Basic tutorial

OTI Employee

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Basic tutorial

This tutorial provides a step by step walk−through of the Workbench.

The Workbench

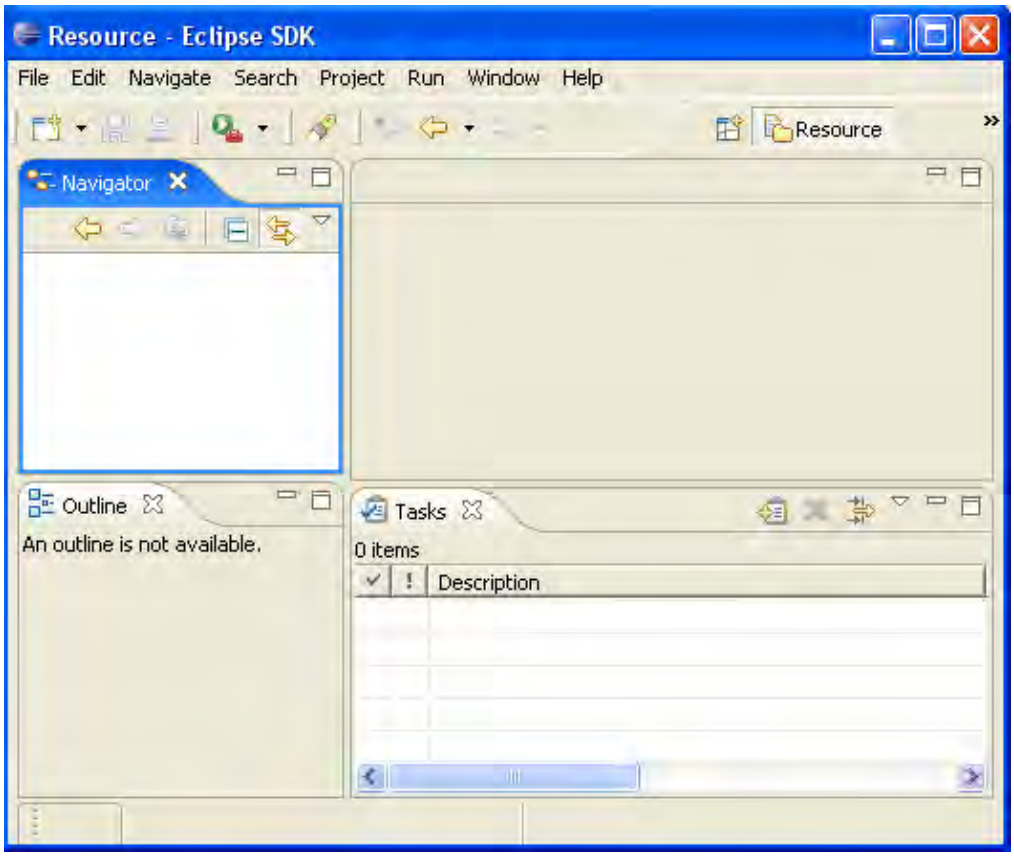
When the Workbench is launched the first thing you see is a dialog that allows you to select where the workspace should be located. The workspace is the directory where your work will be stored. For now, just click OK to pick the default location. (You can also check the checkbox to prevent this question from being asked again.)

After the workspace location is chosen, a single Workbench window is displayed. A Workbench window offers one or more perspectives. A perspective contains editors and views, such as the Navigator. Multiple Workbench windows can be opened simultaneously. Initially, in the first Workbench window that is opened, the Resource perspective is displayed, with only the Welcome view visible. Click the arrow labeled **Workbench** in the Welcome view to cause the other views in the perspective to become visible.

(You can get the Welcome view back at any time by selecting **Help > Welcome**.)

A shortcut bar appears in the top right corner of the window. This allows the user to open new perspectives and switch between ones already open. The name of the active perspective is shown in the title of the window and its item in the shortcut bar is highlighted.

The title bar of the Workbench window indicates which perspective is active. In this example, the Resource perspective is in use. The Navigator, Tasks, and Outline views are open along with an editor.



Editors and views

Prior to commencing the Workbench tutorials found in this section, it is important to first be familiar with the various elements of the Workbench. A Workbench consists of:

- perspectives
- views
- editors

A perspective is a group of views and editors in the Workbench window. One or more perspectives can exist in a single Workbench window. Each perspective contains one or more views and editors. Within a window, each perspective may have a different set of views but all perspectives share the same set of editors.

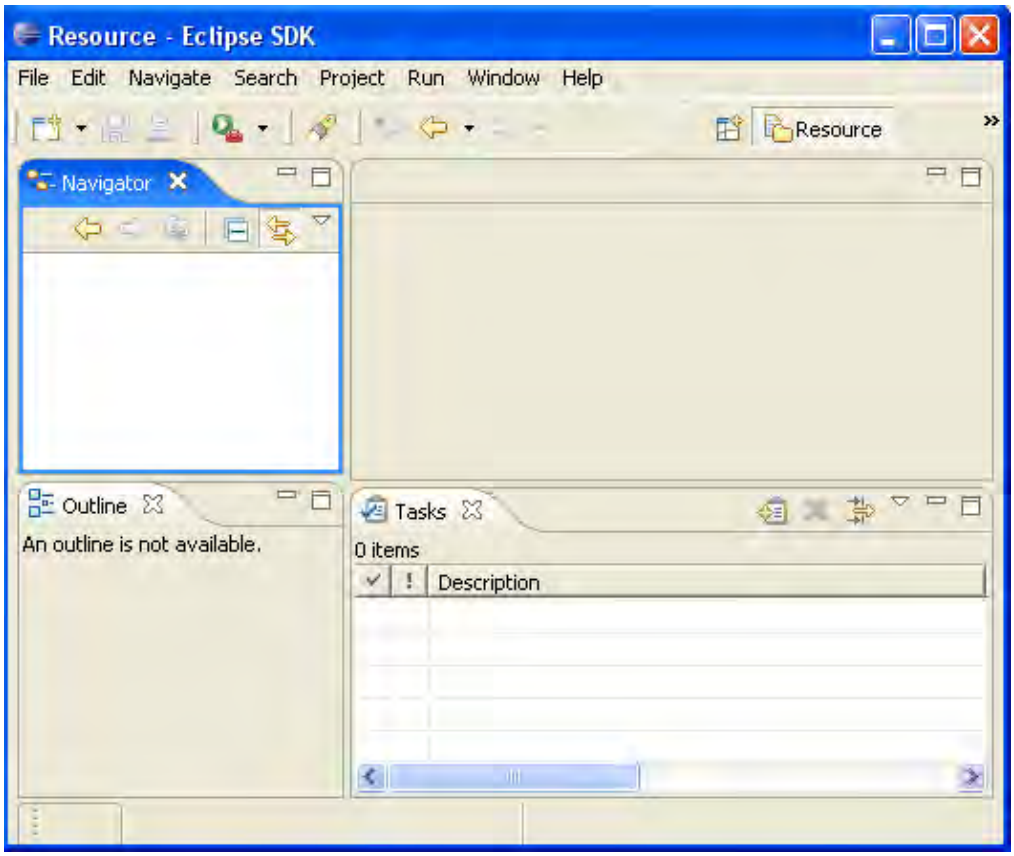
A view is a visual component within the Workbench. It is typically used to navigate a hierarchy of information (such as the resources in the Workbench), open an editor, or display properties for the active editor. Modifications made in a view are saved immediately. Normally, only one instance of a particular type of view may exist within a Workbench window.

An editor is also a visual component within the Workbench. It is typically used to edit or browse a resource. Modifications made in an editor follow an open−save−close lifecycle model. Multiple instances of an editor type may exist within a Workbench window.

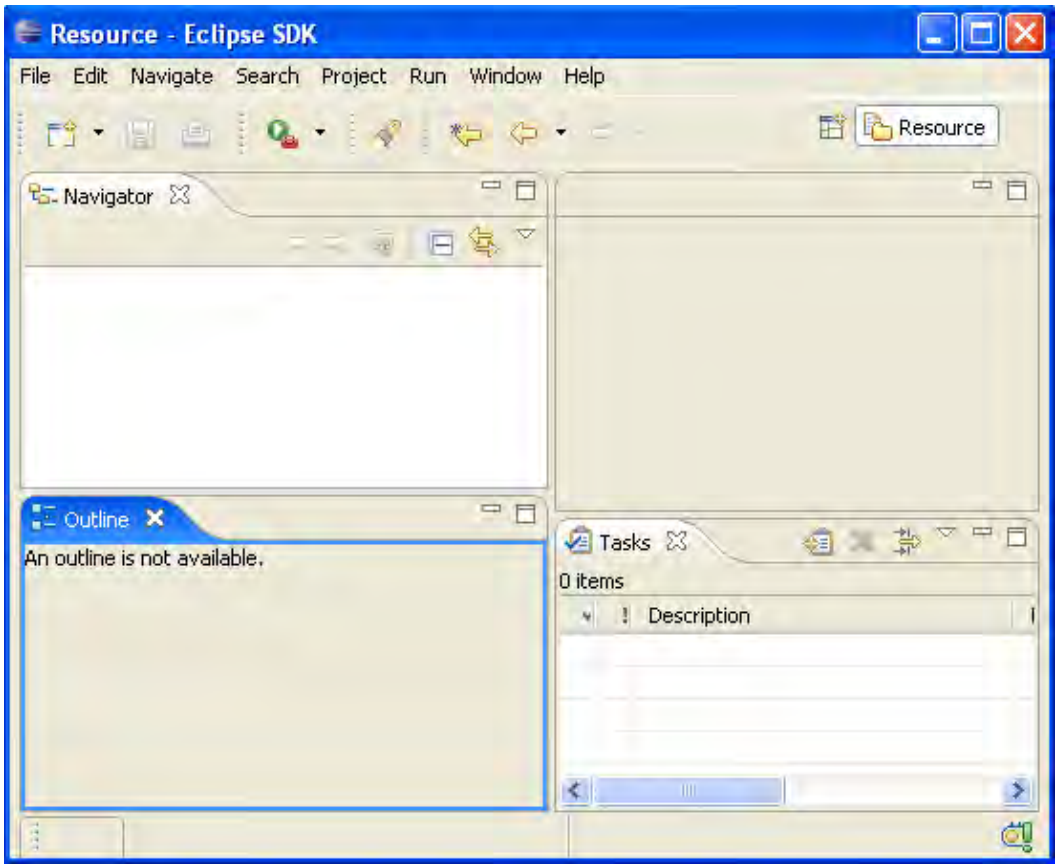
Some features are common to both views and editors. We use the term "part" to mean either a view or an editor. Parts can be active or inactive, but only one part can be active at any one time. The active part is the one whose title bar is highlighted. The active part is the target for common operations like cut, copy and paste.

The active part also determines the contents of the status line. If an editor tab is not highlighted it indicates the editor is not active, however views may show information based on the last active editor.

In the image below, the Navigator view is active.

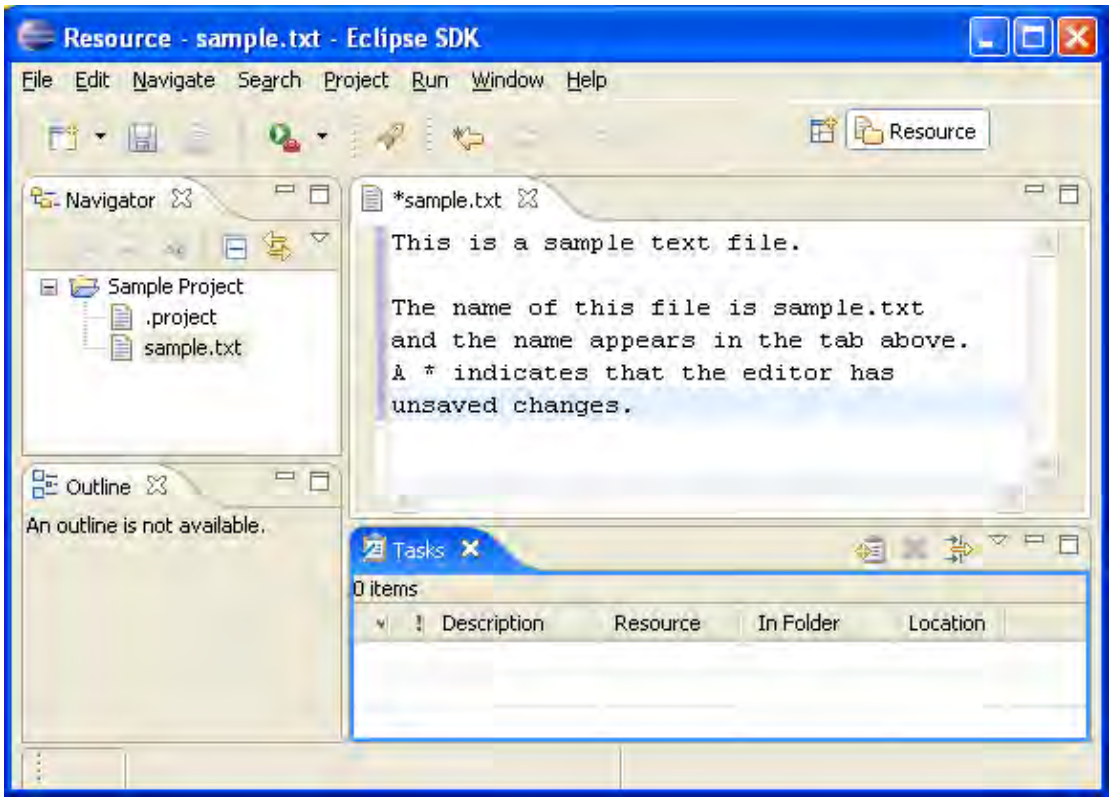


Clicking on the Outline view causes the Outline's title bar to become highlighted and the Navigator's title bar to no longer be highlighted, as shown below. The Outline view is now active.



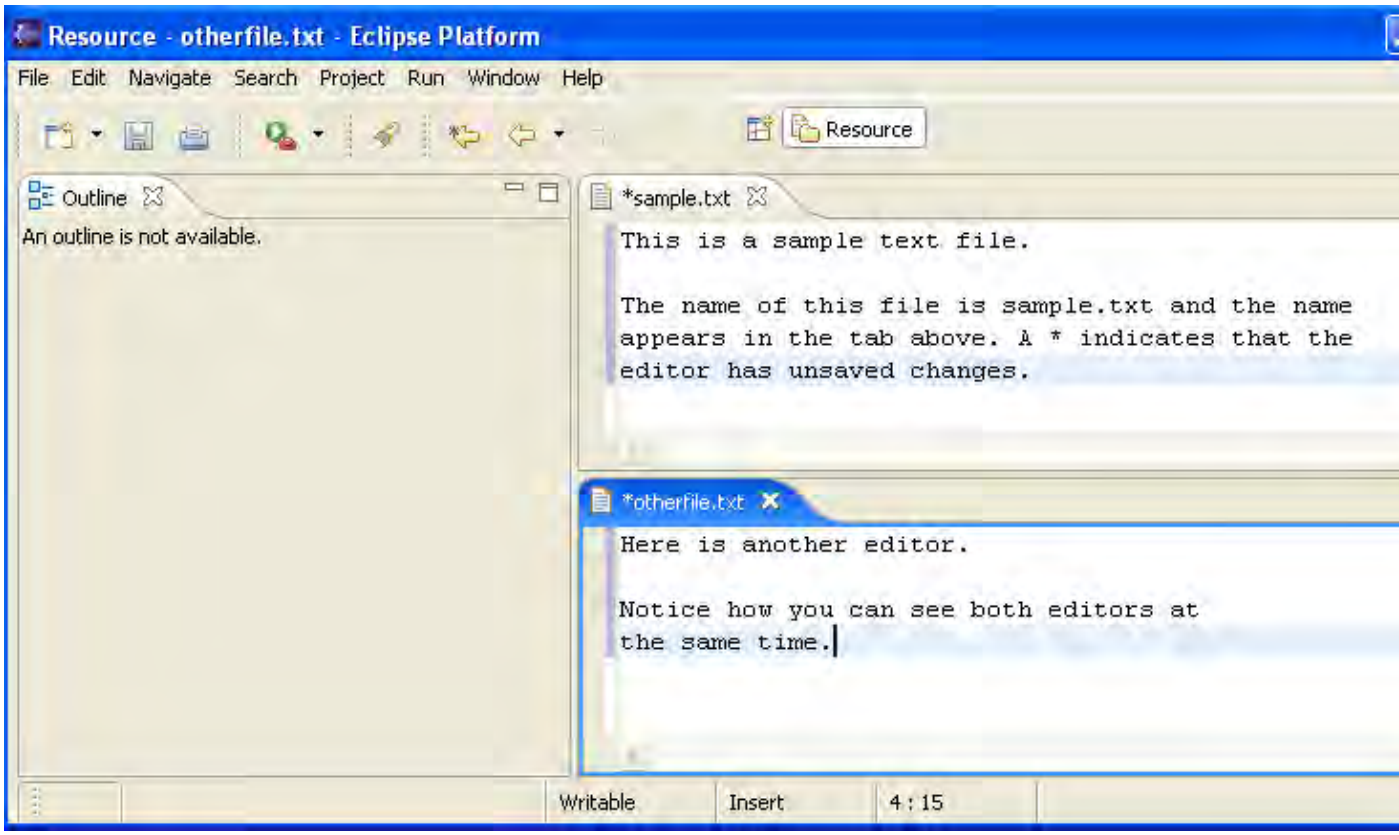
Editors

Depending on the type of file that is being edited, the appropriate editor is displayed in the editor area. For example, if a .TXT file is being edited, a text editor is displayed in the editor area. The figure below shows an editor open on the file sample.txt. The name of the file appears in the tab of the editor. If an asterisk (*) appears at the left side of the tab this indicates that the editor has unsaved changes. If an attempt is made to close the editor or exit the Workbench with unsaved changes a prompt to save the editor's changes will appear.



When an editor is active the Workbench menu bar and toolbar contain operations applicable to the editor. When a view becomes active, the editor operations are disabled. However, certain operations may be appropriate in the context of a view and will remain enabled.

The editors can be stacked in the editor area and individual editors can be activated by clicking the tab for the editor. Editors can also be tiled side−by−side in the editor area so their content can be viewed simultaneously. In the figure below, editors for sample.txt and otherFile.txt have been placed one above the other. Instructions will be given later in this tutorial explaining how to rearrange views and editors.



If a resource that does not have an associated editor is opened, the Workbench will attempt to launch an external editor registered with the platform. These external editors are not tightly integrated with the Workbench and are not embedded into the Workbench's editor area.

Also, the editors can be cycled through using the back and forward arrow buttons in the toolbar and by using the Ctrl+F6 accelerator. The arrow buttons move through the last mouse selection points and permits moving through several points in a file before moving to another one. Ctrl+F6 pops up a list of currently selected editors, by default, the editor used before the current one is selected. (On the Macintosh, the accelerator is Command+F6.)

