new revisions, and incoming changes do not update file decorators.
- You cannot move outgoing changes to change packages.
- You cannot rename or move files that are under source control. You cannot rename or move files that are not under Integrity Lifecycle Manager version control, but exist in an Eclipse project that is under Integrity Lifecycle Manager version control. You can refactor (add, drop, rename, or move) ignored resources.
- Integrity Lifecycle Manager information does not display in property windows.
- If you delete a subproject, the Sandbox icon is removed from the subproject. While in offline mode, do not perform resource operations that affect the repository, such as deleting or renaming resources.

Setting Up an Integrated Workspace

This section discusses the specific details of setting up projects in your integrated workspace, including placing projects under Integrity Lifecycle Manager version control, and working with Project Sets.

Placing Eclipse Projects Under Integrity Lifecycle Manager Version Control

Projects must be placed under Integrity Lifecycle Manager version control, which requires an Integrity Lifecycle Manager project and Sandbox. Using the Team Support approach, you can work through a wizard that creates the Integrity Lifecycle Manager projects and Sandboxes, and adds existing project files as members.

Note

When you create an Integrity Lifecycle Manager project and Sandbox, the project can reside anywhere on the Integrity Lifecycle Manager server. However, the Sandbox must reside in the same directory as the Eclipse project.

Once an Eclipse project is under Integrity Lifecycle Manager version control, you can perform Integrity Lifecycle Manager operations, such as checking in files and checkpointing projects.

Creating an Integrity Lifecycle Manager Configuration Management Project and Sandbox in your Workspace

To create an Integrity Lifecycle Manager configuration management project and Sandbox in your workspace:

1. In your workspace, select the Eclipse project you want to place under Integrity Lifecycle Manager version control.
2. Right-click and select **Team** ▸ **Share Project**. The **Share Project** panel displays.
3. Select **Integrity** as the repository type.
4. Click **Next**. The Integrity Lifecycle Manager **Sharing** wizard displays.
5. Select one of the following options:
   - **Activate the integration for an existing Integrity Source Sandbox** initiates Integrity Lifecycle Manager and opens the Sandbox you previously created for the project in your workspace. This option is selected by default. If a Sandbox does not exist, this option is disabled.
   - **Create a new Sandbox for an existing Integrity Source project** creates a new Sandbox for an existing Integrity Lifecycle Manager project under Integrity Lifecycle Manager version control. If you select this option, you

are prompted to select an Integrity Lifecycle Manager project and, optionally, a development path.

- **Create a new Integrity Source project and Sandbox** creates a new Integrity Lifecycle Manager project accessible to all users and a Sandbox in your Eclipse workspace. You must be granted the correct project permissions for this functionality to work properly.

6. By default, the option for **Add all files when creating the new Integrity Source project** is enabled. You can disable this option if you want to manually add each file in your project to Integrity Lifecycle Manager version control.

   This option is only available if you selected the **Create a new Integrity Source project and Sandbox** option in step 5.

7. Click **Next**. The options for creating an Integrity Lifecycle Manager project display.
8. For the option you selected in step 5, select one of the available options:

   - **Create a new top-level Integrity Source project** and click **Finish**. The **Specify Project** window opens. In the **File name** field, specify the name of the project you want to create on the Integrity Lifecycle Manager server.
   - **Create a new subproject of an existing Integrity Source project** and click **Next**. Choose the project to create the subproject against. Optionally, you can select one of the project's development paths. To create the subproject, click **Finish**.

   Note

   By default, if you create a chain of nested directories, all subdirectories in the directory are also created as subprojects. If a subdirectory does not include a `.pj` file, `project.pj` is also added to the subproject string.

9. If your project contains files and you are using change packages, you are prompted to specify a change package to associate with the files. Select an existing change package, or click **Create** to create a new change package, then click **OK**.

   If you are not using change packages, proceed to the next step. The **Create Archive** window opens.

10. Modify the **Create Archive** options.

Note

If Integrity Lifecycle Manager finds an existing archive, the **Existing archive detected** window opens.

Integrity Lifecycle Manager automatically creates a Sandbox with the project name in your Eclipse workspace. If your Project contains files, you must submit changes to the Integrity Lifecycle Manager repository. How you do this depends on whether you are using change packages.

11. To submit changes, do one of the following:
    - If you are using change packages, submit the change package associated with the files using the Synchronize view. For more information, see Team Synchronizing on page 52.
    - If you are not using change packages, submit the changes by selecting **Integrity ▸ Submit Changes**. For more information, see Integrity Lifecycle Manager Commands on page 41.

Importing a Project from Integrity Lifecycle Manager

If you want to start using an existing Integrity Lifecycle Manager project, you can import it to create a Sandbox in your workspace.

Note

This is the only method of sharing a project that allows you to rename the project in your workspace. This enables you to import multiple versions of the same project.

To import a project from Integrity Lifecycle Manager:

1. In your workspace, select **File ▸ Import**. The **Select** panel of the **Import** wizard displays.
2. Under **Integrity**, select **Projects from Integrity**.
3. Click **Next**. The **Choose the Integrity Project to import** panel displays.
4. Select the project to import. All Eclipse projects currently under Integrity Lifecycle Manager version control are listed. You can use the **Filter** field to filter the projects by name.

> **Note**
>
> If an Eclipse project is missing from this list, it means that the project description file (`.project` file) has not been put under Integrity version control.

5. Specify the configuration of the project to import. If importing a variant project, specify the **Development Path Name** where the project is located. If importing a build project, specify the **Revision** number or **Label** applied to the project.

> **Note**
>
> If you are importing a variant or build configuration of a project that you already have in your workspace, you must edit the project name. This is because you cannot have two projects with the same name in the same workspace.

6. Specify the location to import the project to. This can either be the default workspace location or a location specified by you.

   In either case, the projects are placed in:
   `selectedLocation/projectName`

   where `projectName` is the name of the project as specified in the **Project Name** field.

> **Note**
>
> The relative paths of imported projects are not maintained.

7. Edit the **Project Name** if required.

> **Note**
> - If there are dependencies between projects, renaming the project results in compile errors.
> - The name in the `.project` file is not changed when you edit the project name in this field.

8. Click **Finish**.

   A directory is created for the project in the specified location. A Sandbox is created within that directory and the Eclipse project files are added.

   Importing an Integrity Lifecycle Manager project results in an identical Eclipse project structure and information within your workspace.

> **Note**
> The Import wizard is only available if the Eclipse integration is connected to an Integrity Lifecycle Manager server that uses the database repository option. If the integration is connected to a server that uses the RCS-style repository option, the **Import** wizard is not available. An error message displays when attempting to the run the wizard.

Unassociating Eclipse Projects from Integrity Lifecycle Manager Version Control

If you place Eclipse projects under Integrity Lifecycle Manager version control on a temporary basis, you can unassociate the projects from Integrity Lifecycle Manager once you are done. This capability is particularly useful as part of a pilot or proof of concept. Unassociating removes the association between the Eclipse project and the Integrity Lifecycle Manager project and Sandbox.

To unassociate an Eclipse project from Integrity Lifecycle Manager version control:

1. In your workspace, select the Eclipse project that you want to unassociate from Integrity Lifecycle Manager version control.
2. Right-click and select **Team** ▸ **Unshare Project**.

You can no longer perform Integrity Lifecycle Manager operations on the project.

Sharing Projects with Project Sets

The Team Support approach also allows for collaboration on projects. In an integrated workspace, you can share groups of Integrity Lifecycle Manager projects with other users. To accomplish this, you create a Project Set or export groups of projects in your workspace. Other users can then import the Project Set File (`.psf`). The projects contained in the Project Set are automatically created for them.

This includes the creation and population of the necessary Sandboxes, based on the Integrity Lifecycle Manager projects referenced in the Project Set. Project sets provide a simple method for team members to share their workspaces.

Key Considerations

- The Eclipse project must be under Integrity Lifecycle Manager version control before you create a Project Set.
- Once the `.psf` is created, do not attempt to edit this file unless errors occur when importing using the file.
- You can export subsandboxes with Team Project Sets. Developers can share workspaces in their entirety. Importing a Team Project Set results in identical Eclipse project structure and information within the new workspace, and creates corresponding common root Sandboxes. For example, assume that a top-level Sandbox and a subsandbox are exported using the Team Project Set feature. They are recreated in the same hierarchy when the Team Project Set is imported.
- If a `.project` file is not under Integrity Lifecycle Manager version control and you attempt to export a Project Set, an error message displays. (The `.psf` file is created, but is mostly empty.) If the `.project` file is under Integrity Lifecycle Manager version control, but has been modified since it was last checked in, exporting the Project Set displays a warning message. However, the `.psf` file still exports. As long as the `.project` file exists, the export completes successfully.
- Assume that an Eclipse project referenced by the Team Project Set does not have its `.project` file under version control. Attempting to import the Project Set results in an error message. A Sandbox is created for the project in the Integrity Lifecycle Manager client, even though the project does not exist in your workspace. To successfully import the Project Set, you must complete the following steps:
    - Drop the Sandbox in the Integrity Lifecycle Manager client
    - If you still want to import the project, add the .project file as a member in the Integrity Lifecycle Manager client

- If you do not want to import the project, remove the appropriate entry from the `.psf` file. Redistribute the updated `.psf` file to all users of the Team Project Set.
- Reimport the Project Set

- If the `.classpath` file is not under Integrity Lifecycle Manager version control, any imported Java projects fail to compile. This is because the build path does not include the JRE System Library. Once the `.classpath` file is in the project, subsequent imports include the new file.

Creating a Project Set in Your Workspace

To create a Project Set in your workspace:

1. In your workspace, select **File ▸ Export**. The **Export** wizard displays.
2. Under **Team**, select **Team Project Set**.
3. Click **Next**. The **Team Project Set** window opens.
4. From the list, select the projects you want to include in the Project Set.
5. In the **File name** field, provide the path and file name for the `.psf`, or click **Browse** to browse to a location.
6. Click **Finish**. The Project Set is created. The `.psf` is ready for distribution to other team members who can import it into their workspaces.

Importing a Project Set

To import a Project Set:

1. In your workspace, select **File ▸ Import**. The **Import** wizard displays.
2. Under **Team**, select **Team Project Set**.
3. Click **Next**. The **Import a Team Project Set** window opens.
4. In the **File name** field, type the path and file name for the `.psf` you want to import, or click **Browse** to browse to the `.psf`.
5. If you do not want the Integrity Lifecycle Manager progress window to display while the Sandboxes are being created, select the **Run the import in the background** option. This option is useful when importing a large `.psf` file that requires many Sandboxes to be created.
6. Click **Finish**. The **Integrity Import Team Project Set Wizard** displays, allowing you to create Sandboxes in a common root location.
7. Click **OK** to create the Sandboxes in a common root location. By default, the workspace is specified as the common root location. You can also specify or browse to a location for the new Sandbox, where it is created automatically in the directory you specified.

Under this root location, directories are created for each project in the project team set. Sandboxes are then created within those directories and the associated Eclipse project files are added.

Note

If you specify an invalid location as the common root, Eclipse deciphers that invalid entry to find a valid location.

If the description file for a project in the Team Project Set is not under Integrity Lifecycle Manager version control, the import fails with the following error message:

```
There is no project description (.project) file in the repository for
<Eclipse project name> referenced by the Team Project
Set file. Please use the Integrity client to drop the Sandbox for
<Eclipse project name>, resolve the inconsistencies
with the Team Project Set, and re-try your import.
```

To resolve inconsistencies with the Team Project Set, do one of the following:

- If you still want to import the project, add its `.project` file as a member in the Integrity Lifecycle Manager client
- If you do not want to import the project, remove the appropriate entry from the `.psf` file

Specifying Directories or Files as Team Ignored

In addition to specifying ignored resource name patterns in your preferences, you can specify a particular directory or file as ignored. This is useful when you need a directory or file to be under Integrity Lifecycle Manager version control in one situation but ignored in another.

For example, a Java resource bundle file (`*.properties`) is typically added to version control in source directories. It is also copied over to build directories, where it does not need to be under version control. You can specify the build directory as ignored.

Note the following:

- The first time you specify a directory or file as team ignored, an `.mksignore` file is created in the root directory of the Eclipse project. Once you submit changes to the project, the created `.mksignore` file is added to Integrity Lifecycle Manager version control.
- Use the **Team** shortcut menu to specify directories or files as team ignored.

- Once under Integrity Lifecycle Manager version control, the `.mksignore` file is editable only through Eclipse or after it is checked out in Integrity Lifecycle Manager.
- Ignored files do not display icon decorators or annotations
- With the exception of the **Remove from Ignore List** command, all Integrity Lifecycle Manager commands are disabled when you select a team ignored resource.
- Glob patterns are unsupported in the `.mksignore` file.

To specify a directory or file as team ignored, you right-click the item you want to ignore and select **Team ▸ Add to Ignore List**. Icon decorators and annotations for the resources disappear.

Note

You can only select one file at a time to specify as team ignored.

To remove a directory or file from the team ignored list, right-click the directory or file and select **Team ▸ Remove from Ignore List**. Icon decorators and annotations for the resources appear.

Working in an Integrated Workspace

This section discusses the specific details of working with projects in your integrated workspace, including:

Understanding Integrity Lifecycle Manager Label Decorations

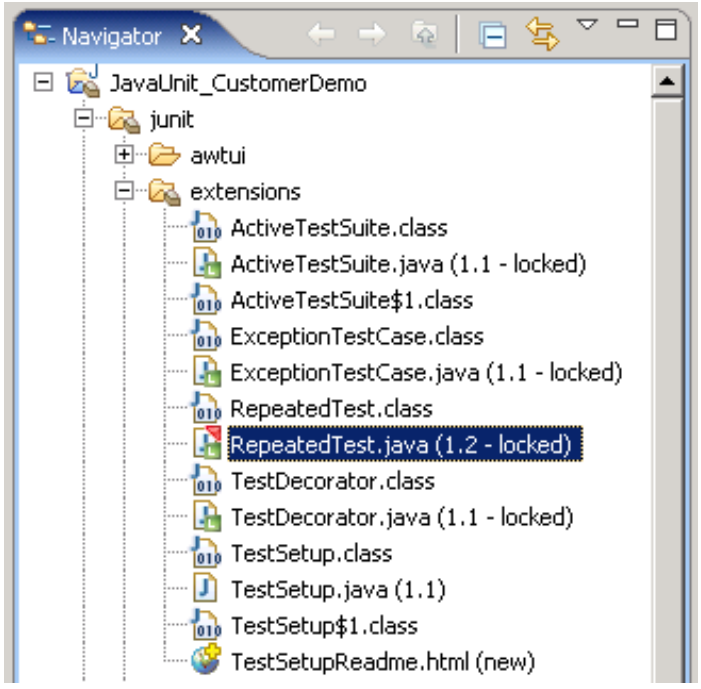
By default, label decorations appear on Eclipse projects and files. Label decorations take the form of icons and annotations. Decorators update dynamically to display the latest Integrity Lifecycle Manager version control status.