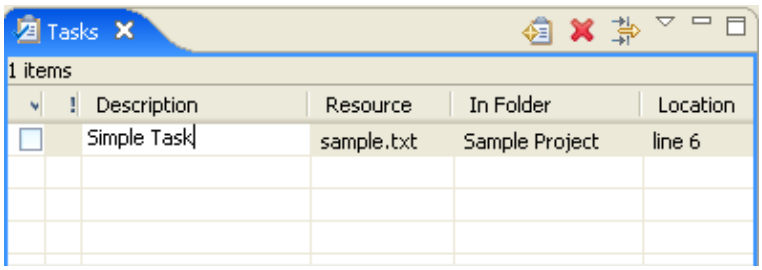
WIN On Windows the Workbench will also attempt to launch the editor in−place as an OLE document editor. For example, editing a DOC file will cause Microsoft Word to be opened in−place within the Workbench if Microsoft Word is installed on the machine. If Microsoft Word has not been installed, Word Pad will open instead.

Views

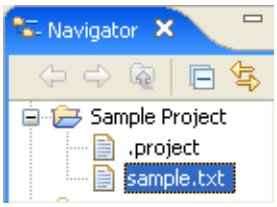
Views support editors and provide alternative presentations or navigations of the information in the Workbench. For example:

- The Bookmarks view displays all bookmarks in the Workbench along with the names of the files with which the bookmarks are associated.
- The Navigator view displays the projects and other resources.

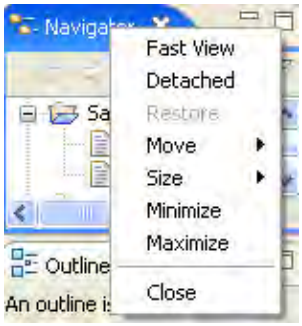
A view might appear by itself or stacked with other views in a tabbed notebook.



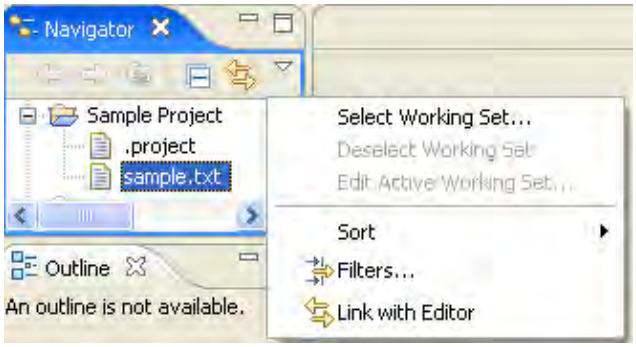
To activate a view that is part of a tabbed notebook simply click its tab. The Workbench provides a number of quick and easy ways to configure an environment, including whether the tabs are at the bottom or top of the notebooks.



Views have two menus, the first, which is accessed by right clicking on the view's tab, allows the view to be manipulated in much the same manner as the menu associated with the Workbench window.

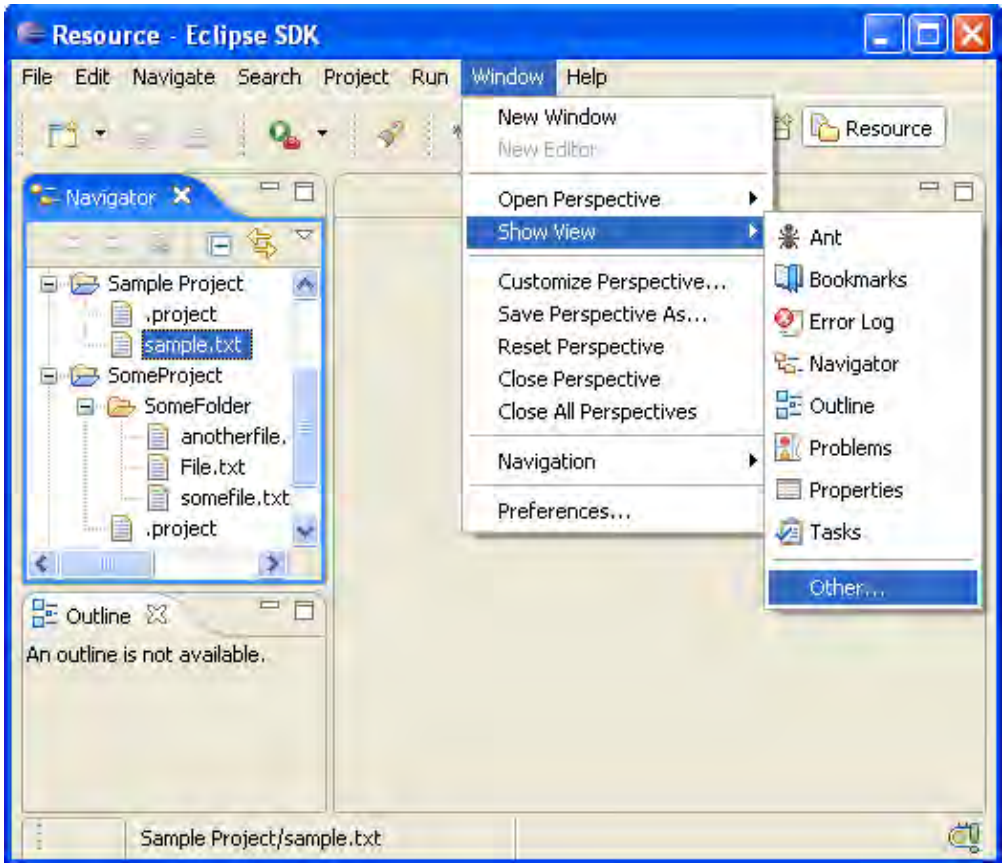


The second menu, called the "view pull−down menu", is accessed by clicking the down arrow . The view pull−down menu typically contains operations that apply to the entire contents of the view, but not to a specific item shown in the view. Operations for sorting and filtering are commonly found on the view pull−down.



Customizing the Workbench is a good opportunity to use the **Window > Reset Perspective** menu operation. The reset operation restores the layout to its original state.

A view can be displayed by selecting it from the **Window > Show View** menu. A perspective determines which views may be required and displays these on the **Show View** submenu. Additional views are available by choosing **Other...** at the bottom of the **Show View** submenu. This is just one of the many features that provide for the creation of a custom work environment.



A simple project

Now that the basic elements of the Workbench have been explained, here are some instructions for creating a simple project. New projects, folders, and files can be created using several different approaches. In this section resources will be created using three different approaches:

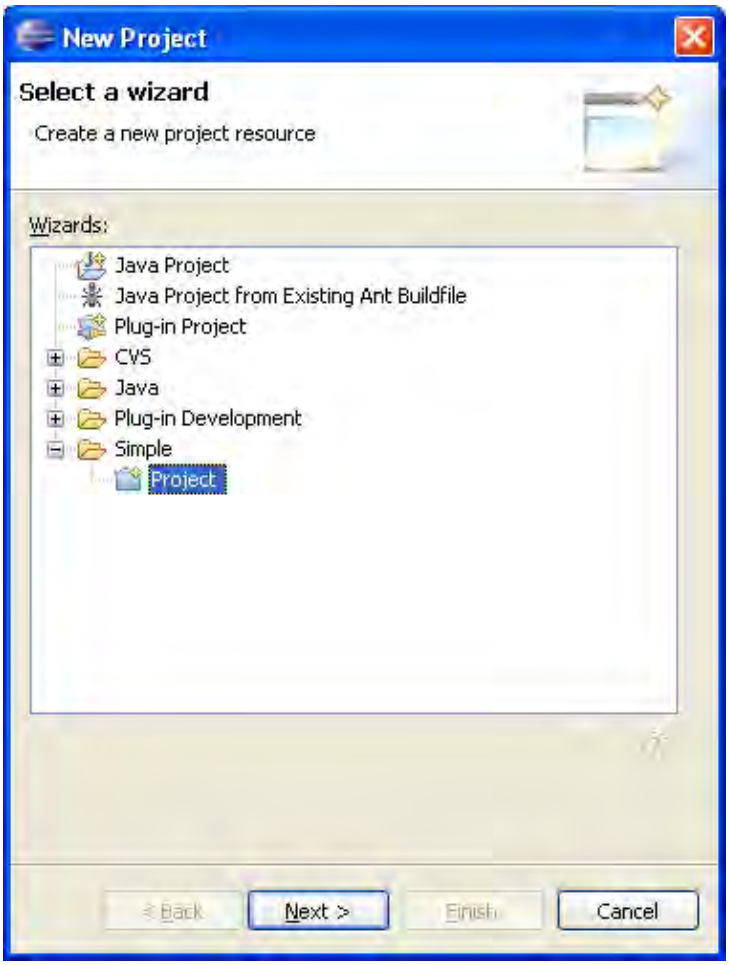
1. **File** menu
2. Navigation view context menu
3. New Wizard button

A project can be created using the **File** menu. Once the project has been created a folder and file can be created as well.

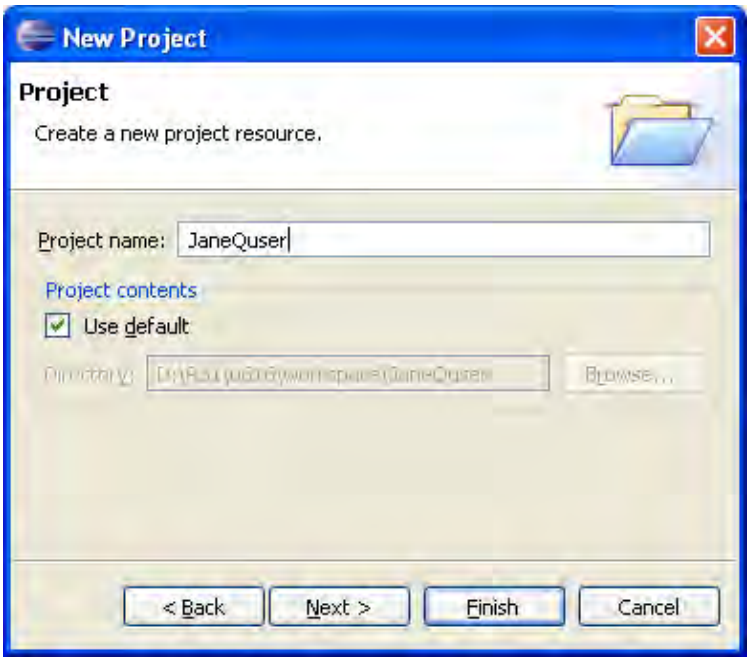
Using the File menu

You can create new resources by using the **File** > **New** menu on the Workbench menu bar. Start by creating a simple project as follows:

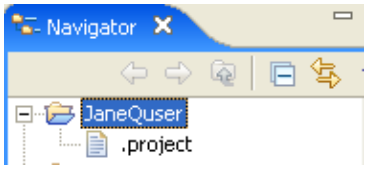
1. From the menu bar, select **File** > **New** > **Project...**



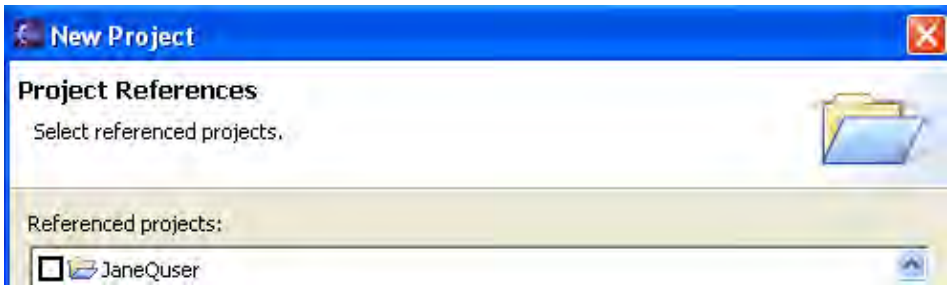
2. In the New Project wizard, select **Simple** > **Project** then click **Next**.
3. In the **Project name** field, type your name as the name of your new project. Do not use spaces or special characters in the project name (for example, "JaneQuser").
4. Leave the box checked to use the default location for your new project. Click **Finish** when you are done.



If you sneak a peek at the navigation view, you will see that it now contains the simple project we just created.



Create a second project called JaneQuser2 using the same steps, but instead of clicking **Finish**, click **Next**. At this point you can specify other projects that project JaneQuser2 depends on. Since we want to create two independent projects we will not select any of the projects in the **Project References** table. Click **Finish** to create your second simple project.



Using the popup

Now that we have our project we will create a folder. We will create our folder using the Navigator view's popup menu.

1. Activate the Navigator view and select the project JaneQuser (the first project we created in the

Navigator view). From the view's popup menu choose **New** > **Folder**.

2. In the New Folder wizard, your project name appears by default in the **Enter or select the parent folder** field. This is because we chose to create the new folder from your project's context menu.
3. In the **Folder name** field, type a unique name for your new folder (for example, using your first and last names in the title). Do not use spaces or special characters in the folder name (for example, "JanesFolder").

4. Click **Finish** when you are done. The Navigator view will update to show your newly created folder.

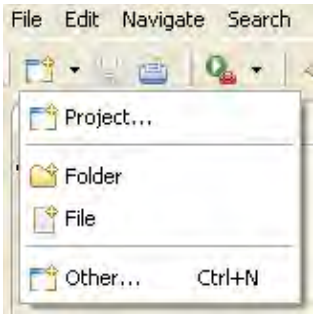
Note: There is now also an **Advanced** button. This button when selected allows you to enter a location outside of a project's hierarchy as the location for one of its folders. This is called a linked folder.

Using the New button

We have seen how to create resources using **File** > **New** and **New** from the context menu of one of the navigation views. We will now create a file for our project using the third alternative, the toolbar **New** button.

1. Select the folder JanesFolder in one of the navigation views.

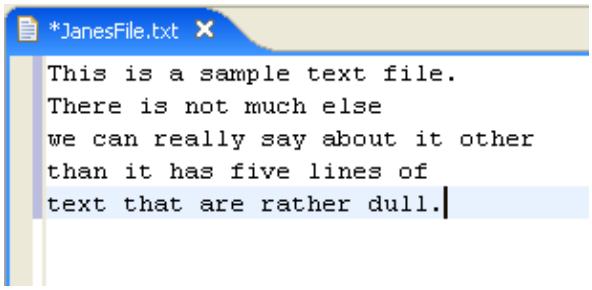
2. In the Workbench window's toolbar, activate the drop−down menu on the New Wizard button  and select **File**. To activate the drop−down menu simply click on the down arrow.



3. In the New File wizard, your folder's name and path already appear by default in the **Enter or select the parent folder** field. This is because you chose to create the new file while this folder was selected in one of the navigation views and the view was active.
4. In the **File name** field, type a unique name for a new text file, including the .txt file extension. Do not use spaces or special characters in this file name (for example, "JanesFile.txt").
5. Click **Finish** when you are done.
6. The Workbench has an editor capable of editing text files. A text editor is automatically opened on the newly created file.
7. In the text editor, type in the following five lines:

   This is a sample text file.
   There is not much else
   we can really say about it other
   than it has five lines of
   text that are rather dull.

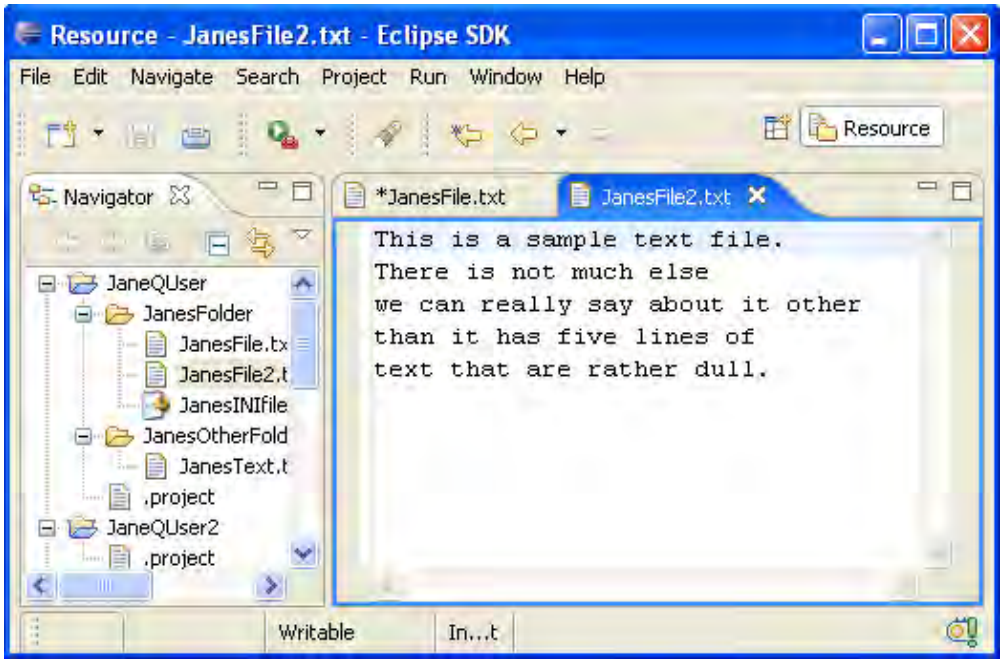
   Notice that the editor tab has an asterisk (*) at the left of the filename. The asterisk indicates that the editor has unsaved changes.



8. In the Workbench window's toolbar, click the Save button  to save your work.
9. In one of the navigation views ensure your folder is still selected and the navigation view is active.
10. Click ***New Wizard*** in the Workbench toolbar. Previously we clicked on the drop−down arrow of the New button. Here we clicked on the button itself which has the same effect as choosing **File > New > Other...**
11. In the New wizard, select **Simple > File**. Then click **Next**.
12. Once again, your folder's name and path appears by default in the **Enter or select the parent folder** field.
13. In the **File name** field, type a unique name for an .ini file. Do not use any spaces or special characters in the file name (for example, "JanesINIFile.ini"). Click Finish when you are done.

14. Since the Workbench does not have any editors registered for .ini files, it may launch an external editor on the file if one is registered with the operating system. For the moment, close the editor.

Now that we have created our resources, one of the navigation views shows our two projects, the folder and its two files. To the right of one of the navigation views is the text editor open on the first file we created (JanesFile.txt). To proceed with the following example create a second file (JanesFile2.txt).



Click the new JanesFile2.txt editor tab. Now select the file JanesINIFile.ini in one of the navigation views. Then select **Link with Editor** on one of the navigation views. Lastly, click on the editor tab for JanesFile.txt. Notice how the navigation view updated itself to select the file you are currently editing (JanesFile.txt). If you don't like this automatic update you can easily turn it off by deselecting **Link with Editor**.