Note the following:

- If decorator rollups are enabled and a container (package, subproject, directory) contains modified files, the container displays the working file changed decorator . Modified files include files with changes, moved or

renamed files, and newly added files. In addition, files that are locked, but not modified are included. This is known as decorator rollup. By default, decorator rollup is disabled.

- Some label decorations display in certain views only. For example, when you delete a file, the dropped member decorator displays in the Synchronize view only.
- Label annotations include revision numbers, development path names for variant projects, and lock status. Integrity Lifecycle Manager labels applied to revisions do not display as label annotations. For detailed Integrity Lifecycle Manager information about a file, view the file's properties. For more information, see Displaying Integrity Lifecycle Manager Information on page 36.
- The Eclipse Platform integration provides specific annotations for shared subprojects, including development path names for variant projects and checkpoint revision numbers for build projects. When changes are made to a project or subproject configuration from outside the Eclipse integration, those changes are dynamically displayed within the integration. For information on the available preferences for annotations, see Annotation of Subprojects on page 15.



Enabling and Disabling Label Decorations

To switch between enabling and disabling label decorations:

1. select **Window** ▸ **Preferences**. The **Preferences** window opens
2. Under **General** ▸ **Appearance**, select **Label Decorations**.
3. From the list of available label decorations, select the **Integrity Decorators** checkbox and click **OK**.

Enabling and Disabling Decorator Rollup

To switch between enabling and disabling decorator rollup:

1. In your workspace, select **Window** ▸ **Preferences**. The **Preferences** window opens.
2. Under **General** ▸ **Appearance**, select **Label Decorations**.
3. From the list of available label decorations, select the **Integrity Decorators Rollup** checkbox and click **OK**.

Icon Decorations

Icon decorations are appended to directory and file icons.

Decorator	Function
	The Sandbox decorator indicates that the Eclipse project is under Integrity Lifecycle Manager version control or the directory is a subproject under Integrity Lifecycle Manager version control. A variant or build project also displays the corresponding development path or revision number. A file belonging to a project or subproject under Integrity Lifecycle Manager version control displays the revision number, and, if applicable, status and change package (Synchronize view only). An example follows: `status.java (1.8 - locked [1:45])`. Packages and directories created or converted to subprojects also display the Sandbox decorator.

	The added member decorator and the annotation (`new`) indicate that the file is not one of the following:
	• A linked resource
	• An ignored team
	• A member of the Integrity Lifecycle Manager repository
	  ○ If a file is a deferred add, this icon displays
	  ○ If a file is a pending add, this icon does not display.

Decorator	Function
	When a file is committed to the Integrity Lifecycle Manager repository, the revision number annotation displays and the added member decorator disappears.
	The dropped member decorator indicates that the file is a candidate for dropping as a member from an Integrity Lifecycle Manager project. **Note** Dropping the member deletes the file locally. However, it still exists in the Integrity Lifecycle Manager repository. From the Synchronize view, the **Submit Changes** command drops the file from the Integrity Lifecycle Manager repository.
	The moved/renamed member decorator indicates that the file is a moved or renamed member. The moved or renamed member is not yet committed to the Integrity Lifecycle Manager repository.
	The blue lock decorator indicates that another user has a lock on the member revision. You do not have a lock on the member or working revision.
	The green lock decorator indicates that you have a lock on the member revision. No other users have locks on the same revision.
	The red lock decorator indicates that you have a lock on the working or member revision. Another user has an exclusive lock on the member revision.
	The yellow lock decorator indicates one of the following: • You have an exclusive or non-exclusive lock on the member revision. Another user has a non-exclusive lock on the member revision. • You have an exclusive or non-exclusive lock on the working revision. Your working revision is not the same as the member revision.

Decorator	Function
	The working file changed decorator indicates that the working file has been modified. A directory containing modified working files also displays this decorator.
	The revision out of sync decorator indicates that the working revision does not match the member revision. In the Packages and Navigator views, this decorator indicates the following: • The new revision does not exist on the current development path. More specifically, the member revision is the working revision. However, a new revision is available. • The working revision is an uncommitted update where a user has submitted a change in a transactional change package. This decorator also appears when a change package has been submitted but not reviewed (if change package reviews are enabled). For example, the user's working revision is 1.4, but the member revision is 1.3.

Displaying Integrity Lifecycle Manager Information

To display detailed Integrity Lifecycle Manager information about a resource that is under Integrity Lifecycle Manager version control, right-click the resource (project or file) and select **Properties**. From the **Properties** page, select the **Integrity Source** node. If the resource is under Integrity Lifecycle Manager version control, detailed information displays in the Integrity Lifecycle Manager page. If the object is not under Integrity Lifecycle Manager version control, the following displays: `This resource is not under Integrity Control.`

Integrity Lifecycle Manager information displays for the following resources:

- For projects or Sandboxes, the **Properties** page displays the corresponding project type (normal, variant, build). It also displays the Sandbox path and name, and the server name and port number on which project resides. Additionally, it displays the corresponding project path and name, and the last checkpoint, description, development path, or build revision information.

  For more detailed information about the project or Sandbox, click **Integrity Sandbox Info**. The **Sandbox Information** window opens.
- For members, the **Properties** page displays project and Sandbox information, the path, and name of the member, and the revision number.

  If the member is locked, the locker, lock type, revision number the member was locked at, and change package ID associated with the revision also display.

For more detailed information about the member, click **Integrity Member Info**. The **Member Information** window opens.

Upgrading and Downgrading Locks

To downgrade an exclusive lock to a non-exclusive lock, in the **Properties** page for a member, click **Downgrade Lock**.

To upgrade a non-exclusive lock to an exclusive lock, click **Upgrade Lock**. The new lock type displays.

Working With Active Change Packages

If you have enabled the **Use Change Package** preference, all changes must be submitted using a change package.

The active change package is a change package that has been set as the default change package for Integrity Lifecycle Manager member operations in the workspace. This also allows other developers to identify what you are currently working on. They can see which files you have locked and the associated change package.

For Implementer users, this integration includes the active item feature. The active item is the default item used for an Implementer check out operation

Active Change Package Display

When you open your workspace, the last change package used in the workspace is displayed as the active change package in the Integrity Lifecycle Manager trim.

By default, the active change package label displays the change package ID and change package summary. Text that exceeds the size of the label is truncated. However, you can hover your mouse over the label to display a tooltip. The tooltip displays the change package ID, server, and port that the change package resides on, and change package summary.

If the last change package used in the workspace has been closed, or if the workspace is new, **<no active Change Package>** displays in the Integrity Lifecycle Manager trim.

If you disable the **Use Change Package** preference, all elements in the Integrity Lifecycle Manager trim are disabled until Eclipse is restarted, at which point the change package components are removed. No Integrity Lifecycle Manager commands explicitly set the change package.

The Integrity Lifecycle Manager trim displays in the bottom-right corner of the workbench by default. However, you can drag the trim anywhere in the workbench. The Integrity Lifecycle Manager trim always displays the Integrity

Lifecycle Manager logo and the online/offline mode button. For more information, see Working in Online or Offline Mode on page 22. If the **Use Change Package** preference is enabled, the following are also shown:

- The active change package label
- A list allowing you to select the active change package
- A button to create new change packages

Specifying an Active Change Package

To specify an active change package or select a different one, click the drop-down arrow button next to the change package label in the Integrity Lifecycle Manager trim. The list of available change packages is displayed immediately, without having to wait for the command to complete on the Integrity Lifecycle Manager client.

To create a new change package, click the create a change package button  . The **Create Change Package** window opens.

Managing Items With Integrity Lifecycle Manager Worktray

The Integrity Lifecycle Manager Worktray provides support for Integrity Lifecycle Manager items and Implementer change packages within Eclipse. Although the Integrity Lifecycle Manager integration must be installed for Integrity Lifecycle Manager Worktray to function, you can configure it to use only Implementer features.

The Integrity Lifecycle Manager Worktray consists of views that display item and change package data. As with any other Eclipse element, the location and size of each view can be customized.

Note

Integrity Lifecycle Manager Worktray view data is not dynamically refreshed. However, you can display changes made since the last time the view was opened or changed based on a link to another view. On the view toolbar, click the refresh button  .

Configuring Integrity Lifecycle Manager Worktray Preferences

Integration-specific preferences allow you to select settings for the Integrity Lifecycle Manager Worktray views. Click **Restore Defaults** at any time to clear the changes you made.

To configure Integrity Lifecycle Manager Worktray preferences

1. In your workspace, select **Window** ▸ **Preferences**. The **Preferences** window opens.
2. Under **Team**, select **Integrity Worktray**. The **Integrity Worktray** preferences display.
3. Make the necessary changes to the following preferences for **Integrity Applications**:
   - **Enable Integrity Source actions only** enables Integrity Lifecycle Manager configuration management commands in the views. This option is enabled by default.
   - **Enable Integrity Implementer actions only** enables Implementer commands in the views.
   - **Enable Integrity Source and Implementer actions** enables all commands in the views.

> **Note**
>
> If the Integrity Lifecycle Manager Worktray view is open when making changes to **Integrity Worktray** preferences, you must click  to refresh the view and show the changes.

Integrity Lifecycle Manager Worktray View

To display the Integrity Lifecycle Manager Worktray view

1. Select **Window** ▸ **Show View** ▸ **Other**. The **Show View** window opens.
2. Open the **Integrity Solution** directory. Then, select **Integrity Worktray**. The Integrity Lifecycle Manager Worktray view displays.

The Integrity Lifecycle Manager Worktray view displays Integrity Lifecycle Manager items based on the selected query. By default, items are displayed based on your Quick Query criteria. Query criteria must be defined and made visible from Integrity Lifecycle Manager before that query can be used from the Integrity Lifecycle Manager Worktray.

> **Note**
>
> To make new queries available in the list, you must refresh the view.

The Integrity Lifecycle Manager Worktray view displays the column set associated with the selected Integrity Lifecycle Manager query, rather than only displaying a defined, default column set. Therefore, the column set selection is no longer available under **Integrity Worktray Preferences**.

To change the column set displayed in the Integrity Lifecycle Manager Worktray view, you modify the column set referenced in the underlying query. For more information on working with queries and column sets, see the *Implementer Installation and Administration Guide*.

Note

Your **Integrity Worktray** preferences settings determine which commands or toolbars display.

The following operations are available from the ▼ menu in the Integrity Lifecycle Manager Worktray view:

Command	Operation
Create Item	Creates an Integrity Lifecycle Manager item. **Tip** The **Create Item** operation is also available from the shortcut menu when you right-click in the Integrity Lifecycle Manager Worktray view.
Edit Item	Edits the selected item.
View Item	For the selected item, displays its details in the Integrity Lifecycle Manager client GUI.
Create Related Item	Creates an item related to the selected item.
Refresh	Refreshes the Integrity Lifecycle Manager Worktray view.

Integrity Lifecycle Manager Implementer Change Package View

1. Select **Window** ▸ **Show View** ▸ **Other**. The **Show View** window opens.
2. Open the **Integrity Solution** directory. Then, select **Integrity Implementer Change Package**. The Integrity Lifecycle Manager Implementer Change Package view displays.

The Integrity Lifecycle Manager Implementer Change Package view displays information for Implementer change packages. The view only displays the information for a single change package at a time. However, you can change the displayed change package by using the arrows on the toolbar.

The link button  links the Integrity Lifecycle Manager Implementer Change Package view with the Integrity Lifecycle Manager Worktray view. When an item is selected, the corresponding change package displays. Clicking the button again remove the linking. For detailed information on Implementer change packages, see the *Implementer Installation and Administration Guide*.

The following operations are available from the ▼ menu in the Integrity Lifecycle Manager Implementer Change Package view.

Command	Operation
Previous Change Package	Displays the previous change package by order of change package ID if more than one change package is associated with the selected item.
Next Change Package	Displays the next change package by order of change package ID If more than one change package is associated with the selected items.
Open in Integrity	Displays the change package details in the Integrity Lifecycle Manager client GUI.
Refresh	Refreshes the Integrity Lifecycle Manager Implementer Change Package view.

Integrity Lifecycle Manager Commands

To provide a more seamless Integrity Lifecycle Manager experience within Eclipse, basic Integrity Lifecycle Manager commands occur implicitly in the Integrity Lifecycle Manager repository when you perform the equivalent Eclipse commands. Basic Integrity Lifecycle Manager commands include adding and dropping members as well as checking them in and out. File status is displayed immediately to other Eclipse users working in the project. For more information on performing operations, refer to the following sections:

- Adding Members to an Integrity Lifecycle Manager Project on page 42
- Dropping Members from an Integrity Lifecycle Manager Project on page 43
- Checking Out Members on page 43
- Checking In Members on page 44

Advanced Integrity Lifecycle Manager version control functionality includes submitting changes or checkpointing a project. To access advanced Integrity Lifecycle Manager commands, do one of the following:

- Select an Eclipse project or one or more files and then select the **Integrity** menu.
- Right-click the selected item and then select the **Team** menu.

Note that not all commands are available on both menus. The commands that are available depend on your selection. For example, the **View Member Differences** command is available only when you select a file under Integrity Lifecycle Manager version control. The Eclipse status bar indicates when an Integrity Lifecycle Manager command is complete. For a list of Integrity Lifecycle Manager commands, see Advanced Integrity Lifecycle Manager Commands on page 44.

Adding Members to an Integrity Lifecycle Manager Project

You can add members to an Integrity Lifecycle Manager configuration management project in your workspace.

To add members:

1. In your workspace, create one or more new files.

   Note

   The integration also adds members if you refresh Eclipse and it discovers one or more new files that are unknown to the Integrity Lifecycle Manager repository. For example, if you copy a file to your Sandbox controlled in Eclipse, and then refresh Eclipse, the file is added to the Integrity Lifecycle Manager project.

   The files are added as members and decorators display immediately for each member. If change packages are enabled and an active change package exists, the new files are associated with the active change package. The files can then be committed to the Integrity Lifecycle Manager repository.

   Note

   If you are using change packages, it is possible that you did not set an active change package when the file was created. However, you can use the **Move to Change Package** command to associate the file with a change package.

2. Commit the changes to the Integrity Lifecycle Manager repository by doing one of the following:

- If you are using change packages, submit the change package associated with the new files. For more information, see Team Synchronizing on page 52.
- If you are not using change packages, select the new files, and then select **Integrity ▸ Submit Changes**. For more information, see Advanced Integrity Lifecycle Manager Commands on page 44.

Dropping Members from an Integrity Lifecycle Manager Project

To drop members from an Integrity Lifecycle Manager configuration management project in your workspace

1. In your workspace, select one or more files to drop from the Integrity Lifecycle Manager project.
2. Select **Edit ▸ Delete**.
3. Commit the changes to the Integrity Lifecycle Manager repository by doing one of the following:
   - If you are using change packages, submit the change package associated with the deleted files. For more information, see Team Synchronizing on page 52. If an active change package was not set when the file was deleted, you can use the **Move to Change Package** command to associate the file with a change package.
   - If you are not using change packages, select the files to delete in the Synchronize view, and then select **Integrity ▸ Submit Changes**. For more information, see Advanced Integrity Lifecycle Manager Commands on page 44.