Closing an editor

Now that there are a couple of editors open, here's how to close them.

1. Select the JanesFile.txt editor tab.
2. In the text area add a 6th line of text:

   This is a 6th line
3. To close the editor, choose one of the following options:

   - Click the close button ("X")  in the tab of the editor.
   - Select **File > Close** from the menu bar.
4. Note the prompt to save the file before the editor is closed.
5. Click OK to save any changes and close the editor.

If the editor was closed using **File > Close**, notice that the option **File > Close All** was also displayed. This is a quick way to close all of the open editors. If **File > Close All** is chosen, a prompt will appear to choose which editors with unsaved changes should be saved.

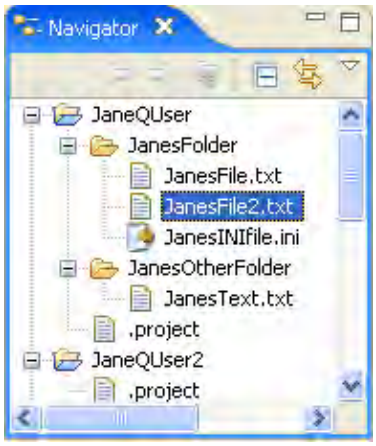
The preference to close editors automatically can be found in **Window > Preferences** then **General > Editors**. The number of editors that can be opened prior to editors being reused and what should occur when all editors have unsaved changes can be configured there.

Navigating resources

This section will work with the Navigator and Tasks views. These views are initially part of the resource perspective. To experiment with other views, they can be displayed by using the **Window > Show View** menu.

One important view to become familiar with is one of the navigation views, which displays information about the contents of the Workbench and how the resources relate to each other in a hierarchy.

In the Workbench, all resources reside in projects. Projects can contain folders and/or individual files.



Opening resources in the Navigator

Using the Navigator there are several ways to open an editor.

1. In one of the navigation views select the file JanesFile.txt
2. To open an editor on the file choose one of the following approaches:

    - To edit a resource using the default editor for that resource, either double click on the resource (in one of the navigation views), or select the resource and choose **Open** from its popup menu.
    - To use a specific editor to edit the resource, start by selecting the resource in one of the navigation views, and choose the **Open With** option from the popup menu.

   The Workbench remembers the last editor that was used for editing a specific file. This makes it easier to use the same editor down the road.

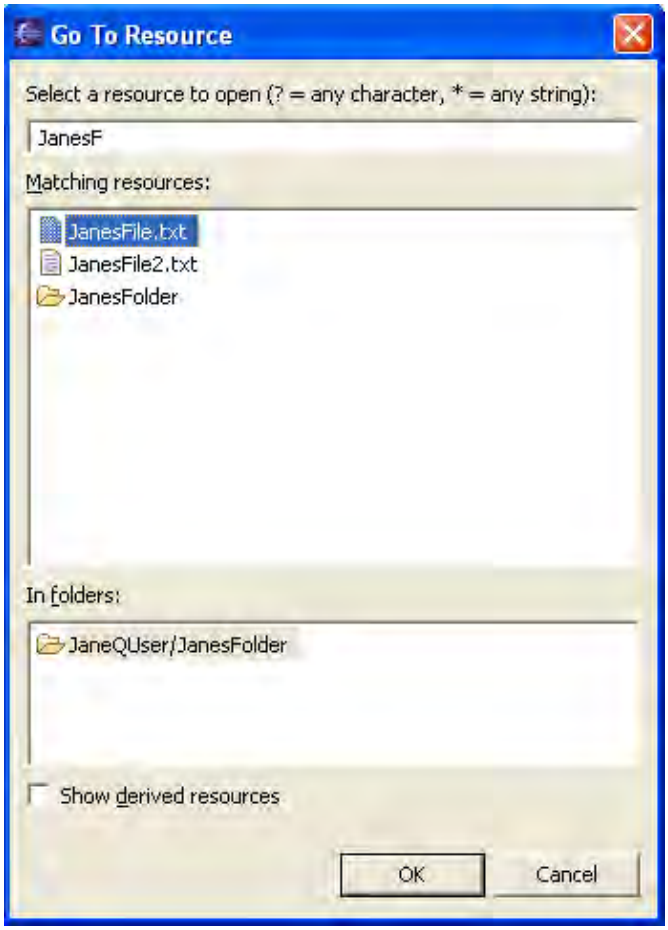
The default editors can be configured using **Window > Preferences > General > Editors > File Associations**.

Go To

The Go To operation makes it easy to jump to a specific resource in the navigation views.

1. Select one of the navigation views. Its title bar will be highlighted (according to the operating system's color scheme).
2. From the menu bar choose **Navigate > Go To > Resource...**.
3. In the Go To Resource dialog type "JanesF" into the **Pattern** field at the top of the dialog.

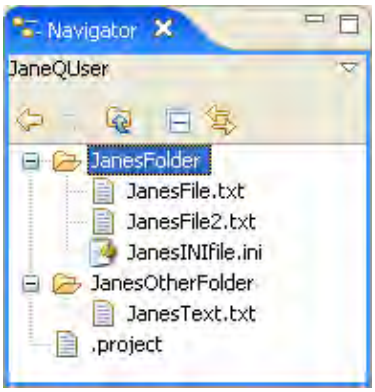
   As the filename is typed, the dialog filters the set of possible matches based on what has been entered so far.
4. Select JanesFile.txt from the **Matching Resources** field and click OK or simply press Enter.
5. The navigation view will select the file JanesFile.txt.



Go Into

The navigation view shows all of the resources in the Workbench. Select a project or folder, choose to "Go Into it" and hide all other resources as demonstrated with the JaneQuser project.

1. Select the Navigator view. Its title bar will be highlighted (according to the operating system's color scheme).
2. Select the JaneQuser project.
3. Choose **Go Into** from the popup menu.
4. The Navigator now shows only the contents of the JaneQuser project. The title of the Navigator shows the name of the resource that is currently displayed.



5. The back, forward and up buttons  can be used to move between showing all resources and showing only the JaneQuser project.

   Click the back button to show all of the resources.

Files

The projects, folders and files that you create with the Workbench are all stored under a single directory that represents your workspace. The location of the workspace was set in the dialog that first opens when you start the Workbench.

If you have forgotten where that location is, you can find it by selecting **File > Switch Workspace...**. The workspace directory will be displayed in the dialog that appears. **IMPORTANT:** After recording this location, hit **Cancel** to close the dialog, or the Workbench will exit and re−open on whatever workspace was selected.

All of the projects, folders and files that you create with the Workbench are stored as normal directories and files on the machine. This allows the use of other tools when working with the files. Those tools can be completely oblivious to the Workbench. A later section will look at how to work with external editors that are not integrated into the Workbench.

Exporting files

Files can be exported from the Workbench either by:

- Dragging and dropping to the file system (Windows and Linux GTK only), or
- WIN Copying and pasting to the file system, or
- Using the Export wizard.

Drag and drop or copy and paste

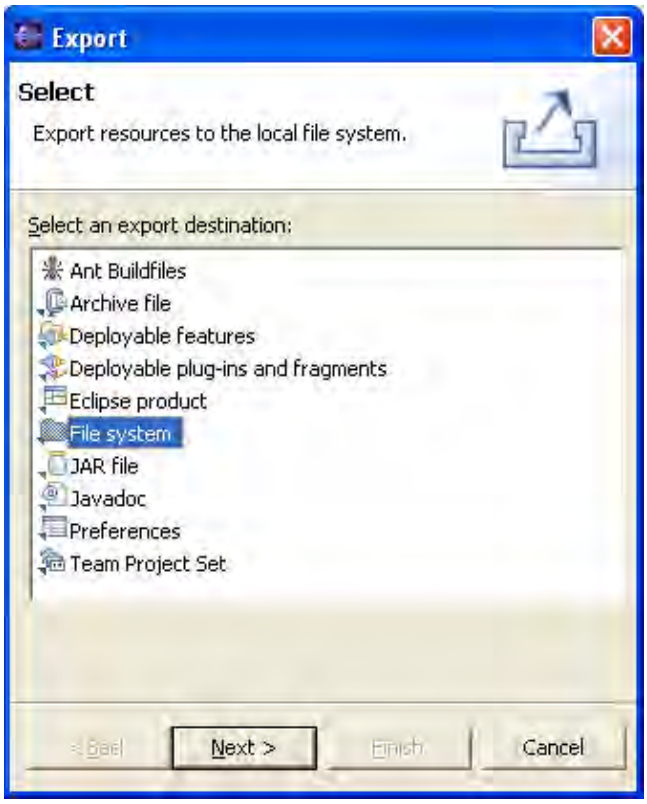
WIN The operating system's file system explorer can be used to export a copy of a folder or file from the Workbench to the file system.

1. Open the operating system's file system explorer.
2. Drag the file JanesFile.txt from one of the navigation view to the file system explorer.
3. Depending on where you are dragging to, you may need to hold down the Ctrl or Shift key while dragging to ensure the file is copied. Look for a small plus sign on the drag cursor to know whether the file is being copied or moved.
4. The export can also be achieved by selecting the file in the Navigator and choosing **Edit > Copy**, then pasting it in the file system explorer.

Export wizard

The Export wizard can be used to export from the Workbench to the file system.

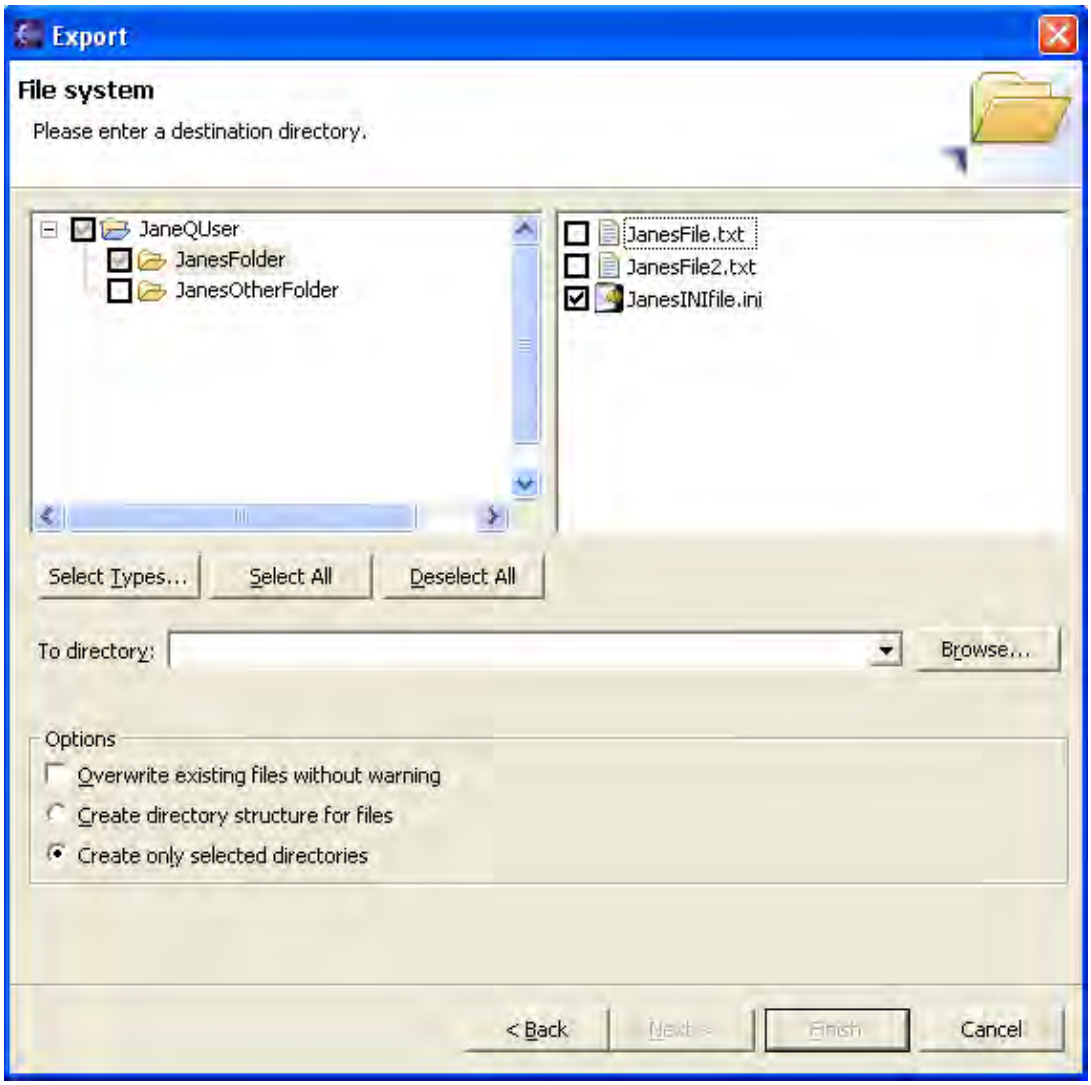
1. Select the project JaneQuser in the navigation view.
2. From the popup menu, select **Export**.
3. In the Export wizard, select **File system**, then click **Next**.



4. Expand JaneQuser project, and click on JanesFolder. In the right pane ensure that only JanesINIFile.ini is selected. Notice the folder and project in the left pane now have a grayed checkbox indicating that some, but not all, of their contents will be exported.

   The **Select Types** button can be used to filter the types of resources to export.

   Note: To export JanesINIFile.ini only, simply select it in the navigation view and choose **File** > **Export**. The Export wizard will automatically ensure it is the only file selected for export.



5. In the **To directory** field, type or browse to select a location in the file system for the exported resources to reside.

   If the name of a directory that does not exist is entered the Export wizard will offer to create it once **Finish** is selected.
6. In the **Options** area, options are given to:

   - ♦ Overwrite existing resources without warning
   - ♦ Create directory structure for files or Create only selected directories
7. Click **Finish** when done.

Importing files

Files can be imported into the Workbench either by :

- dragging and dropping from the file system, or
- copying and pasting from the file system, or
- Using the Import wizard.

Using drag and drop or copy/paste to import files relies on operating system support that is not necessarily available on all platforms. If the platform you are using does not have this support, you can always use the Import wizard.

In this section the two files that were just exported will be imported and placed into the second project JaneQuser2.

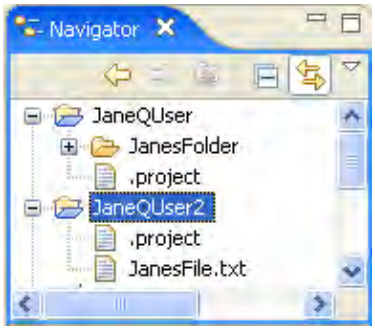
Drag and drop or copy and paste

On some platforms, for example on the Macintosh and Microsoft Windows, the operating system's file system browser can be used to copy folders and files from the file system into the Workbench.

Note: The resource(s) must be dragged to the exact location in the hierarchy of one of the navigation views where the resources are to reside; they cannot be simply dragged and dropped onto a blank area in the navigation view.

1. Open the operating system's file system explorer.
2. Locate the file JanesFile.txt that was recently exported and drag it to a specific location in one of the navigation views in the Workbench.

   When dragging resources into one of the navigation views, the project/folder that the resource is being dropped into will be selected.

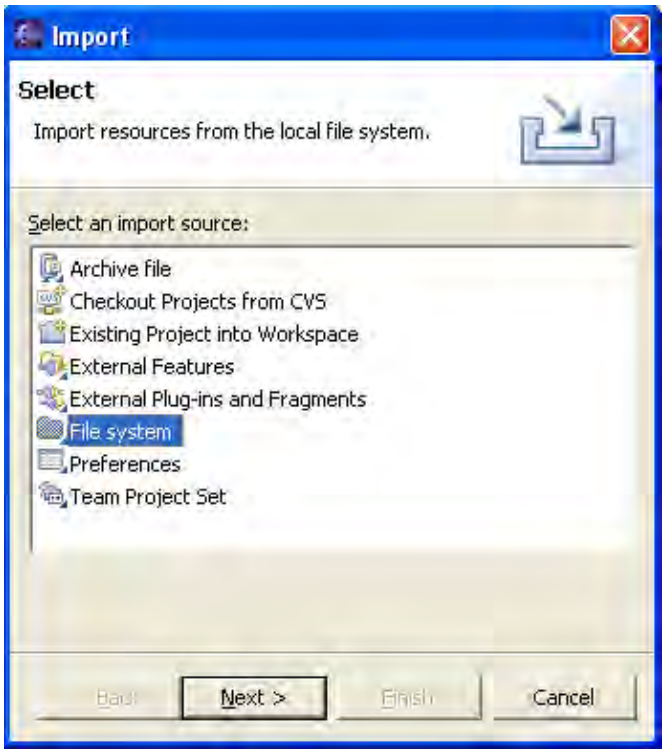


   Drag the file over JaneQuser2 and release the mouse button.
3. Notice that the file is copied into the Workbench and placed into JaneQuser2.
4. This can also be achieved by copying the file in the file system explorer, then selecting the destination in the navigation view and choosing **Edit > Paste**.

Import wizard

The Import wizard can be used to copy resources into the Workbench.

1. Select the project JaneQuser.
2. Select **Import** from the popup.
3. In the Import wizard, select **File system**, then click **Next**.



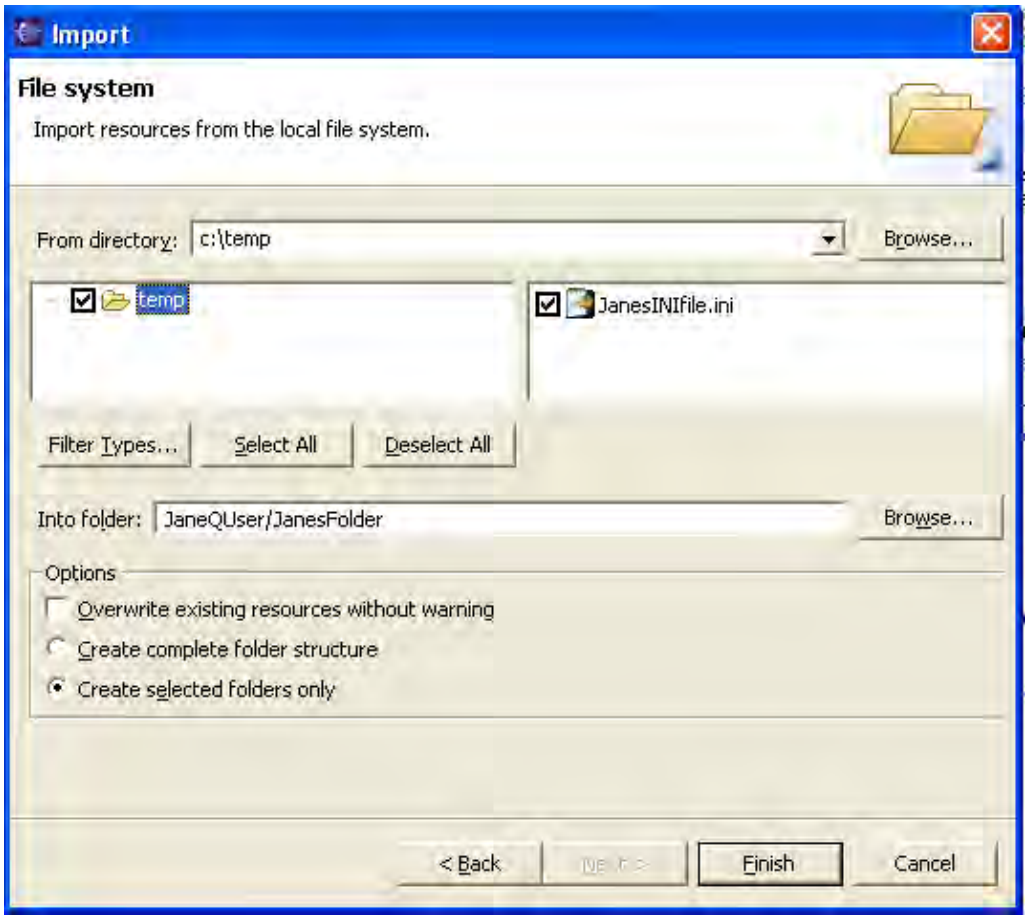
4. In the **From directory** field, type or browse to select the directory containing the file JanesINIFile.ini that was recently exported.

   Recent directories that have been imported from are shown on the **From directory** field's combo box.

5. In the right pane check the file JanesINIFile.ini

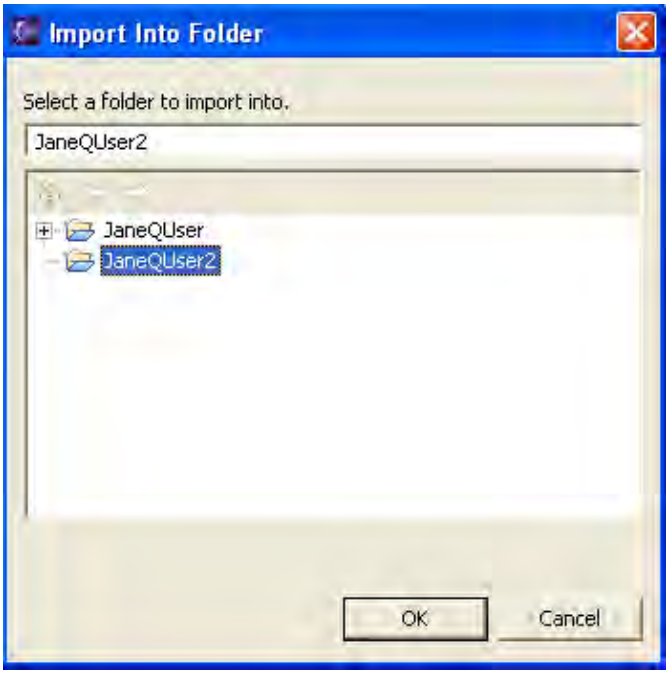
   Checking a folder in the left pane will import its entire contents into the Workbench. A grayed checkbox (as shown below) indicates that only some of the files in the folder will be imported into the Workbench.

   The **Filter Types** button can be used to filter the types of files that will be imported.



6. The **Into folder** field should already be filled in with the name of the project (JaneQuser).
7. The destination project or folder can be changed by clicking **Browse**;

   Click the **Browse** button and choose the second project JaneQuser2.



8. In the **Options** area, options are given to:

- Overwrite existing resources without warning
- Create complete folder structure or Create selected folders only

9. Click **Finish** when done. The file JaneINIFile.ini is now shown in the one of the navigation views in the project JaneQuser2.

Deleting resources

Now that a few files have been imported into the second project (JaneQuser2), here are instructions on how to delete the project.

1. Select project JaneQuser2 in one of the navigation views.
2. To delete the project do one of the following:

   - From the project's pop−up menu choose **Delete**
   - Press the DEL key
   - Choose **Edit > Delete** from the pull−down menu

3. A prompt will ask for confirmation of the deletion and also what type of deletion should be performed.

   It is possible to:

   - Delete the contents of the project from the file system
   - Delete the project from the workspace but keep its contents in the file system
   - Accept the default (do not delete contents) and click **Yes**.

Note: This is only an option for projects. Files and folders are always deleted from the file system when deleted.

The same steps work for any resource shown in the navigation view.

Working with other editors

Instructions have been given explaining how to import and export resources from the Workbench. This section will look at how to edit Workbench resources using the following three approaches:

- External editors launched by the Workbench
- Embedded OLE documents WIN
- External editors launched without the Workbench's knowledge

Before continuing, take a moment and confirm that the Navigator contains the following resources:

External editors

When opening a resource the Workbench first consults its list of registered editors. If no registered editors are found for the resource the Workbench checks with the underlying operating system to determine if it has any editors registered for the particular file type. If an editor is located, the Workbench will automatically launch that editor. This type of editor is referred to as an external editor because it does not show up as an editor tab in the Workbench.

1. Select the file JanesINIFile.ini.
2. Double−click the file in one of the navigation views to launch the external editor.

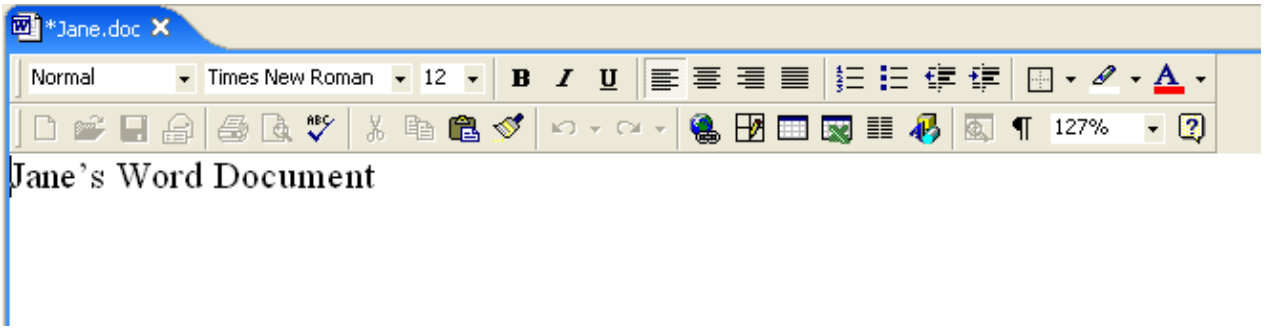
   If an editor for INI files is not registered with the underlying operating system the Workbench will attempt to use its own default text editor. If this happens, to see an external editor, import another file (see previous sections) that is associated with a third party editor. Double click again on this new file and the selected editor will open in its own window.

   WIN

   The Workbench supports OLE document editors, and some editors provide OLE document support allowing it to be opened in its own window, or embedded inside another window like the Workbench. This will be discussed in more detail in the next section.

Embedded editors

WIN The Workbench supports OLE document editors.



1. Select JanesFolder in the navigation view.
2. Create the file Jane.doc.
3. Notice how the Workbench automatically opens the OLE document editor in place and integrates its pull−down menu options into the menu bar. There should also be an editor tab for Jane.doc.
4. Make a change to the file.
5. Close the editor by clicking the X on the editor tab; when prompted, save its contents.
6. Reopen the file by double−clicking it in the navigation view.

Editing files outside the Workbench

In a previous section an external editor was launched on the file JanesINIFile.ini by double clicking on it in the Navigator. The same external editor can also be used by launching it outside of the Workbench.

1. Close any editors that are open on JanesINIFile.ini.

2. In the file system explorer, navigate to the directory where the Workbench was installed and go into the workspace sub−directory.
3. Edit the file JanesINIFile.ini and save it . Do not use the Workbench's Navigator to open the editor.
4. Return to the Workbench and in the Navigator view, select the project JaneQuser.
5. Choose **Refresh** from the project's context menu. This instructs the Workbench to look for any changes to the project that have been made in the local file system by external tools.
6. Select the file JanesINIFile.ini.
7. For a little variety choose **Open With > Text Editor** from the file's popup menu.
8. Observe that the Workbench's own default text editor is opened.

   In the default text editor verify that the externally made changes are reflected in the Workbench.

The Workbench stores all of its resources in the local file system. This means the system file explorer can be used to copy files from the Workbench's workspace area even when the Workbench is not running. Resources can also be copied into the workspace directory. Use Refresh to update the Workbench with any changes made outside the Workbench.

Copying, renaming and moving

Workbench resources can be copied, moved and renamed using popup menu operations in one of the navigation views. In this section several of the files that have been created will be copied and renamed.

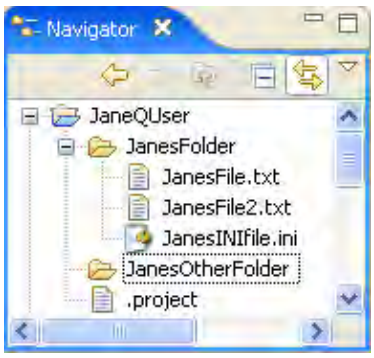
Prior to copying files, some setup is required:

Setup

1. In one of the navigation views, delete the file JanesWordDoc.doc. The navigation view should look like this:

2. Double click on JanesFile.txt and ensure that it contains the following text.

   This is a sample text file
   There is not much else
   we can really say about it other
   than it has five lines of
   text that are rather dull.
3. Close the editor on JanesFile.txt.
4. Select the project JaneQuser. Using the project's pop−up menu create a folder named JanesOtherFolder.



Copying

You can copy JanesFile.txt to the new folder (JanesOtherFolder) using the following steps.

1. Ensure that the setup described in the introduction to this section has been performed.

2. In one of the navigation views, select JanesFile.txt.
3. From the file's context menu, select **Copy** (or Ctrl+C)
4. In one of the navigation views, select JanesOtherFolder as the destination.
5. From the folder's context menu, select **Paste** (or Ctrl+V).

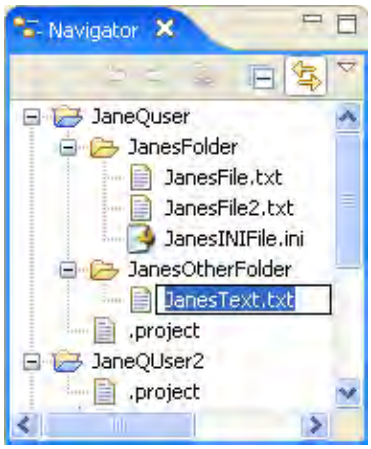
As an alternative to copying files using the copy operation, it is also possible to copy files by holding down the Ctrl key while dragging a file from one folder to another folder. (On the Macintosh, the Option key is used for this.)

Once the file has been copied it can be renamed.

Renaming

Now that JanesFile.txt has been copied from JanesFolder to JanesOtherFolder it is ready to be renamed as something else.

1. In one of the navigation views, select JanesFile.txt in JanesOtherFolder.
2. From the file's context menu, select **Rename**.
3. The navigation view overlays the file's name with a text field. Type in JanesText.txt and press Enter.



To halt the renaming of a resource, Escape can be pressed to dismiss the text field.

Copy and rename works on folders as well.

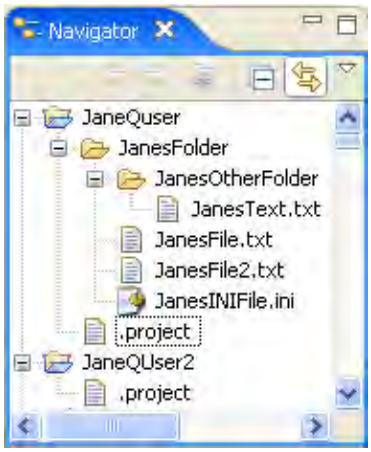
1. In one of the navigation views, select the folder JanesOtherFolder.
2. From the folder's context menu choose **Rename**.
3. Once again the navigation view overlays the folder name with an entry field to allow the typing in of a new name. Change the folder name to be JanesSecondFolder.
4. Rename the folder back to its original name (JanesOtherFolder).

Moving

Having copied and renamed several of the resources, now it's time to move some resources around. JanesOtherFolder and its file will be moved to be a sub−folder of the original folder JanesFolder.

1. In the Navigator view, select JanesOtherFolder.
2. From the file's context menu, select **Move**.
3. In the Folder Selection dialog choose JanesFolder and click **OK**.

4. In the Navigator JanesFolder now contains JanesOtherFolder. Expand JanesOtherFolder and confirm that it contains JanesText.txt.



As an alternative to moving files using the move operation, it is also possible to move files by dragging a file from one folder to another folder. Remember that to copy files from one folder to the other, Ctrl needs to be held down while performing the drag and drop operation. (On the Macintosh, the Option key is used for this.)

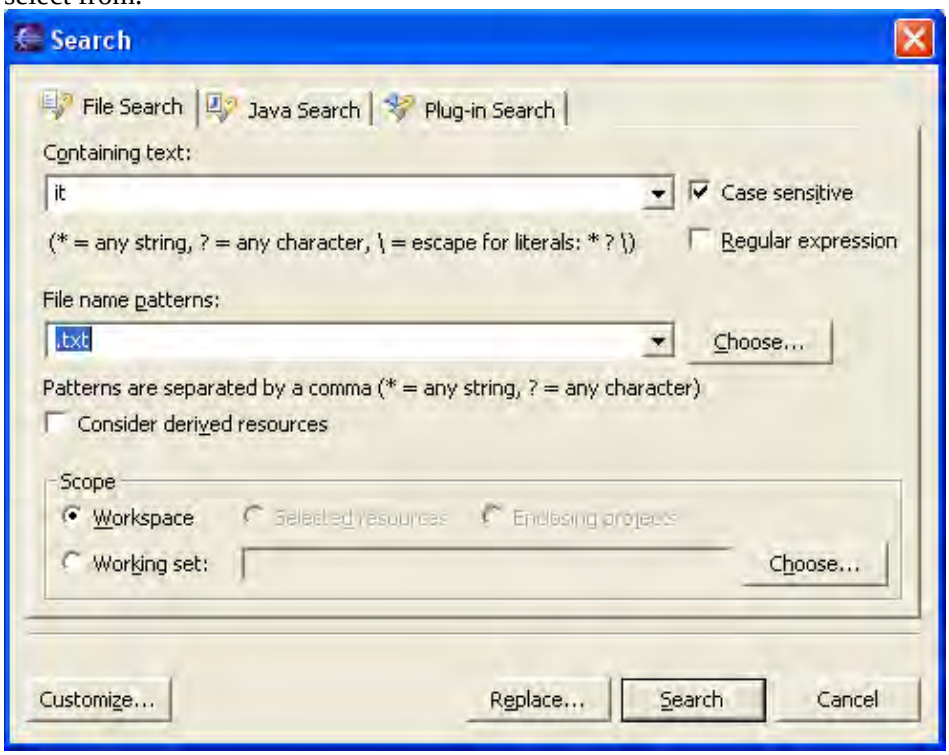
Searching

Text strings and files can be searched for in the Workbench. In this section, ***Search***  will be used to perform a text search across the resources that are shown in the navigation view. Instruction will also be given on how to use the Search view to work with the results.

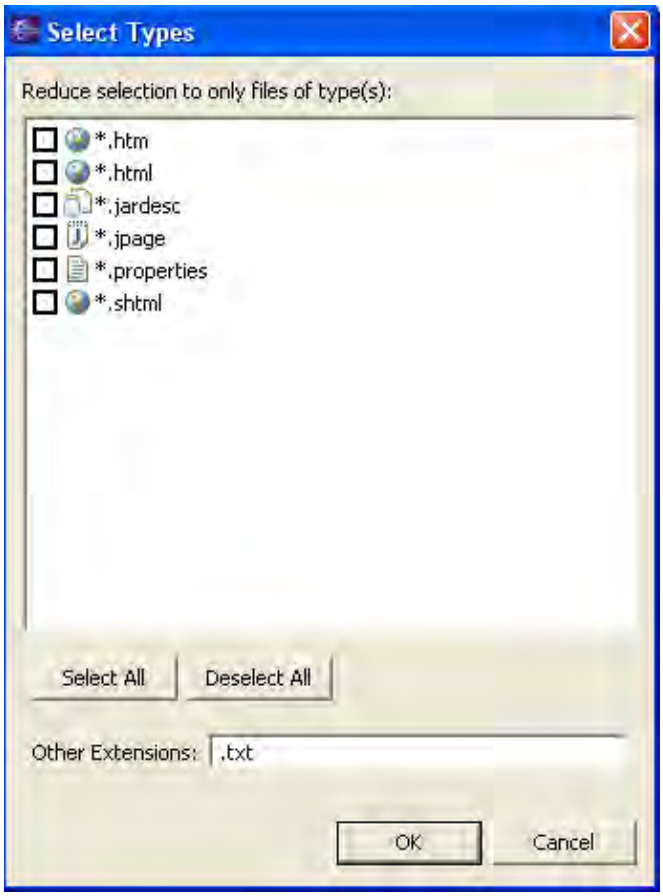
Starting a search

Text strings can be searched for in the Workbench as follows:

1. In the Workbench toolbar, click the search button  .
2. In the **Containing text** field, type in: *it*
   The combo box for the **Containing text** field also displays a list of recently performed searches to select from.

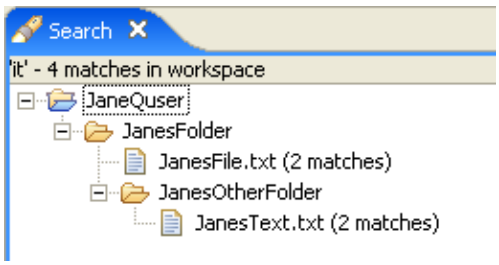


3. The **Case sensitive** checkbox can be selected or deselected depending on whether or not a case sensitive or insensitive search is to be performed. You can also select the **Regular expression** checkbox to enable more powerful searching capabilities. To see what is available in regular expression mode, you can hit Ctrl−Space over the text field to get content assistance that lists the possibilities. For this example, just check the **Case sensitive** box to search for lowercase "*it*".
4. The kinds of files to include in the search can be specified in the **File name patterns** field. Click **Choose...** to open the Select Types dialog. This dialog provides a quick way to select from a list of registered extensions.



For the moment, the search will be confined to .txt files. Ensure *.txt is checked and click **OK**.