Checking Out Members

You can check out members in your workspace. In your workspace, open a file and begin editing. If you have the **Lock file** option selected in the preferences, the file is locked and the appropriate lock decorator displays.

Note

If the **Lock file** option is not selected in the Integrity Lifecycle Manager preferences, the working file is made writable when you edit the file. You must either manually lock the modified working file later or associate it with a change package. For more information, see Setting Preferences on page 15.

Checking In Members

You can check in members in your workspace. To commit changes to the Integrity Lifecycle Manager repository, do one of the following:

- If you are using change packages, submit the change package associated with the modified files. For more information, see Team Synchronizing on page 52.
- If you are not using change packages, select the modified files, and then select **Integrity** ▸ **Submit Changes**. For more information, see Advanced Integrity Lifecycle Manager Commands on page 44.

Advanced Integrity Lifecycle Manager Commands

The following advanced Integrity Lifecycle Manager commands are available from the **Integrity** or **Team** shortcut menu. Some of these commands are available on both of these shortcut menus.

Integrity Lifecycle Manager Command	Function
Move to Change Package	Moves the selected files to an existing or new change package. The change package containing the associated changes can then be submitted to the Integrity Lifecycle Manager repository. From the list, select a change package or click **Create Change Package**. The status bar indicates when the command is complete. **Note** • You cannot move a rename or move change package entry out of a change package. You can only move these types of entries to another change package. • You cannot move a rename or move operation to a change package if it is not already associated with a change package.
Submit Changes	Equivalent to the Integrity Lifecycle Manager **Submit** command. Submits uncommitted changes on individual files. The status bar indicates when the command is complete.

Integrity Lifecycle Manager Command	Function
	If the selection is from the Packages view, a submit does not include dropped files. However, a submit includes dropped files displayed in the Synchronize view. **Note** The **Submit Changes** command is not available if you have selected the **Use active Change Package** option. For more information, see Working with Active Change Packages on page 37.
Resynchronize	Equivalent to the Integrity Lifecycle Manager **Resynchronize** command. Gets the latest version of the selected file and puts it in your working directory. An overwrite window can open. The status bar indicates when the command is complete.
Resynchronize by Change Package	Equivalent to the Integrity Lifecycle Manager **Resynchronize by Change Package** command. Processes the change packages associated with the member you are resynchronizing, and brings the changes from the project to your Sandbox. Depending on preferences set for the **Resynchronize** command, an overwrite window can open.
Revert	Equivalent to the Integrity Lifecycle Manager **Revert** command. Replaces the working file with the revision that was checked out, as it appeared before modification. It also unlocks the file and removes it from the associated change package. **Note** You can revert a file in a change package that is not the active change package. Reverting the file removes it from the change package. An overwrite window can open. The status bar indicates when the command is complete.
Lock Member	Equivalent to the Integrity Lifecycle Manager **Lock**

Integrity Lifecycle Manager Command	Function
	command. Right-click the selected file in the Package Explorer or Project Explorer view, and select **Lock**.
View Member Differences	Equivalent to the Integrity Lifecycle Manager **Differences** command. Compares the selected working file with the member revision. The **Visual Difference** window opens, displaying the two files.
View Annotated Revision	Equivalent to the Integrity Lifecycle Manager **View Annotated** command. Displays the annotated revision history of the selected file. The Annotated Revision view displays.
View Member History	Equivalent to the Integrity Lifecycle Manager **View Member History** command. Displays the revision history of the selected file. The Member History view displays.
Create Change Package	Equivalent to the Integrity Lifecycle Manager **Create Change Package** command. Creates a change package. The **Create Change Package** window opens.
View Active Change Package	Equivalent to the Integrity Lifecycle Manager **View Change Package** command. Displays the active change package associated with the selected file. If there is no active change package, this command is disabled. The Change Package view displays. For more information on active change packages, see Working With Active Change Packages on page 37.
Resynchronize Change Packages	Equivalent to the Integrity Lifecycle Manager **Resynchronize Change Package** command. Previews the changes listed in change packages in the context of a Sandbox before propagating them to the project. The **Resynchronize Change Packages** window opens.

Integrity Lifecycle Manager Command	Function
	After you resynchronize the change package containing the changes, the changes appear as operations not yet committed to the Integrity Lifecycle Manager repository. Once you are satisfied with the changes and want to commit them to the Integrity Lifecycle Manager repository, submit the change package, and then resynchronize the members. **Note** After resynchronizing by change package, the changes can appear as incoming changes in the Synchronize view. Do not resynchronize the changes; otherwise, they are lost. Update the revisions by submitting the changes. For more information on submitting changes, see Team Synchronizing on page 52.
Submit Active Change Package	The **Submit Active Change Package** command allows you to submit an active change package directly from the **Integrity** menu within Eclipse. The **Submit Active Change Package** command operates in the same way as the **Submit Change Package** command on the Integrity Lifecycle Manager client, and follows most client preferences as set for that command.
Create Subproject	Equivalent to the Integrity Lifecycle Manager **Create Subproject** command. Creates a subproject in the selected directory and adds it to the Integrity Lifecycle Manager project. After you type a name for the subproject, the **Create Subproject** window opens. **Note** If a conflict or error occurs when attempting to name the subproject, you are prompted to type a new name for the subproject.
Convert to Subproject	Converts an empty directory or a directory containing files that are not under Integrity Lifecycle Manager version control to a subproject. If the directory contains

Integrity Lifecycle Manager Command	Function
	files, they must be added to Integrity Lifecycle Manager version control after the subproject is created. This command is useful for defining a directory structure for a build project.
Drop Subproject	Equivalent to the Integrity Lifecycle Manager **Drop Subproject** command. Drops the selected subprojects from the Integrity Lifecycle Manager project, deleting all files and directories under the subprojects. The **Drop Subproject** window opens.
View Sandbox	Equivalent to the Integrity Lifecycle Manager **View Sandbox** command. Displays a Sandbox view.
Merge child Development Path	Equivalent to the Integrity Lifecycle Manager **Merge Child Development Path** command. Merges a development path into its parent development path. The parent development path can be a mainline, or it can be a development path itself. The merge destination must be the parent of the child being merged into it. The command creates a propagation change package containing the changes necessary to perform the development path merge. For more information, see the *Integrity Lifecycle Manager Help Center*.
View Project Differences	Equivalent to the Integrity Lifecycle Manager **Project Differences** command. Displays the differences between project checkpoints. The Project Differences view displays.
Checkpoint	Equivalent to the Integrity Lifecycle Manager **Checkpoint** command. Checkpoints one or more Eclipse projects under Integrity Lifecycle Manager version control. If the project is a subproject, the checkpoint recursively checkpoints everything under it; parent projects in the hierarchy are not checkpointed.

Integrity Lifecycle Manager Command	Function
	A window opens, prompting you to select one or more Eclipse projects, and to, optionally, add a label and description. **Note** Labels are applied to the Eclipse project, but not project members. A confirmation window opens. It indicates which projects were successfully or unsuccessfully checkpointed. A maximum of 15 projects are shown.
Work Offline/Online	Switches between the offline and online modes. For more information, see Working in Online or Offline Mode on page 22.
Open Integrity Client	Equivalent to launching the Integrity Lifecycle Manager client GUI. **Tip** You can also launch the Integrity Lifecycle Manager client by clicking the PTC button in the Integrity Lifecycle Manager trim.

Note the following:

- You can make the **Integrity** menu always available from the perspective you have open by selecting **Window** ▸ **Customize Perspective**. In the **Commands** panel, enable **Integrity Source Menu**.
- You can use Eclipse's key binding functionality to assign key sequences to commands in the **Integrity** and **Team** menus. Note that Integrity Lifecycle Manager commands are only visible when **Include unbound commands** is enabled under **General** ▸ **Keys** in the **Preferences** window. For more information, refer to the Eclipse documentation.
- Although Eclipse supports linked resources, known as out-of-tree members in Integrity Lifecycle Manager, they cannot be placed under Integrity Lifecycle Manager version control. As a result, Integrity Lifecycle Manager commands are disabled. Additionally, decorators do not appear when you select linked resources.
- If you revert or resynchronize selected files, only those files are reverted or resynchronized. If you revert or resynchronize selected containers (Eclipse projects, directories, packages), the integration examines the first Sandbox or subsandbox of each container. It then reverts or resynchronizes the entire

Sandbox or subsandbox. If you select an Eclipse container that maps to a directory in Integrity Lifecycle Manager, the Sandbox containing the directory is resynchronized or reverted.

Refactoring

The Eclipse integration allows you to refactor your source code and preview the changes before you commit them to the Integrity Lifecycle Manager repository. Refactoring activities include adds, drops, moves, and renames. They are handled as member operations not yet committed to the Integrity Lifecycle Manager repository. These activities are recorded in change packages that you can either submit or discard.

Submitting a change package commits your changes to the Integrity Lifecycle Manager repository. This is done by clicking one of the Submit buttons in the Synchronize view. For more information, see Team Synchronizing on page 52. You can also select **Submit Changes** and **Submit Change Package** commands from the **Integrity** menu. For more information, see Advanced Integrity Lifecycle Manager Commands on page 44.

To discard your changes, discard the change package or change package entry using the **Integrity ▸ Revert** command, or undo the change through Eclipse. For example, if you want to undo a move operation, move the file back to its original location.

Note the following:

- Where possible, use Eclipse's **Undo** command to undo changes. If it is not possible to use the **Undo** command, use the Integrity Lifecycle Manager **Revert** command.
- Assume that you create an Integrity Lifecycle Manager configuration management project in the Integrity Lifecycle Manager client and then import it into Eclipse. Renaming a package in the project modifies the subproject organization in Integrity Lifecycle Manager. More specifically, the directory is created on the file system (in the Sandbox directory). You then use member commands to move files to make the changes happen in Integrity Lifecycle Manager.
- Renaming an Eclipse project under Integrity Lifecycle Manager version control is not supported.
- When you rename a package that corresponds to a subproject in Eclipse, the operation is recorded as move operations for the members contained in that package. When these changes are submitted, the old subproject folder is not removed until the corresponding subproject is dropped. To drop a subproject, the **Drop Subproject** operation must be submitted in a separate change package.

If you rename a folder in the context of the Eclipse workspace, the old folder is removed as a result of submitting the changes to the files in that folder.

- If you delete a project from your workspace, the associated Sandbox is automatically dropped if the preference for **Drop Integrity Sandbox when Eclipse project is deleted** is enabled. For more information, see Setting Preferences on page 15.
- When you use the Eclipse **Edit ▸ Delete** command (or press the DELETE key) on a directory, the integration drops files in the directory. To remove the entire subproject, use the **Integrity ▸ Drop Subproject** command.

Comparing Revisions

The integration with Integrity Lifecycle Manager supports functionality for comparing and merging revisions. When conflicts are detected during a synchronization, you are automatically prompted to resolve those conflicts by differencing the files. The integration with Integrity Lifecycle Manager supports both two and three-way differencing. For three-way differencing, you are prompted to perform a merge on the branch and then a second merge against the selected project root.

After synchronizing a project, you can navigate all revisions that show differences using the go to next difference button and go to previous difference button in the Synchronize view. All differences in the file are visited before opening the next file in the view.

Differencing is supported for both plain text, binary files, and models, provided the required tools are available on your system. The differencing tool presented is based on the tools you use in your IDE environment and on the file type, for example, `UML`, `JAVA`, or `TEXT`.

To compare differences in a revision

In the Synchronize view, right-click the target file and select **Open in Compare Editor**. The local and remote revisions of the target file display in the Compare window for the available differencing tool. Perform the required merge operations to resolve the differences. Save your changes.

Tip

To compare differences between the working file and member revision, select the file in the Navigator or Packages view, then select **Integrity ▸ Member Differences**. If differences exist, the **Visual Difference** window opens.

History View

The History view displays the Integrity Lifecycle Manager revision history of a file.

Tip

Tip You can also view a file's revision history by selecting the file, then selecting **Integrity ▸ View Member History**. The Member History view displays.

To display the History view, select **Window ▸ Show View ▸ Other**. The **Show View** window opens. Open the **Team** directory, and then select **History**.

The History view displays the file name, revision number, modification date, author, and change package. It also displays the name of any user with a locked revision and corresponding change package and state information. A * beside a revision number indicates the member revision, and bold indicates the working revision.

To refresh the History view, click the refresh button  .

The link button  links the History view with the Editor view. This means that when a file is opened in the Editor view, the corresponding Integrity Lifecycle Manager revision history displays. Clicking the button again removes the linking and display the local history.

To pin the History view, click the pin current synchronization button  .

Team Synchronizing

Eclipse synchronization support provides an interface for managing dynamic collections of information. Synchronizing is the operation that displays the changes between resources in different locations. You can modify the synchronization state by performing an action.

The Team Synchronizing perspective allows you to see all project resources that require synchronization in your workspace. Essentially, the Team Synchronizing perspective summarizes all changes present in the selected resources. The integration supports the Eclipse Team Synchronizing perspective through the Integrity Lifecycle Manager synchronizer.

Synchronizing an Integrity Lifecycle Manager Project

Synchronizing provides dynamic updates of changes to shared resources. The Integrity Lifecycle Manager synchronizer provides a view that allows you to see all project resources that show comparative changes from your local workspace to the remote Integrity Lifecycle Manager configuration management project on the Integrity Lifecycle Manager server.

The Integrity Lifecycle Manager synchronizer provides filters to control whether you view incoming changes, outgoing changes, conflicts, or all changes. Incoming changes come from the Integrity Lifecycle Manager project. If accepted, they update your local revision to the latest version currently committed in the project. Outgoing changes come from your local workspace. If committed, they change the project member to match the revision present in your Sandbox.

Integrity Lifecycle Manager provides several options for filtering the Synchronize view. You can filter projects by specific resources or change packages containing uncommitted member operations. Once the Synchronize view is populated, you can further filter changes in the view according to the type of change, whether incoming, outgoing, or a conflict.

Whichever mode (or filter) you select, the Integrity Lifecycle Manager synchronizer shows you any conflicts for your locally modified revision when a more recent version is available. In this situation, you can do one of three things:

- Update the revision from the Integrity Lifecycle Manager project
- Commit your version of the resource to the Integrity Lifecycle Manager project
- Merge your work with the changes into the Integrity Lifecycle Manager project

Generally you merge your changes to avoid losing modifications. For more information on resolving conflicts, see Comparing Revisions on page 51.

The collapse all button  allows you to instantly collapse the project directory structure. Expanding a project tree expands according to the selected filter, if any. For other native Eclipse options, see the Eclipse product documentation.

The following commands are available from the menu bar in the Synchronize view.