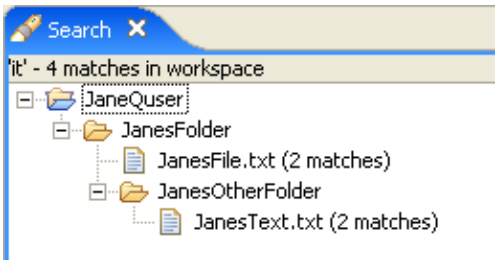
5. In the **Scope** field, specify the files and folders to include in the search. The choices are: the entire workspace, the currently selected resources in the Workbench, or a working set which is a named, customized group of files and folders. Leave the scope as workspace.
6. Use the **Customize** button to choose what kinds of searches are available in the dialog. This setting may be left unchanged.
7. Click **Search**. At this point, the Search view will be made visible, and it will begin to fill in with the results of the search. The stop button in the Search view can be clicked to cancel the search while it is in progress.
8. Observe that the Search view displays:



The next section will describe how to work with the Search view.

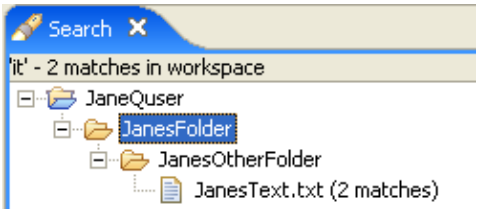
The Search view

Now that the search for "it" has been completed, the Search view is visible. The title of the Search view shows that four matches were found.

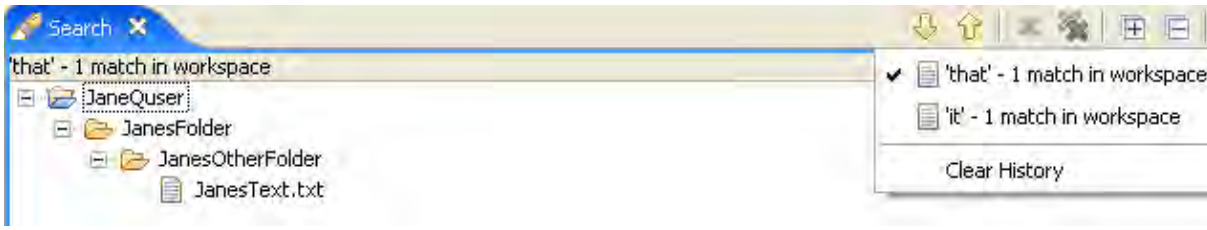


Within the Search view two files are shown and within each file there were 2 matches found.

1. Click the Show Next Match button to navigate to the first match of the search expression ("it"). Notice that the file JanesFile.txt is automatically selected and opened in the editor area.
   Click Show Next Match button two more times. Once again the Search view automatically opens the file (JanesText.txt).
2. It is sometimes useful to remove uninteresting matches from the search results. The Search view's popup menu allows you to do this using **Remove Selected Matches** which removes any selected file entries (and all matches in them) from the Search view. Note that this ***only*** removes the entries in the Search view, it does ***not*** effect the files themselves. Select JanesFile.txt and choose **Remove Selected Matches** from the popup menu. The Search view now shows only the matches for JanesText.txt



3. Perform a second search for "that" by clicking on the Search button in the Workbench's toolbar.
4. The Search view updates to show the results of the new search.
   Use the drop down button on the Search view's toolbar to move back and forth between the two search results.



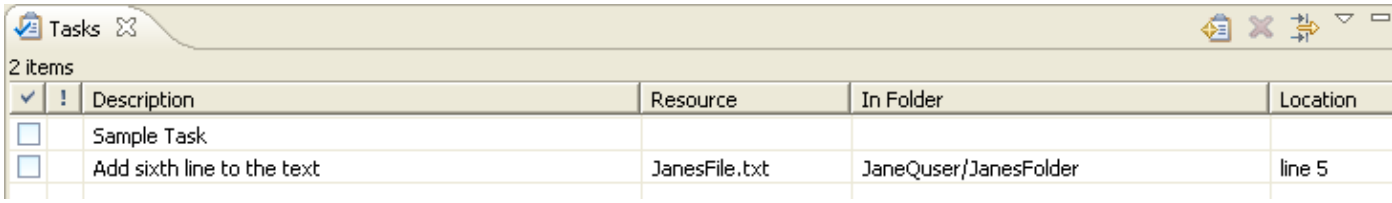
5. In the drop down button choose "it" − 1 Occurrence. The Search view switches back to show the original search. On the context menu choose **Search Again** to repeat the initial search. Notice that once again there are four matches.

Tasks and markers

There are various types of markers including bookmarks, task markers, debugging breakpoints and problems. This section will focus on tasks and the Tasks view.

The Tasks view displays all the tasks in the Workbench. The view displays tasks associated with specific files, specific lines in specific files, as well as generic tasks that are not associated with any specific file.

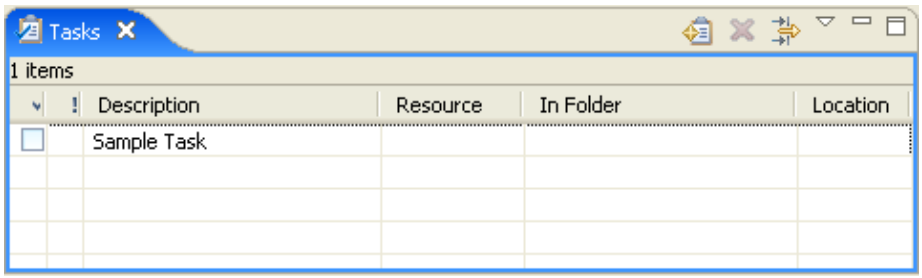
In the figure below "sample task" is a generic task not associated with any specific resource. The second task ("add sixth line to the text") is associated with the file JanesFile.txt.



Unassociated tasks

Unassociated tasks are not associated with any specific resource. To create an unassociated task:

1. In the Tasks view, click the New Task button  . A new task dialog appears.
2. Type a brief description for the task and press Enter. To cancel the dialog while entering the description press Escape . The new task appears in the Tasks view.

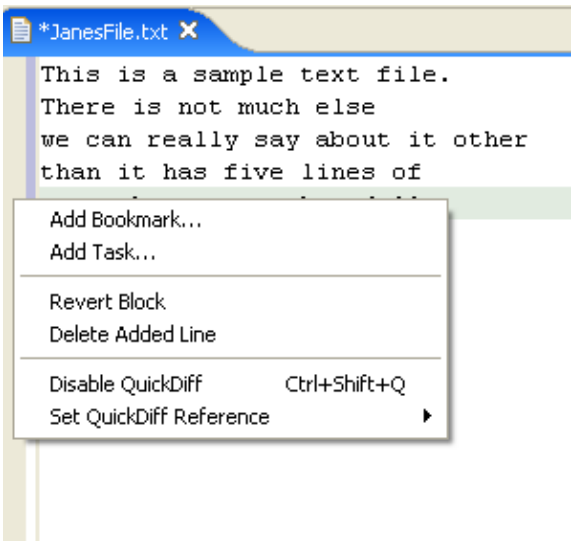


Associated tasks

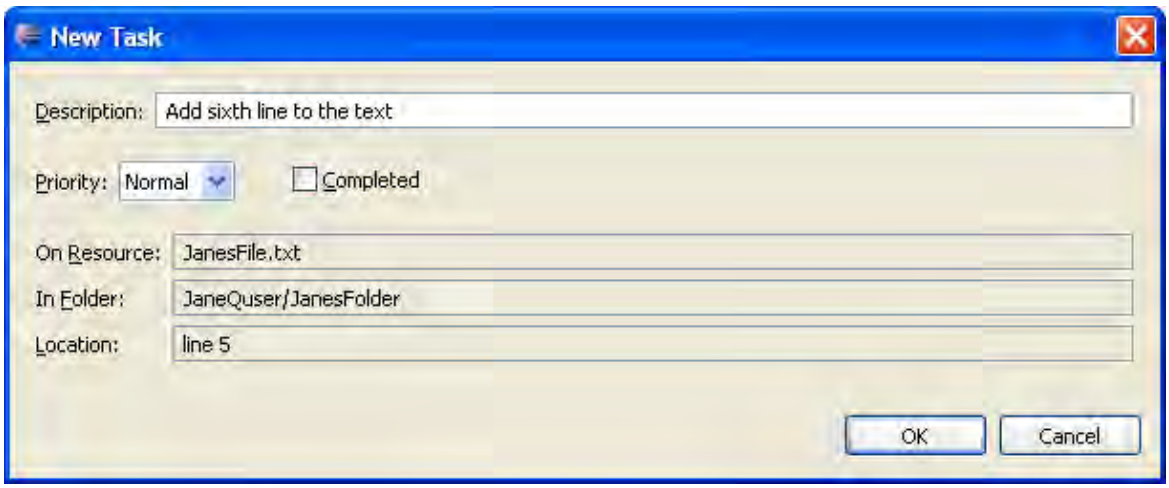
Associated tasks are associated with a specific location in a resource. To associate a task with the JanesFile.txt:

1. Open a text file (JanesFile.txt) by double clicking on it in one of the navigation views.
2. In the editor area directly to the left of any line in the text editor, access the context menu from the marker bar. The marker bar is the vertical bar to the left of the main text area.
3. From the marker bar's context menu, select **Add Task**.

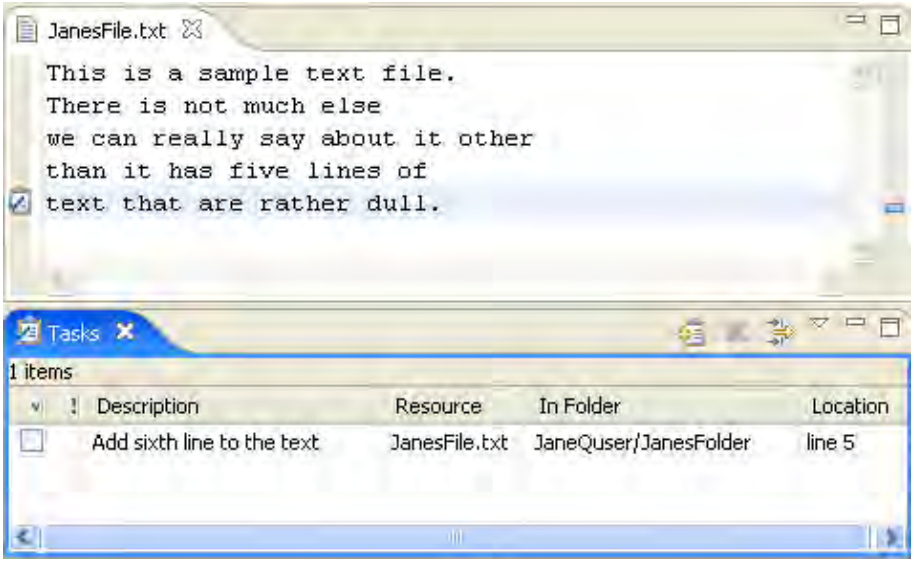
   The marker bar displays any marker including bookmarks, task markers (for associated tasks), and/or debugging breakpoints. Various markers can be associated with specific lines in a file by accessing the context menu from the marker bar directly to the left of that line.



4. In the **Description** field, type a brief description for the task that will be associated with that line in the text file.



5. Click **OK** when done.
6. Notice that a new task marker appears in the marker bar, directly to the left of the line where the task was added. Also, notice that the new task appears in the Tasks view.



7. After the new task has been added, click in the editor on the first line or any other line above the line with which the new task is associated.
8. Add several lines of text to the file at this point.
9. Notice that as lines of text are added above it, the task marker moves down in the marker bar in order to remain with the associated line in the file. The line number in the Tasks view is updated when the file is saved.
10. In the Tasks view, access the context menu for the task that was just created.
11. Select **Mark Completed**.
12. Now select **Delete Completed Tasks** from the marker's context menu.
13. Notice that the task marker disappears from the marker bar and the task is removed from the Tasks view.

Opening files

The Tasks view provides two approaches for opening the file associated with a task:

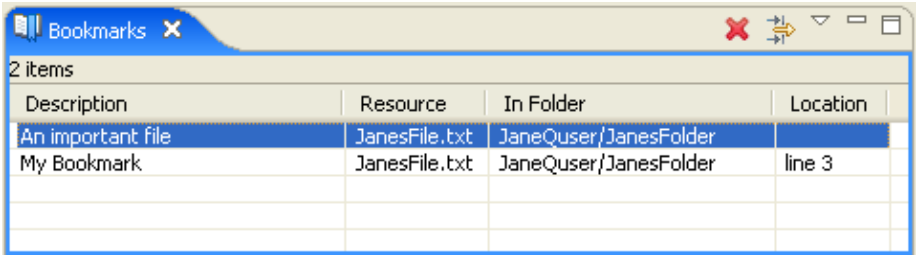
- Select the task, and from the context menu choose **Go To**
- Double click on the task

In both cases the file's editor is opened and the line with which the selected task is associated is highlighted.

Bookmarks

Bookmarks are a simple way to navigate to resources that are used frequently. This section will look at setting and removing bookmarks and viewing them in the Bookmarks view.

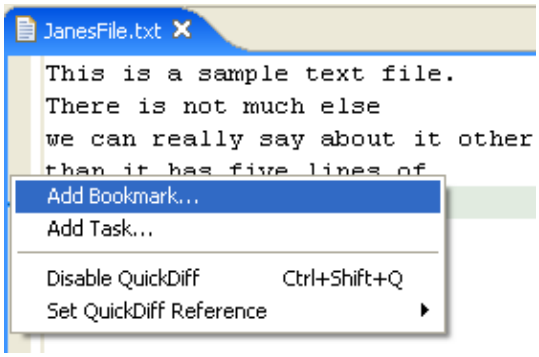
The Bookmarks view displays all bookmarks in the Workbench. To show the Bookmarks view select **Window > Show View > Bookmarks** while in the Resource perspective.



Adding and viewing bookmarks

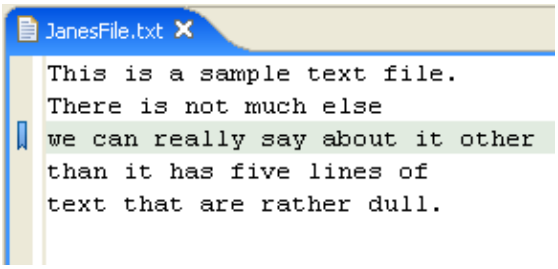
The Workbench allows the bookmarking of individual files or locations within a file. This section will demonstrate how to set several bookmarks and how to view them using the Bookmarks view.

1. From the menu bar, select **Window > Show View > Bookmarks**. The Bookmarks view appears in the Workbench.
2. Edit the file JanesFile.txt.
3. Position the cursor over the editor's marker bar next to any line in the file. Then, from the context menu on the marker bar, select **Add Bookmark**.

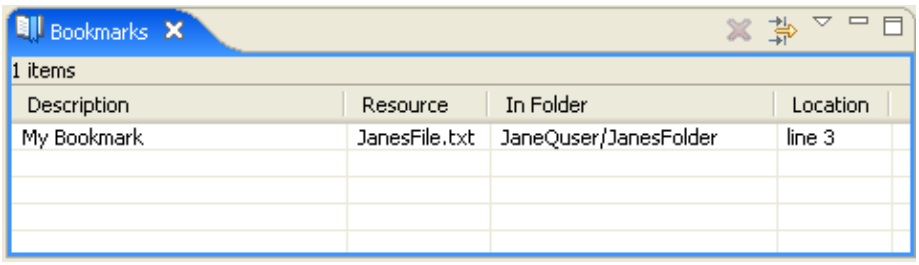


When the Add Bookmark dialog opens type in a description for this bookmark. Type in "My Bookmark".

4. Notice that a new bookmark appears in the marker bar.

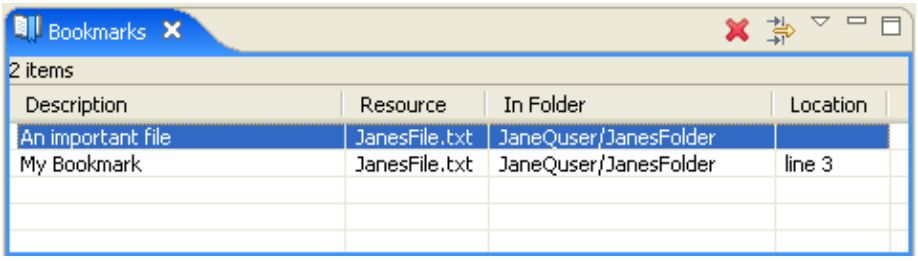


The new bookmark also appears in the Bookmarks view.



5. In one of the navigation views select the file JanesText.txt. From the main Workbench menu select ***Edit*** > **Add Bookmark**.

   This will bookmark the file using the filename to describe the bookmark. Observe the Bookmarks view now contains two bookmarks.



Using bookmarks

Now that some bookmarks have been created, instructions will be given on how to get to the files associated with the bookmarks.

1. Close all of the files in the editor area.
2. In the Bookmarks view, double−click on the first bookmark that was created (My Bookmark).
3. Notice that a file editor opens displaying the file with which the bookmark is associated and that the line associated with the bookmark is highlighted.
   Note: The Bookmarks view supports an additional way to open the file associated with a selected bookmark, simply select **Go To** from the bookmark's context menu.

In the Bookmarks view, select the associated file in the Navigator.

1. In the Bookmarks view, select My Bookmark.
2. From the bookmark's context menu choose **Show in Navigator**.
3. Notice that the Navigator view is now visible and the file JanesFile.txt is automatically selected. JanesFile.txt is the file My Bookmark was associated with.

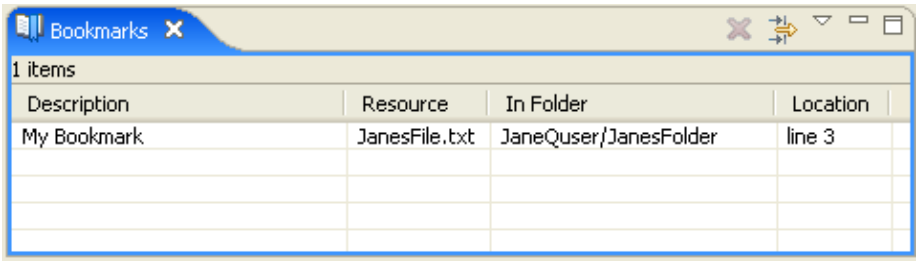
Removing bookmarks

This section will demonstrate how to remove the bookmarks that have been created.

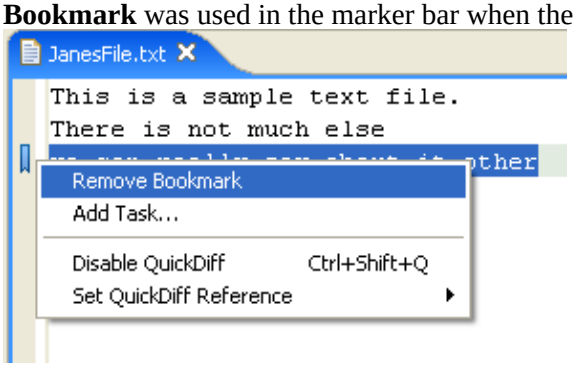
1. In the Bookmarks view, select JanesFile.txt (the second bookmark we created) and do one of the following:

   - Click the Delete button on the view's toolbar.
   - From the bookmark's context menu choose delete.
   - Hit the Delete key on the keyboard.

Notice that the bookmark is removed from the Bookmarks view.



2. There should be one remaining bookmark. This bookmark is associated with a line in the file JanesFile.txt. There are two other approaches to removing this bookmark.
   - Use **Remove Bookmark** in the marker bar of the JanesFile.txt editor. Remember, **Add Bookmark** was used in the marker bar when the bookmark was first created.



   - Delete the bookmark (as was done above) by using **Delete** from the bookmark's popup menu in the Bookmarks view.

   Here is the second approach.
3. Ensure there is an editor open on JanesFile.txt.

   Although the editor doesn't actually need to be open, the editor update will be viewable as the bookmark is deleted.
4. In the Bookmarks view, select JanesFile.txt (the remaining bookmark). Click the Delete button on the view's toolbar. Notice that the bookmark is removed from the Bookmarks view and the JanesFile.txt editor.

Rearranging views and editors

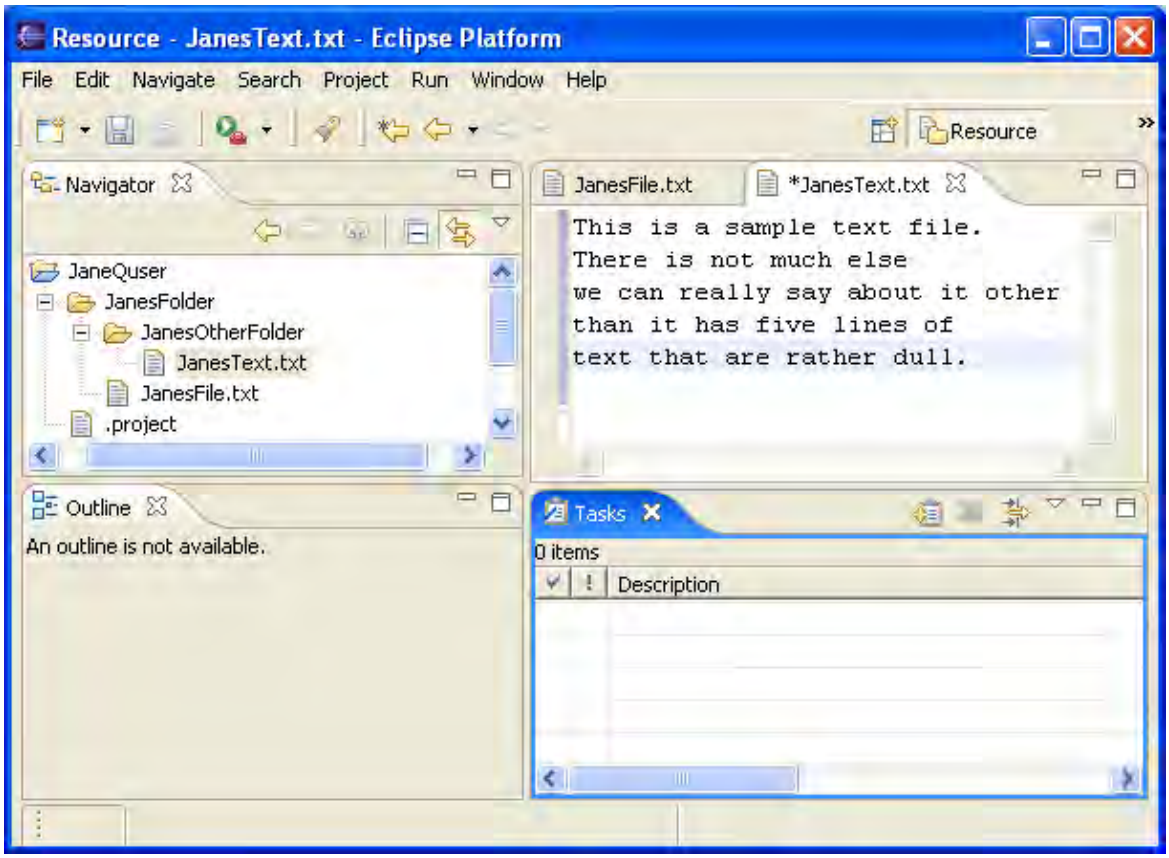
This section will explain how to rearrange editors and views to customize the layout of the Workbench.

Setup

Before rearranging the Workbench, a little housekeeping is required.

1. Start by choosing **Window > Reset Perspective** and selecting **OK**. This will reset the current perspective to its original views and layout.
2. Ensure there are editors open for JanesFile.txt and JanesText.txt. Close any other editors.

The Workbench should now look like this:



Drop cursors

Drop cursors indicate where it is possible to dock views in the Workbench window. Several different drop cursors may be displayed when rearranging views.

	Dock above: If the mouse button is released when a dock above cursor is displayed, the view will appear above the view underneath the cursor.
	Dock below: If the mouse button is released when a dock below cursor is displayed, the view will appear below the view underneath the cursor.
	Dock to the right: If the mouse button is released when a dock to the right cursor is displayed, the view will appear to the right of the view underneath the cursor.
	Dock to the left: If the mouse button is released when a dock to the left cursor is displayed, the view will appear to the left of the view underneath the cursor.
	Stack: If the mouse button is released when a stack cursor is displayed, the view will appear as a tab in the same pane as the view underneath the cursor.
	Restricted: If the mouse button is released when a restricted cursor is displayed, the view will not dock there. For example, a view cannot be docked in the editor area.

Rearranging views

The position of the Navigator view in the Workbench window can be changed.

1. Click in the title bar of the Navigator view and drag the view across the Workbench window. Do not release the mouse button yet.

2. While still dragging the view around on top of the Workbench window, note that various drop cursors appear. These drop cursors (see previous section) indicate where the view will dock in relation to the view or editor area underneath the cursor when the mouse button is released. Notice also that a rectangular highlight is drawn that provides additional feedback on where the view will dock.
3. Dock the view in any position in the Workbench window, and view the results of this action.
4. Click and drag the view's title bar to re−dock the view in another position in the Workbench window. Observe the results of this action.
5. Finally, drag the Navigator view over the Outline view. A stack cursor will be displayed. If the mouse button is released the Navigator will be stacked with the Outline view into a tabbed notebook.

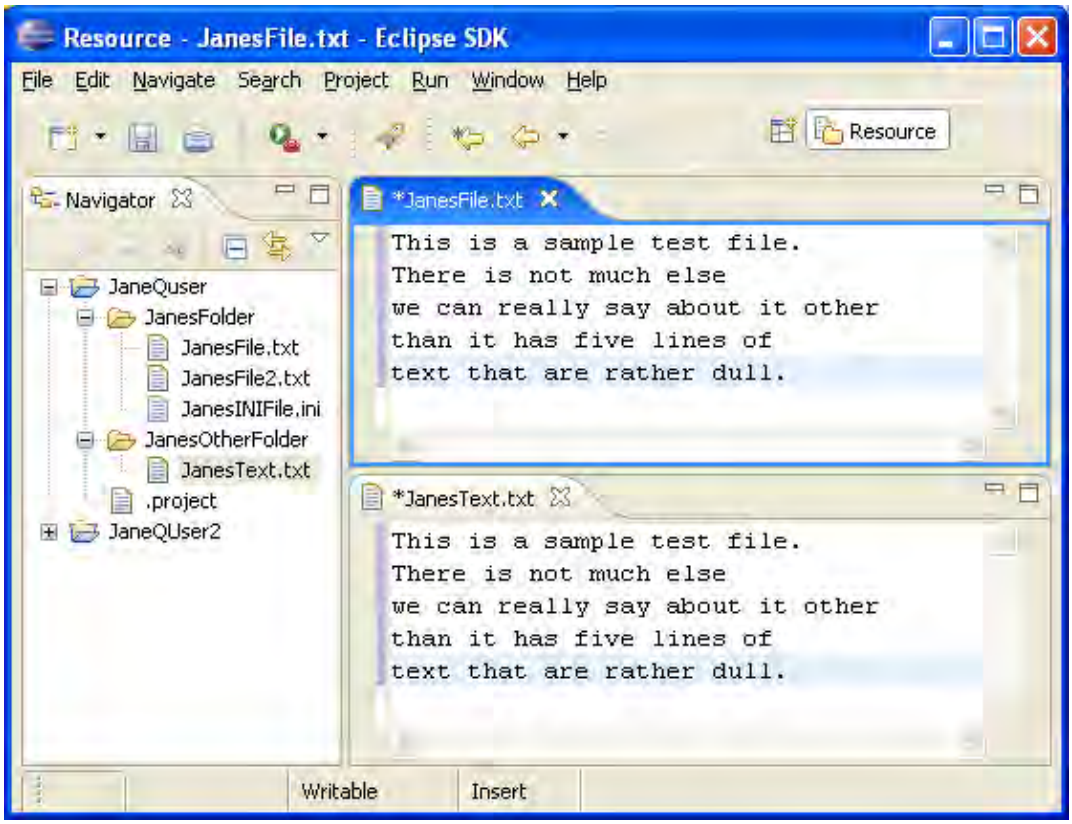
Tiling editors

The Workbench allows for the creation of two or more sets of editors in the editor area. The editor area can also be resized but views cannot be dragged into the editor area.

1. Open at least two editors in the editor area by double−clicking editable files in one of the navigation views.
2. Click and drag one of the editor's tabs out of the editor area. Do not release the mouse button.
3. Notice that the restricted cursor displays if an attempt is made to drop the editor either on top of any view or outside the Workbench window.
4. Still holding down the mouse button, drag the editor over the editor area and move the cursor along all four edges as well as in the middle of the editor area, on top of another open editor. Notice that along the edges of the editor area the directional arrow drop cursors appear, and in the middle of the editor area the stack drop cursor appears.
5. Dock the editor on a directional arrow drop cursor so that two editors appear in the editor area.
6. Notice that each editor can also be resized as well as the entire editor area to accommodate the editors and views as necessary.
7. It is important to observe the color of an editor tab (in the figure below there are two groups, one above the other)

   blue − indicates that the editor is currently active

   default (gray on Windows XP) − indicates that the editor was the last active editor. If there is an active view, it will be the editor that the active view is currently working with. This is important when working with views like the Outline and Properties that work closely with the editor.
8. Drag and dock the editor somewhere else in the editor area, noting the behavior that results from docking on each kind of drop cursor. Continue to experiment with docking and resizing editors and views until the Workbench has been arranged to satisfaction. The figure below illustrates the layout if one editor is dragged and dropped below another.



Rearranging tabbed views

In addition to dragging and dropping views on the Workbench the order of views can also be rearranged within a tabbed notebook.

1. Choose **Window > Reset Perspective** to reset the Resource perspective back to its original layout.
2. Click on the Outline title bar and drag it on top of one of the navigation views. The Outline will now be stacked on top of one of the navigation views.
3. Click the tab of the navigation views and drag it to the right of the Outline tab.

4. Once the cursor is to the right of the Outline tab and the cursor is a stack cursor release the mouse button.

   Observe the Navigator tab is now to the right of the Outline tab.

Maximizing

Sometimes it is useful to be able to maximize a view or editor. Maximizing both views and editors is easy.

- To maximize a view, either double click on its tab or choose **Maximize** from the tab's popup menu.
- To maximize an editor double click on the editor tab or choose **Maximize** from the tab's popup menu.

Restoring a view to its original size is done in a similar manner (double click or choose **Restore** from the menu).

Fast views

Fast views are hidden views, which can be quickly made visible. They work the same as normal views, only when they are hidden they do not take up screen space on the Workbench window.

This section we will explain how to convert the Navigator view into a fast view.

Creating fast views

Fast views are hidden views, which can be quickly made visible. These instructions commence by creating a fast view from one of the navigation views and then explain how to use the view once it is a fast view.

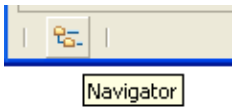
There are two ways to create a fast view

- Using drag and drop.
- Using a menu operation available from the view System menu.

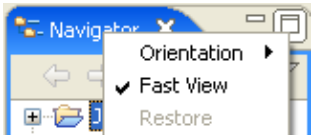
Create a fast view using drag and drop as follows.

1. In the Navigator view click on the title bar and drag it to the shortcut bar at the bottom left of the window.
2. Once over the shortcut bar, the cursor will change to a "fast view" cursor. Release the mouse button to drop the Navigator onto the shortcut bar.

   The shortcut bar now includes a button for the Navigator fast view



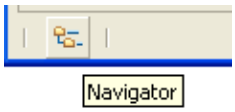
To create a fast view using the second approach, start by popping up the context menu over the Navigator view's tab. From this menu, select **Fast View**.



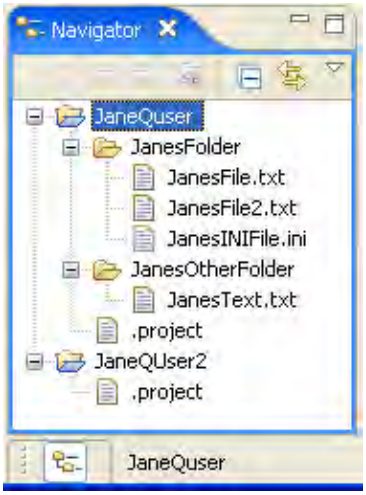
Working with fast views

The Navigator has now been converted into a fast view. This section will demonstrate what can now be done with it.

Confirm that the shortcut bar at the bottom left of the window still has the Navigator view and looks like this:



1. In the shortcut bar click on the Navigator fast view button.
2. Observe the Navigator view slides out from the left side of the window.



3. The Navigator fast view can be used as it would be normally. To resize a fast view, move the mouse to the right edge of the fast view where the cursor will change to a double−headed arrow. Then hold the left mouse button down as the mouse is moved.
4. To hide the fast view simply click on another view or editor or click on the **Minimize** button on the fast view's toolbar

Note: If a file is opened from the a navigation view fast view, the fast view will automatically hide itself to allow the file to be worked with.

To convert a fast view back to a regular view either:

- Choose **Fast View** from the context menu of the icon in the top left corner of the view.
- Drag the fast view icon from the tool bar, and drop it somewhere in the Workbench window.

Perspectives

A perspective defines the initial set and layout of views in the Workbench window. One or more perspectives can exist in a single Workbench window.

Perspectives can be opened in one of two ways:

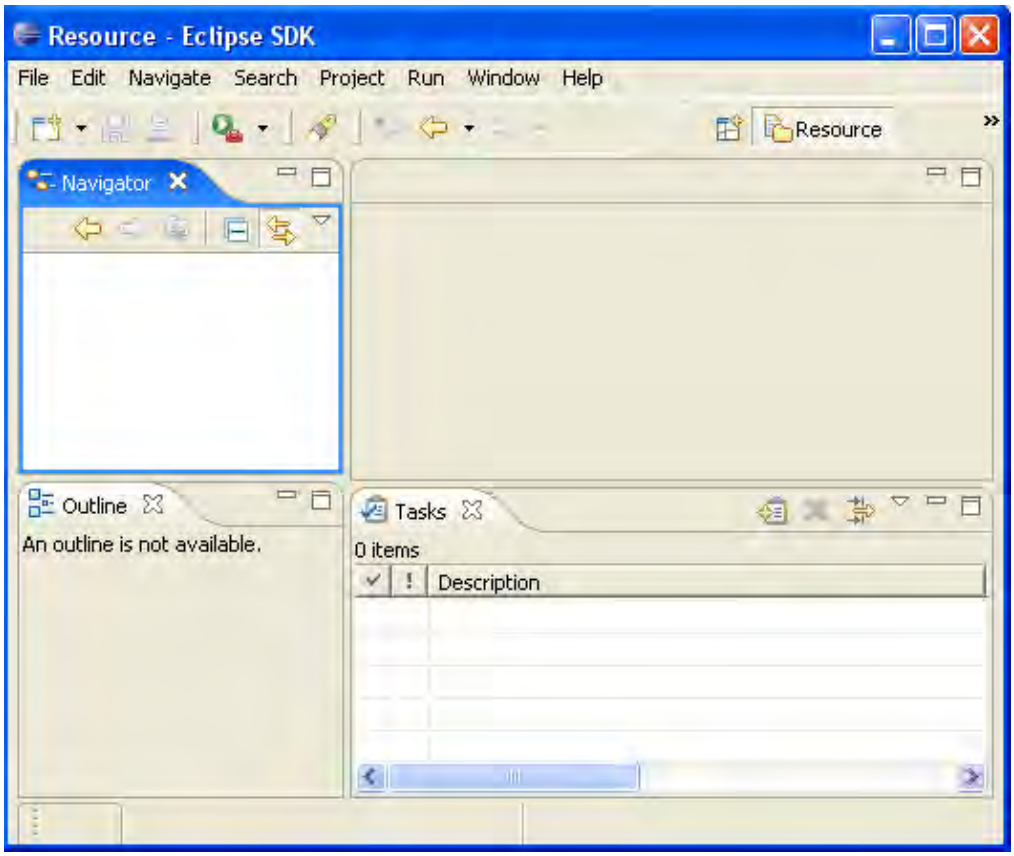
- In the same (existing) Workbench window.
- In a new Workbench window.

Perspectives define visible action sets, which can be changed to customize a perspective. A perspective that is built in this manner can be saved, creating a custom perspective that can be opened again later.

The Workbench window displays one or more perspectives. Each product determines initially what default perspective is displayed, in this example it is the Resource perspective. A perspective consists of views such as the Navigator as well as editors for working with resources. More than one Workbench window can be open at any given time.

So far, only the Resource perspective (shown below) has been used in this tutorial. This section will explore how to open and work with other perspectives.

A perspective provides a set of functionality aimed at accomplishing a specific type of task, or working with a specific type of resource.