Button Icons	Function
	The synchronize button displays the **Synchronize** wizard, where you can see the state of your workspace as it relates to the Integrity Lifecycle Manager project.
	The pin current synchronization button prevents dynamic updates of the view, allowing you to hold your workspace

Button Icons	Function
	synchronization. **Note** You cannot pin a Synchronize view that is filtered on the active change package. However, you can on an open change package or a set of resources.
	The go to next difference button navigates through the project hierarchy to the next revision containing differences. Automatically launches the available differencing tool.
	The go to previous difference button navigates through the project hierarchy to the previous revision containing differences. Automatically launches the available differencing tool.
	The collapse all button closes all expanded entries in the view.
	The incoming mode button filters the view to show only those working files with associated incoming changes or incoming add operations.
	The outgoing mode button filters the view to show only those working files with associated outgoing changes or outgoing add operations.
	The incoming/outgoing mode button filters the view to show all working files with incoming and outgoing changes.
	The conflicts mode button filters the view to show only those working files with incoming and outgoing changes containing conflicts.
	The submit this change package button submits the selected or active change package containing your changes to the Integrity Lifecycle Manager repository. This button is only visible when you filter the Synchronize view by change packages.
	The submit all outgoing changes button submits outgoing changes to the Integrity Lifecycle Manager repository. This button is only visible when you are not using active change package tracking and you filter the Synchronize view by resources.
	The resync all incoming changes button resynchronizes your workspace with all incoming changes. This button is only visible when you filter the Synchronize view by resources.

Team Synchronizing Icons

The following team synchronizing decorators are appended to file icons.

Decorator	Function
	The incoming changes decorator indicates that there are incoming changes available for the member.
	The incoming add decorator indicates that the member was added to the Eclipse project under Integrity Lifecycle Manager version control, but does not exist in your workspace.
	The incoming drop decorator indicates that the member was dropped from the Eclipse project under Integrity Lifecycle Manager version control. However, the file still exists in your workspace.
	The outgoing changes decorator indicates that there are outgoing changes available for the member.
	The outgoing add decorator indicates that the member was added to your workspace. However, it is not currently a member of the Eclipse project under Integrity Lifecycle Manager version control.
	The outgoing drop decorator indicates that the member was deleted from your workspace, but has not been dropped from the Eclipse project under Integrity Lifecycle Manager version control.
	The conflict changes decorator indicates that there are conflicting changes between incoming and outgoing changes for a member. This can include changes to an earlier revision of the member.
	The conflicting drop decorator indicates that you modified a member that was dropped from the Eclipse project.

Performing a Team Synchronization

To perform a Team synchronization:

1. From your Eclipse workspace, open the Team Synchronizing perspective by selecting **Window** ▸ **Open Perspective** ▸ **Other** and then selecting **Team Synchronizing** from the list.

> **Note**
>
> You can also open the Synchronize view by selecting **Window** ▸ **Show View** ▸ **Other** ▸ **Team** ▸ **Synchronize**. When you finish running the wizard and Eclipse prompts you to switch perspectives, click **No**.

2. Click **OK**. The Synchronize view displays.
3. To run a synchronization, click the synchronize button  . The **Synchronize** wizard displays.

> **Note**
>
> The `F5` function always refreshes the view, and operations within Eclipse trigger an update to the Team Synchronizing perspective.

4. From the list, select **Integrity**. Then, click **Next**.
5. From the list, select an option for populating the Synchronize view:
   - **Filter by the active Change Package** populates the Synchronize view with project resources that have uncommitted changes in the active change package.
   - **Filter by one of my open Change Packages** populates the Synchronize view with project resources that have uncommitted changes in an open change package.

     After clicking **Next**, you select an open change package from the list.
   - **Filter by a selection of resources** populates the Synchronize view with specific project resources that have changed. You can optionally select **Show only changes not associated with a change package**.
6. Click **Next**. Then, select specific resources to synchronize, or select a scope to automatically select a group of resources:

- **Workspace** synchronizes all available resources in the workspace.
- **Selected Resources** synchronizes the resources that you select in the **Available resources to Synchronize** list.
- **Working Set** synchronizes the resources in the working sets that you select from the **Select Working Sets** window. To select a working set, click **Choose**.

Note

The Synchronize view filters on the entire workspace if you select the following options:

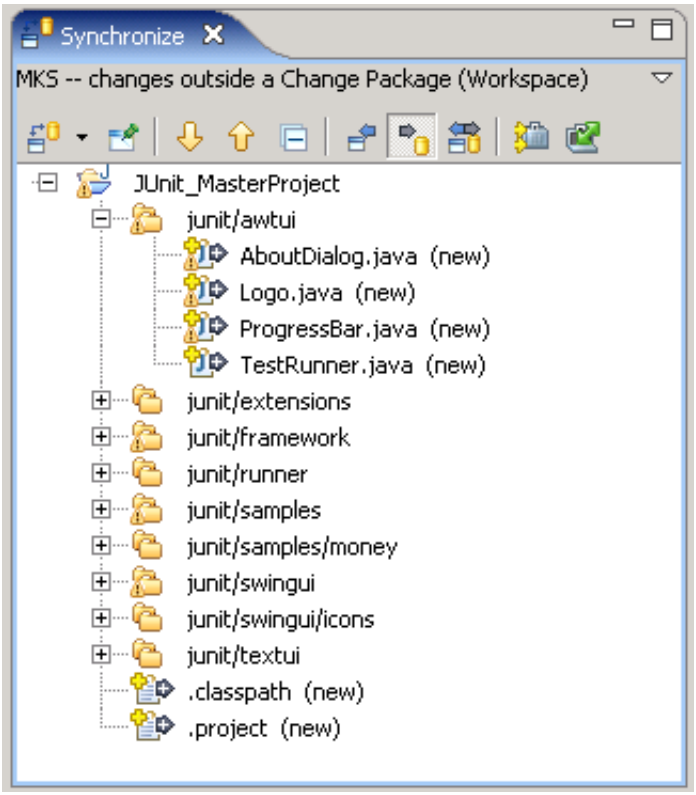
- **Selected Resources** with no projects selected in the **Available resources to Synchronize** list.
- **Working Set** with one of the following options selected in the **Select Working Sets** window: **Window Working Set**, **No Working Set**, or **Selected Working Set** with no projects selected in the list.

7. To populate the Synchronize view, click **Finish**. If there are changes, they display in the Synchronize view. If there are no changes, the Synchronize view is empty.

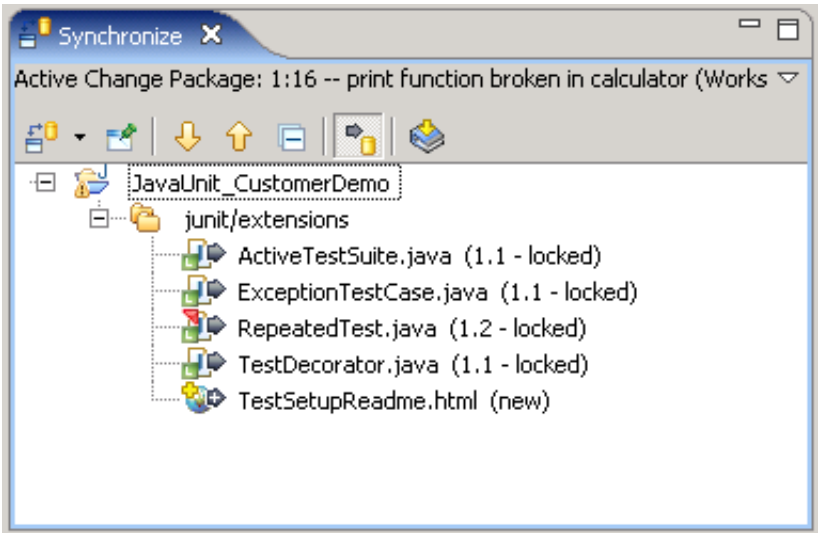
Note

Ignored or derived resources do not display in the Synchronize view.