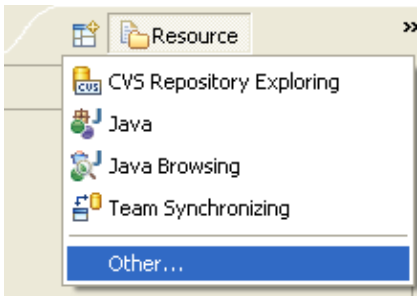
New perspectives

There are several ways to open a new perspective within this Workbench window:

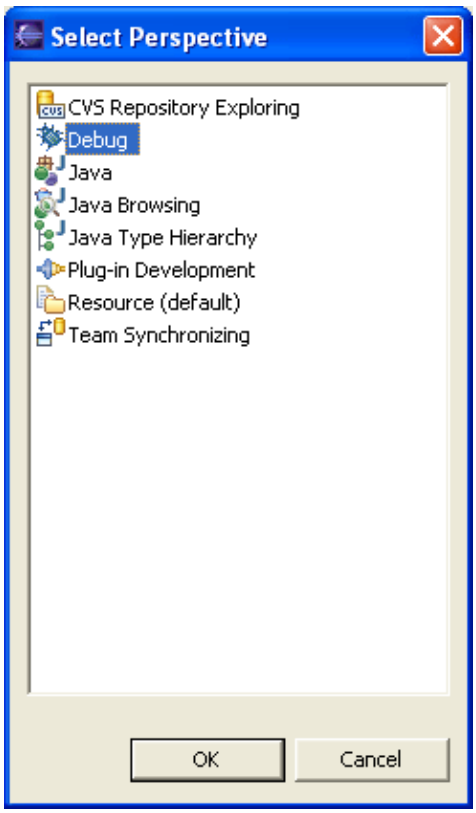
- Using the Open Perspective button  on the shortcut bar.
- Choosing a perspective from the **Window > Open Perspective** menu.

To open one by using the shortcut bar button:

1. Click on the Open Perspective button  .
2. A menu appears showing the same choices as shown on the **Window > Open Perspective** menu. Choose **Other** from the menu.



3. In the Select Perspective dialog choose **Debug** and click **OK**.



The Debug perspective is displayed.

4. There are several other interesting things to take note of.

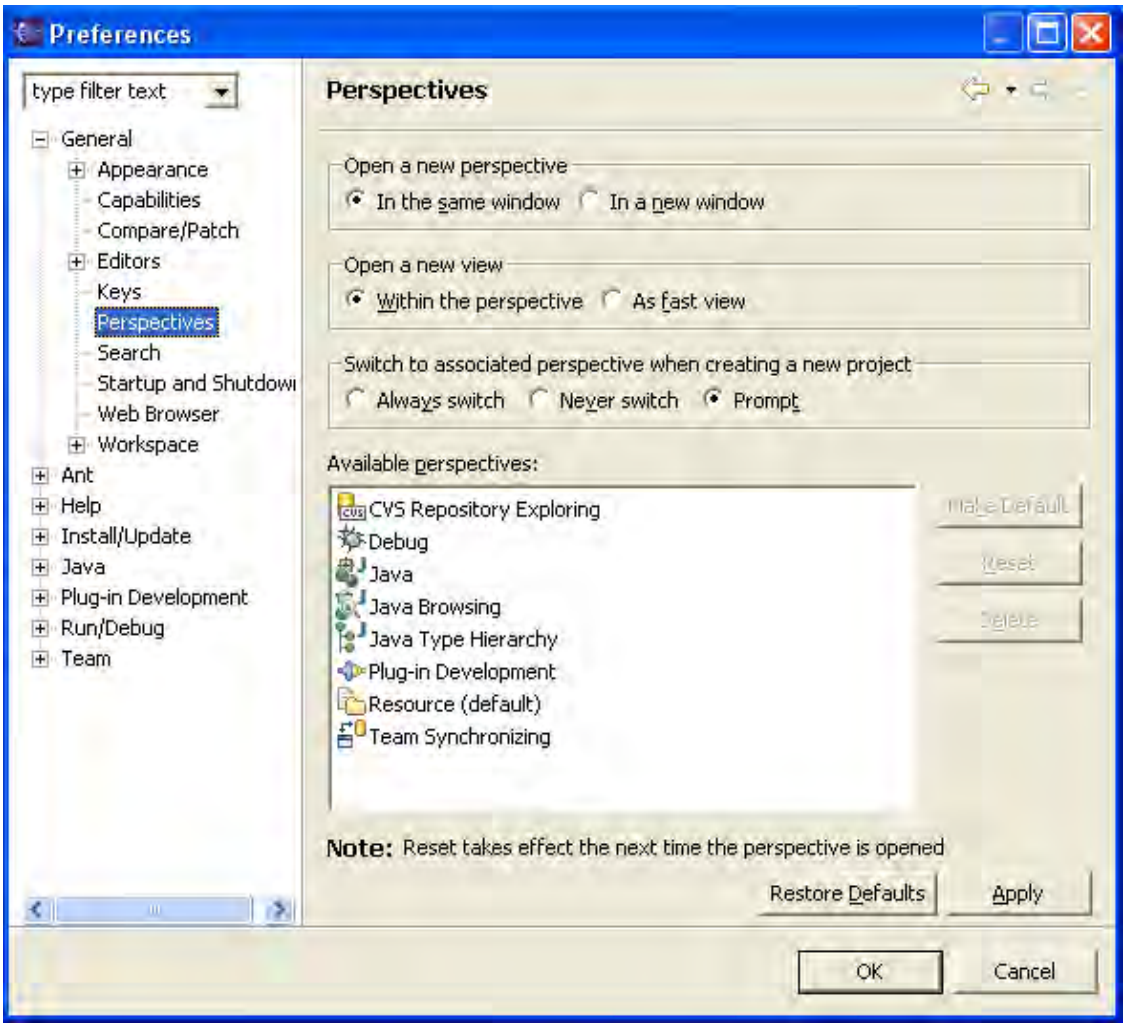
- The title of the window now indicates that the Debug perspective is in use.
- The shortcut bar contains several perspectives, the original Resource perspective, the new Debug perspective and a few others. The Debug perspective button is pressed in, indicating that it is the current perspective.
- To display the full name of the perspective right click the perspective bar and check ***Show Text***.

5. In the shortcut bar, click on the Resource perspective button. The Resource perspective is once again the current perspective. Notice that the set of views is different for each of the perspectives.

New windows

As well as opening a perspective inside the current Workbench window, new perspectives can also be opened in their own window.

By default, new perspectives are opened in the current window. This default behavior can be configured using **Window > Preferences > General > Perspectives**.

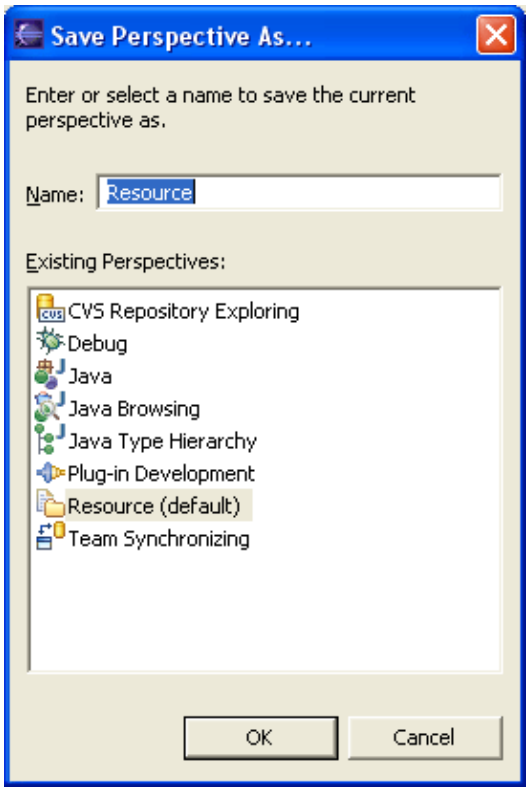


Saving perspectives

This tutorial has demonstrated how to add new views to the perspective, rearrange the views and convert views into fast views. The Workbench also allows this layout to be saved for future use.

1. In the shortcut bar click on the Resource perspective. The Resource perspective is now active.
2. Drag the Outline view and stack it with one of the navigation views.
3. Choose **Window > Save Perspective As...**
4. The Save Perspective As dialog allows for an existing perspective to be redefined or for a new perspective to be created.

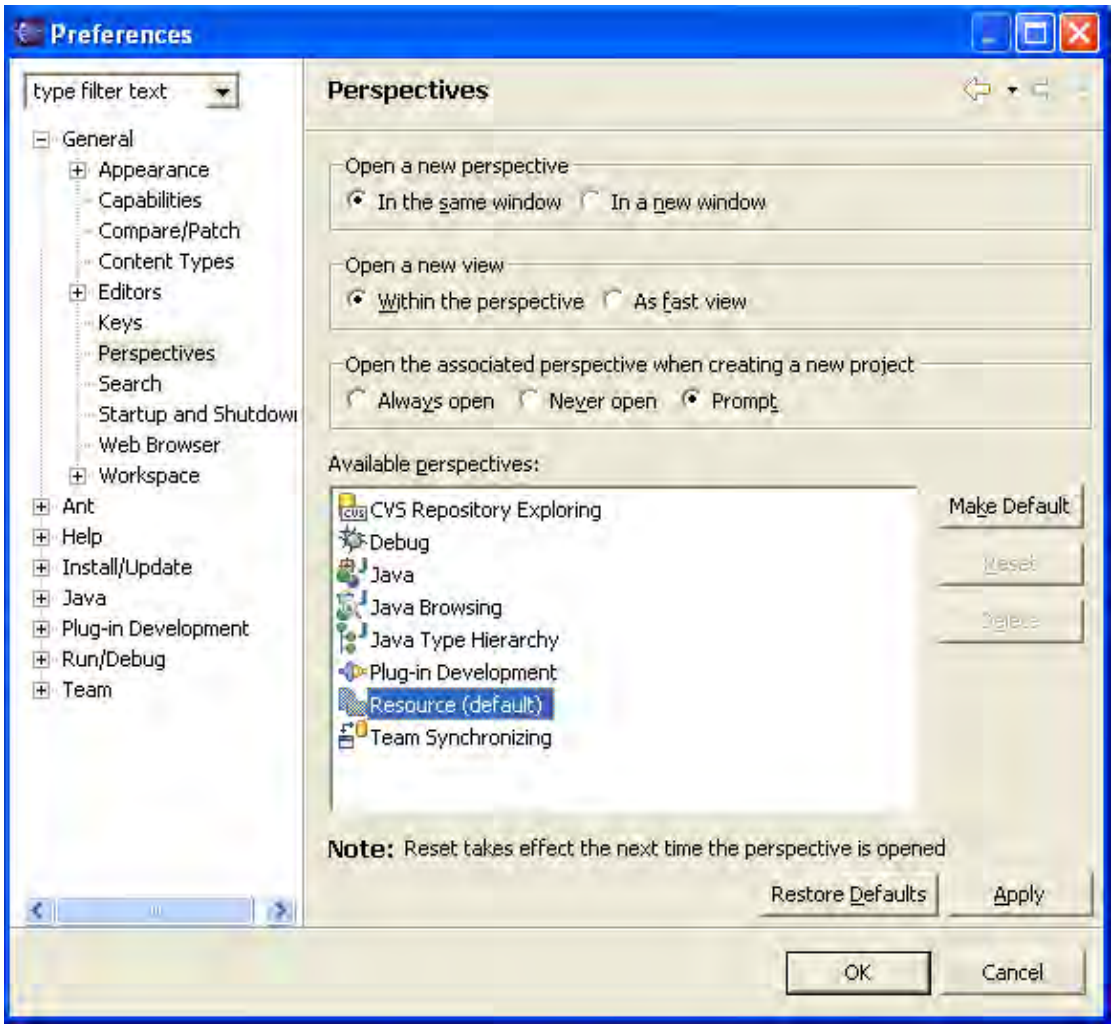
   Click **OK** to update the Resource perspective and **Yes** to the subsequent confirmation dialog. The new perspective layout will be used if the perspective is reset or if a new one is opened.



5. In the Resource perspective move the Outline view so that it is now stacked with the Tasks view.
6. Choose **Window > Reset Perspective**. Notice the Outline view is stacked with the navigation view. Originally when the Workbench was first started it was below the navigation view, but because the perspective was saved with the navigation view and Outline stacked, it now considers this its initial layout.
7. Choose **Window > New Window** to open a second window showing the resource perspective. Observe that it uses the newly saved layout.
8. Close the second window.

While the Resource perspective has been changed, there is a way to get back the original layout. To reset the Resource perspective to its original layout:

1. Choose **Window > Preferences**.
2. Expand **General** and select **Perspectives**.
3. Select **Resource (default) > Make Default** and then click **OK**.



4. Any changes to the saved state of the perspective has now been undone. To update the current copy of the Resource perspective that is being worked with, also choose **Window > Reset Perspective** from the Workbench's menu bar.

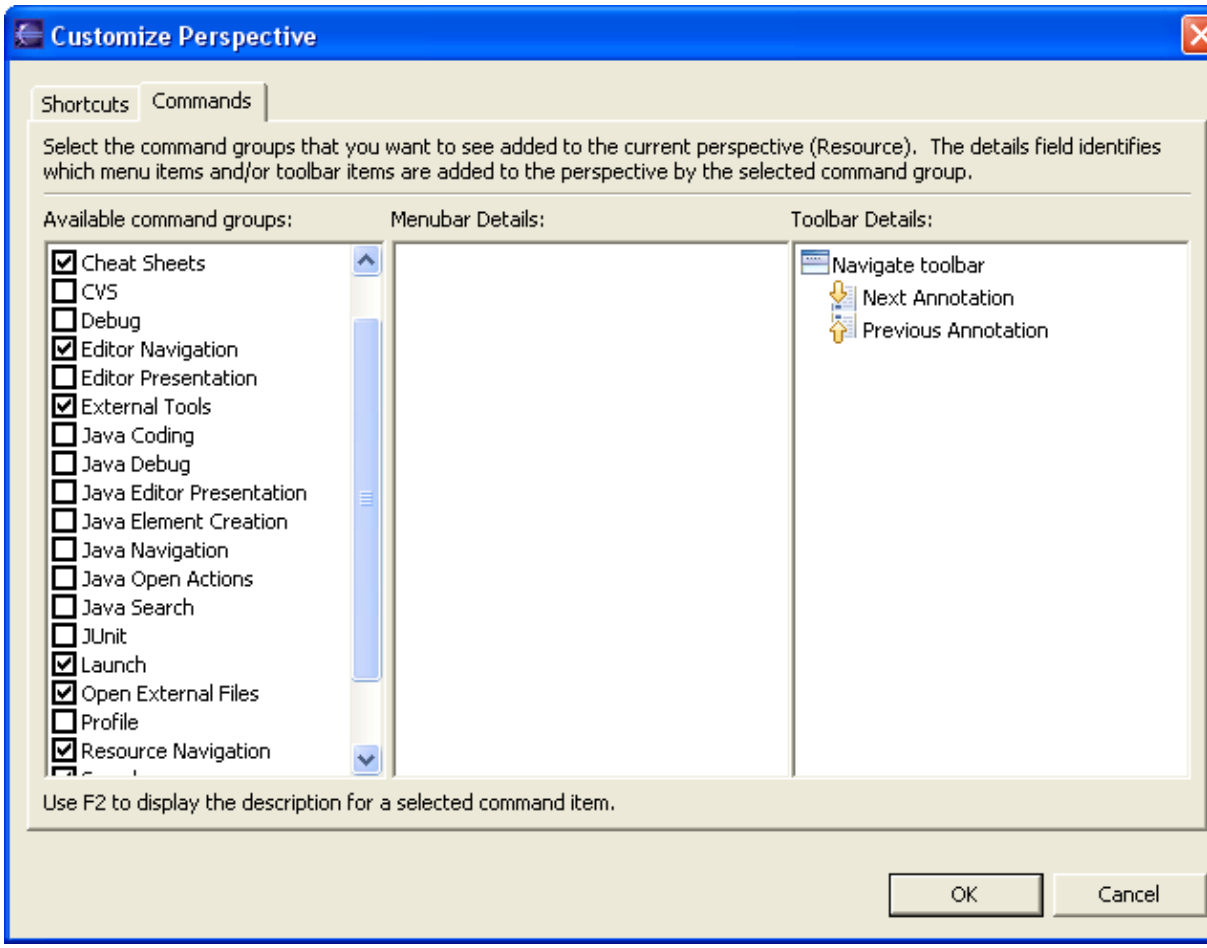
Configuring perspectives

In addition to configuring the layout of a perspective you can also control several other key aspects of a perspective. These include:

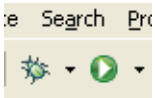
- The **New** menu.
- The **Window > Open Perspective** menu.
- The **Window > Show View** menu.
- Action sets that show up on the toolbar.

Try customizing one of these items.

1. In the shortcut bar click on the Resource perspective.
2. Select **Window > Customize Perspective...**
3. Select the ***Commands*** tab.
4. Check **Launch** and click **OK**.



5. Observe that the toolbar now includes buttons for debug/run launching.



6. After experimenting with the other options on the Customize Perspective dialog, choose **Window > Reset Perspective** to return the perspective to its original state.

Comparing

The Workbench allows for the comparison of multiple resources and for the presentation of the results in a special compare editor.

Setup

Before commencing with compare a few files must be created. This will also be a good time to recap some of the basic features that have already been introduced.

1. Start by selecting all of the projects in one of the navigation views and deleting them by using **Delete** on the pop−up menu.
2. Create a new simple project using **File > New > Project**. Be sure to give the project a distinct name by typing unique name as the name of the new project (for example, "JaneQUserCompare"). Do not use spaces or special characters in the project name.
3. Use the project's pop−up menu to create a file called file1.txt.

In the editor for file1.txt type the following lines of text and save the file:

This is line 1.
This is line 2.
This is line 3.
This is line 4.
This is line 5.

4. In one of the navigation views select file1.txt and use Ctrl+C to copy the file.
5. Use Ctrl+V (Paste) to create the copy. In the name conflict dialog which appears, rename the file to file2.txt.

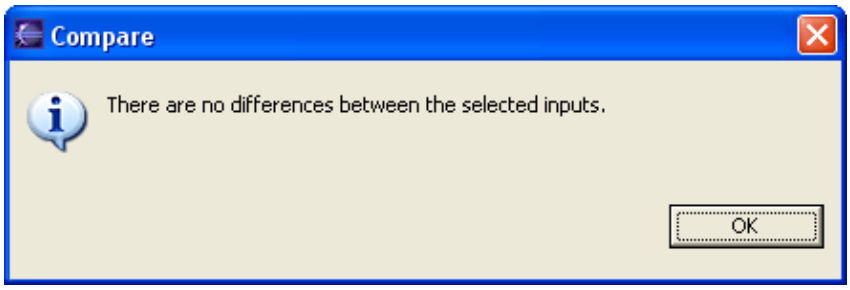
(On the Mac, use Command+C and Command+V.)

There are now two identical files, file1.txt and file2.txt.

Simple compare

In one of the navigation views select file1.txt and file2.txt choose **Compare With > Each Other** from the context menu.

A dialog will appear indicating that the two files are the same.