If you filtered the Synchronize view by specific resources, the Synchronize view appears similar to the following:



If you filtered the Synchronize view by change package, the Synchronize view appears similar to the following:



8. To review the changes in a member in the Integrity Lifecycle Manager view, right-click the member and select **View Member Differences**. The **Visual**

Differences window opens, displaying the differences between the working file and the member revision. To launch the **Eclipse Compare** editor, select the **Open in Compare Editor** command or double-click the Synchronize view entry.

9. If you filtered the Synchronize view by specific resources, do one of the following:

   - To submit all outgoing changes when not using a change package, click the submit all outgoing changes button  .

     To submit all outgoing changes when using a change package, click the submit this change package button  .
   - To resynchronize all incoming changes, click the resync all incoming changes button  .
   - To resynchronize a working file with its corresponding change package, select the incoming members. Then right-click and select **Resynchronize by Change Package**.

   If you filtered the Synchronize view by change package, submit the change package by clicking the submit this change package button  .

5

Best Practices

This section describes best practices for using the Eclipse integration. It also points out efficiencies that can be gained by using the integration in a certain way. Additionally, this section identifies any risks, constraints, or other limits within the integration or within a particular implementation of the integration.

- If you are sharing an Eclipse project containing a large number of subdirectories and files, do one of the following:
  - Place the project under version control in the Integrity Lifecycle Manager client, and then share the project in Eclipse.
  - Disable the **Add all files when creating the new Integrity Source project** option when placing an Eclipse project under Integrity Lifecycle Manager version control. After the Integrity Lifecycle Manager project and Sandbox are created, add the project files in small batches using a separate change package for each batch.
- For ease of distribution, control, and versioning, create a specific Integrity Lifecycle Manager project and Sandbox for `.psf` files. Users can then access the required `.psf` file from a central location.
- Mark `.class` files as team ignored. For more information, see Setting Ignored Resources Preferences on page 18.
- To optimize the resynchronization of project resources and submission of changes, the integration is designed to use change packages and the Synchronize view. When you begin using the integration, open the Synchronize view and filter it by the resources you want to work on.

- A change package is a container for distinct tasks, not a generic container for all tasks performed during a project. A change package containing too many entries can affect the integration's performance. Before you perform a task that requires a change package, create a change package and enable it as the active change package.
- Resynchronize project resources frequently and incrementally. Resynchronizing a large number of resources in a single operation can take a long time.
- Allocate memory to Eclipse and the Integrity Lifecycle Manager client according to the size of your Eclipse projects. Use the following as a guideline to help you determine memory allocations:
  - 50 projects with 100,000 members
    - Set the Integrity Lifecycle Manager client maximum heap size to 512 MB
    - Set the Integrity Lifecycle Manager client cache size to 100 MB
    - Allocate at least 512 MB of memory to Eclipse
  - Set the 100 projects with 200,000 members
    - Integrity Lifecycle Manager client maximum heap size to at least 1 GB
    - Set the Integrity Lifecycle Manager client cache size to 200 MB
    - Allocate at least 1 GB of memory to Eclipse
  - 200 projects with 400,000 members
    - Set the Integrity Lifecycle Manager client maximum heap size to the highest possible number for your platform (most commonly this is 1.5 GB)
    - Set the Integrity Lifecycle Manager client cache size to 400 MB
    - Allocate at least 2 GB of memory to Eclipse. Use the 64-bit version of Eclipse if you need more than 2 GB

Note

To set the Integrity Lifecycle Manager client cache size, edit the `si.Cache.default.size` setting in *<installdir>*`/IntegrityClientSite.rc` where *<installdir>* is the path to the directory where you installed the Integrity Lifecycle Manager client.

- To improve performance with large projects, use working sets to display only necessary resources in navigation views. For more information, see the Eclipse documentation.
- The Eclipse integration is tightly coupled with the Integrity Lifecycle Manager client. When using Eclipse, focus issues can occur if you open the Integrity Lifecycle Manager client GUI. If this occurs, close the Integrity Lifecycle Manager client interface but do not shut down the Integrity Lifecycle Manager client process. If Eclipse shuts down, shut down the Integrity Lifecycle Manager client, and then restart the client.
- Working in offline mode for long periods of time is not recommended. When you switch to online mode after an extended period of time, performance issues can arise from rebuilding the cache.
- To ensure the expected prompts when obtaining locks and change packages, make sure that change package options are consistently configured between the Integrity Lifecycle Manager server and Eclipse. If change packages are mandatory on the Integrity Lifecycle Manager server, set the integration option for **Use active Change Package** and enable prompting for the active change package. If change package policies are not consistent, changes that you make to the files in your workspace are not necessarily recorded as expected in Integrity Lifecycle Manager. For more information on the available preferences, see Setting Integrity Lifecycle Manager Preferences on page 15.

Limitations

- After switching from offline to online mode, you cannot submit changes in the Synchronize view while the Integrity Lifecycle Manager status cache refreshes. Before you submit changes, check the status of the Eclipse progress bar and ensure that there is no Integrity Lifecycle Manager command activity.
- When a subproject is dropped in another user's Sandbox, resynchronizing the dropped subproject's members in the Synchronize view displays an error message. To avoid this, resynchronize the dropped subproject in the Package Explorer.
- When filtered by the active change package, the Synchronize view updates if do one of the following:
    - Change the active change package from **<none>** to a change package.
    - Change the active change package from one change package to another change package.

  The view does not update when you change the active change package from a change package to **<none>**. To display changes not associated with a change package in the Synchronize view, run the **Synchronize** wizard, filtering by resources.
- Existing locks on dropped members are not removed. To resolve this, open the Locks view in the Integrity Lifecycle Manager client GUI and unlock the members.
- Using the **View Sandbox** command on a dropped subproject incorrectly displays an error message.
- The initial attempt to rename a subproject after adding a member to the subproject displays an error message: `the Resource is out of sync with the filesystem`. Refreshing the subproject and performing the rename command successfully completes the operation.
- Attempts to revert deferred added members from the Synchronize view do not successfully remove them from the Synchronize or Packages views unless you also delete the file. Reverting the files from the Integrity Lifecycle Manager client successfully removes them from both views.
- To avoid potential focus problems when using the Integrity Lifecycle Manager **Sharing** wizard, close the Integrity Lifecycle Manager client if it is open.
- To hide files, such as the Integrity Lifecycle Manager project registry file (`project.pj`), certain resource filters are enabled by default in Eclipse. By default, these filters apply to the Packages and Navigator views.
- There are a number of conditions that can cause Eclipse to identify new files in a project. This results in deferred add operations for non-member files that

have not been modified by the user within the Eclipse IDE. For example, assume that automatic refresh is enabled in Eclipse, build scripts generate files. Integrity Lifecycle Manager identifies the generated files as new files to be added to the project.

If you routinely encounter this situation and do not want Integrity Lifecycle Manager to add such generated files to your projects, you can add the file type to the Eclipse **Ignored Resources** list. First, select **Window ▸ Preferences**. Under **Team ▸ Ignore Resources**, add the pattern for the file type that you want the integration to ignore.

Getting Help

PTC Technical Support is focused on delivering the right solutions to issues as they arise. Online support provides easy access to e-mail, Web request services, automatic product notifications, and the PTC Integrity eSupport portal. This secure database provides helpful resources such as product documentation, knowledge base articles, product downloads, user forums, presentations, and more. For online support, browse to http://www.ptc.com/support/integrity.htm.

PTC global technical support professionals comprise a tightly knit team of problem solvers. They share critical information to help you resolve issues in the shortest possible time with optimal results. Support representatives can provide you with various product-related tips and innovative solutions to your unique requirements.