Edit file1.txt as follows:

- delete line 1 "*This is line 1.*"
- change line 3 to be "This is a much better line 3."
- insert a line 4a (before line 5) that reads "This is line 4a and it is new"

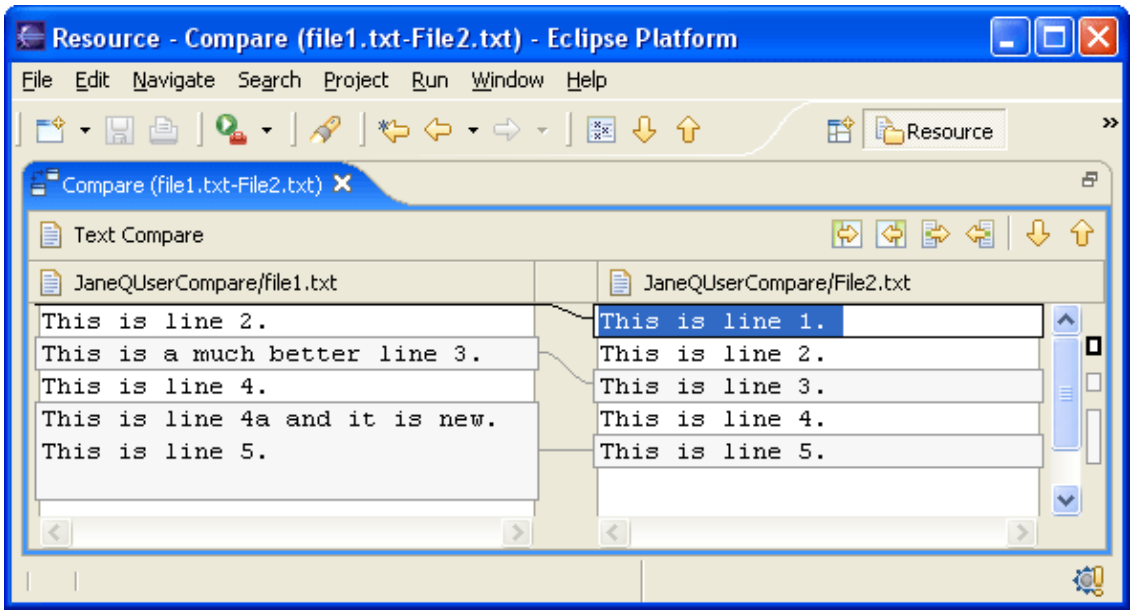
The file (file1.txt) should now read as follows:

This is line 2.
This is a much better line 3.
This is line 4.
This is line 4a and it is new
This is line 5.

Save the contents of the file by choosing **File > Save** (or pressing Ctrl+S).

To compare the files, once again select file1.txt and file2.txt and choose **Compare With > Each Other** in the navigation view context menu.

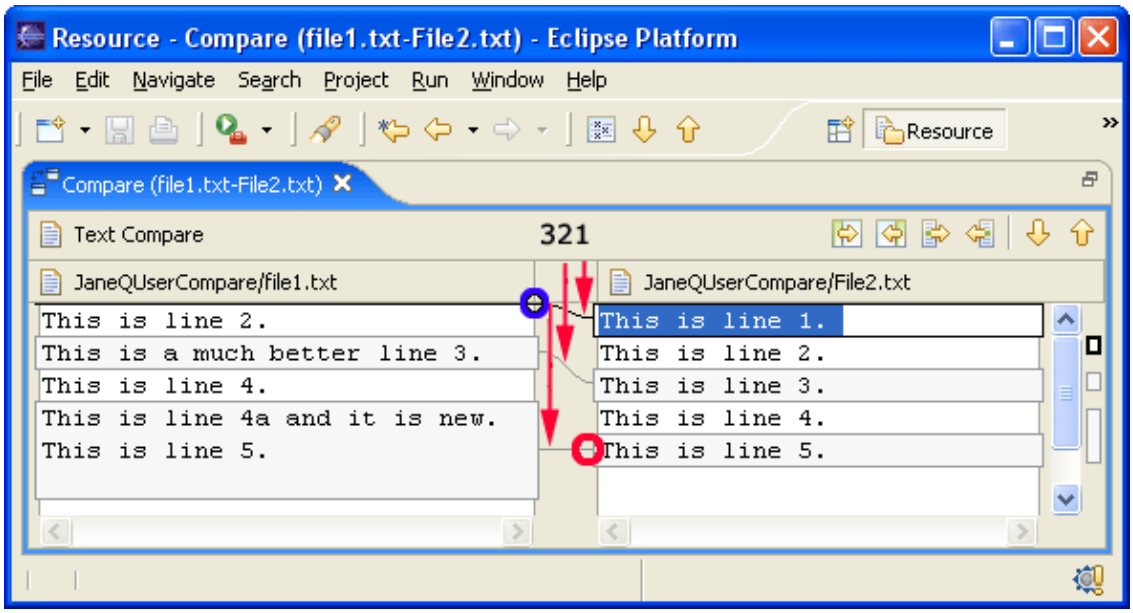
A special compare editor opens. The next section will explain how to use this compare editor.



Understanding the comparison

Comparing file1.txt and file2.txt resulted in the following compare editor. The left side shows the contents of file1.txt and the right side shows the contents of file2.txt. The lines connecting the left and right panes indicate the differences between the files.

If more room is needed to look at the comparison, the editor tab can be double clicked to maximize the editor.



The numbered changes on the left side of the difference editor are as follows:

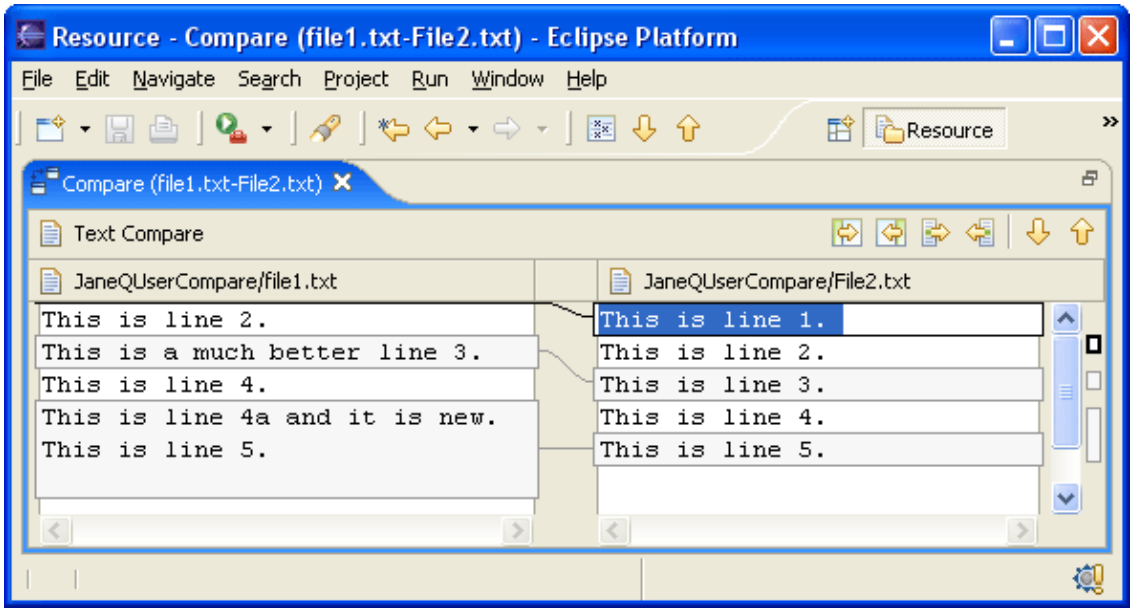
- Starting with the top line (in the left pane) the difference bar (in the area of the blue circle) indicates something is missing from the very top of the left file. Follow the difference band (see #1) to the right file. It contains "This is line 1" .
- The next line "This is line 2." is white indicating it matches the right file.

- Moving onto the next line (colored in the background color), see that the left file and right file have different contents for this line (see #2).
- The next line (This is line 4) is once again in white, so it can be skipped.
- The next line exists in the left file but since it is in the background color its difference bar can be followed to the right (see #3) and notice that the right file does not contain the line (see red circle).

Initially the compare editor might seem a bit daunting but when simply working down the left side and focusing on the items marked as gray, and those items missing from the left side, it turns out not to be as tricky as it first seems.

Working with the comparison

Comparing file1.txt and file2.txt resulted in the following compare editor. This section demonstrates how to use the compare editor to resolve the differences between the two files.



There are two parts to the compare editor's local toolbar. Move to the next or previous change using the right group of local toolbar buttons.

1. Click the Select Next Change button  . Observe how it selects the next difference.
2. Click Select Next Change button a second time to go to the next change.
3. Click the Select Previous Change button.

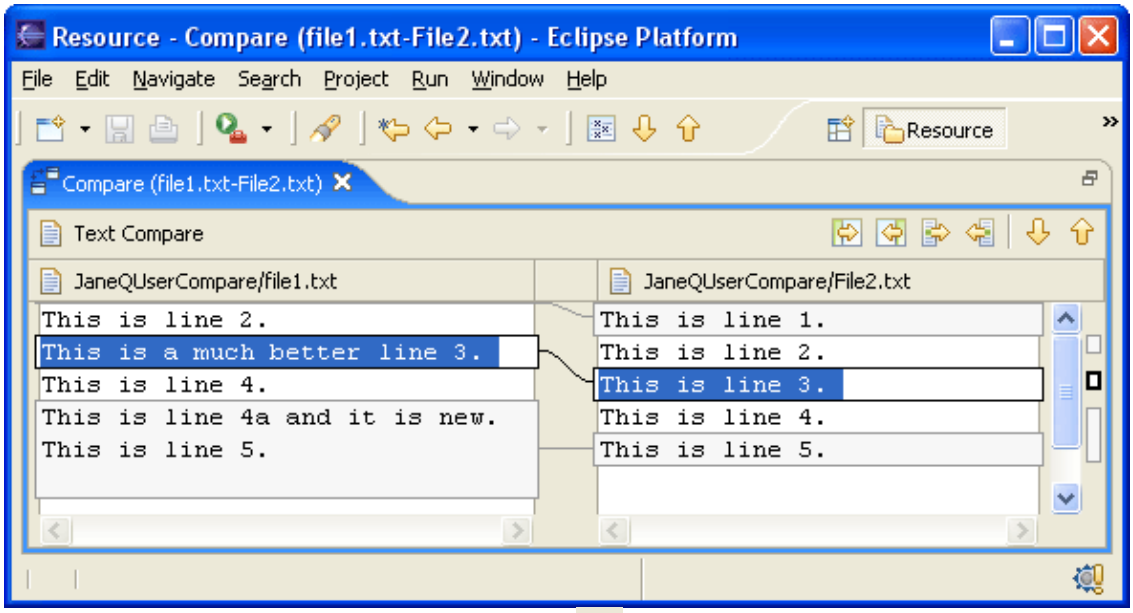
To merge changes from the left file to the right file and vice versa use the left group of local toolbar buttons. There are four types of merges that can be performed:

- Copy whole document from left to right.
- Copy whole document from right to left.
- Copy current change from left to right.
- Copy current change from right to left.

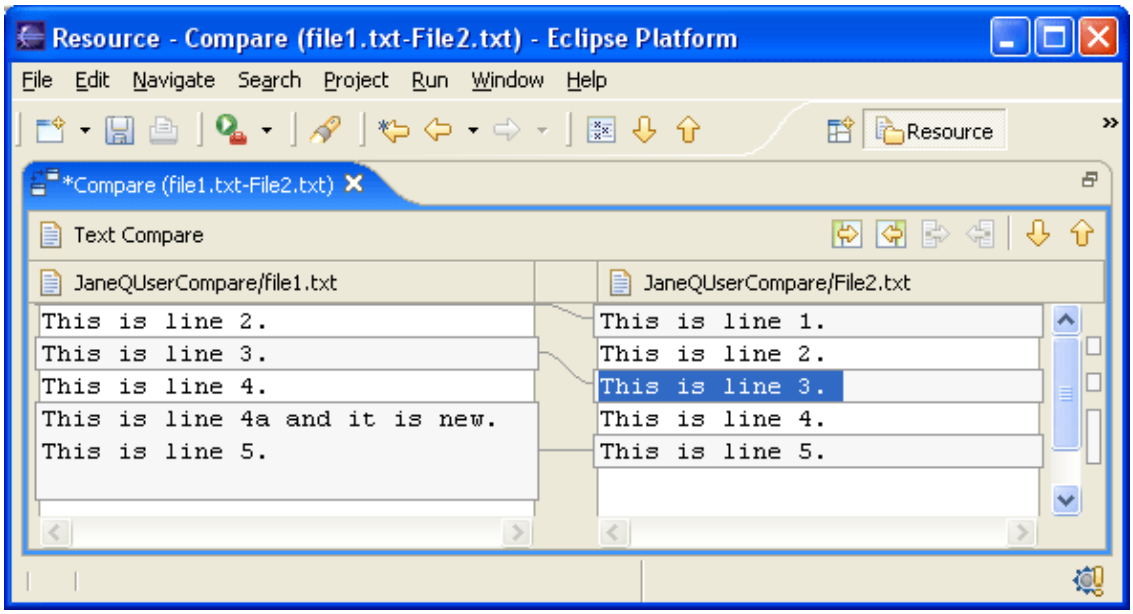
Typically the copy whole document actions are used when the entire file on either the left or right can just be replaced by the contents of the other file.

The Copy current change buttons allow a single change to be merged.

1. Ensure that the second difference is selected (as shown below):



2. Click ***Copy Current Change from Right to Left***  . Observe that the selected text from the right file is copied to the left file.

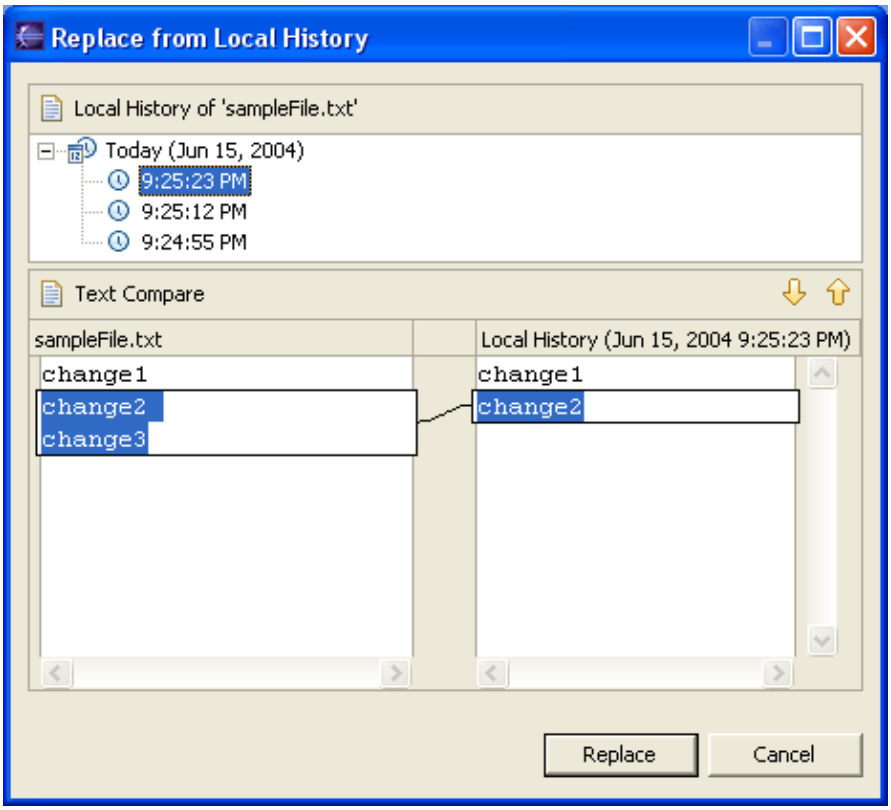


3. Close the compare editor and choose **Yes** to save the changes. Alternatively, save the changes by choosing **File** > **Save** (Ctrl+S).

Local history

Every time an editable file is saved in the Workbench, the Workbench updates the local history of that file and logs the changes that have been made. The local history of a file can then be accessed and a previously saved copy of the file can be reverted to, as long as the desired state is recent enough in the save history.

1. Create a new file named sampleFile.txt.
2. In the editor for sampleFile.txt modify the resource by adding the line "change1" and saving the file.
3. Repeat this by entering a new line "change2" and saving it again.
4. Add a third line "change3" and save it again.
5. From the resource's context menu in one of the navigation views, select **Replace With > Local History**.
6. The Replace from Local History dialog opens and shows the previous local history of the file.



The left pane of the dialog contains the Workbench's copy of the file. The figure above shows that the Workbench contains the copy with all 3 lines − the same copy that is currently shown in the editor area of the Workbench.

The first item in the local history (see above) contains the last saved copy of the file. This is the one with only two lines of text. The final entry in the tree is the first copy of the file.

The bottom area of the dialog displays the differences between the Workbench file and the specific copy of the file selected in the local history.

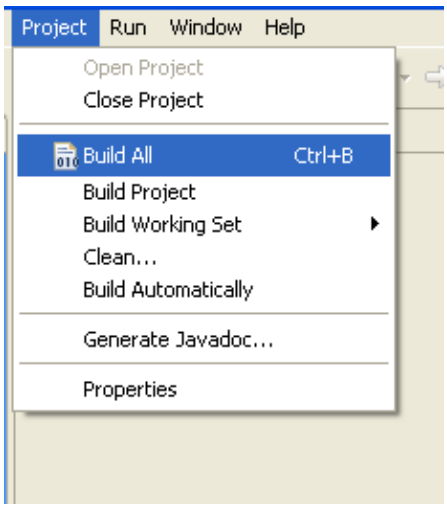
7. Select the first item (shown above) in the local history. The right pane should show one line of text.

8. Click **Replace**. This replaces the Workbench's copy of sampleFile.txt with the chosen local history item.
9. Observe that the sampleFile.txt editor now contains two lines.

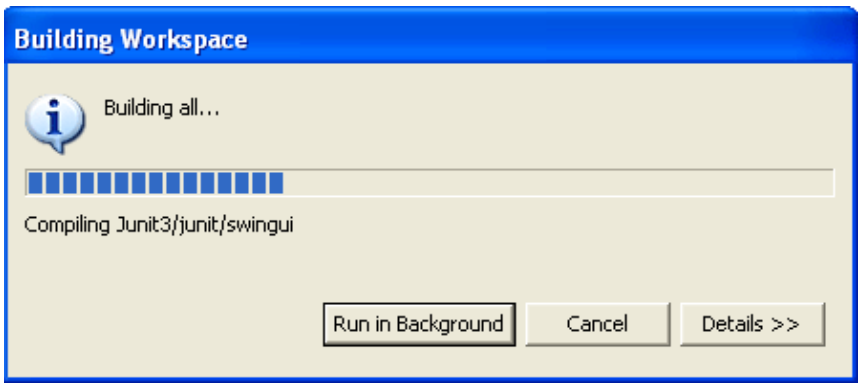
Responsive UI

By default all Eclipse operations run in the user interface thread. Employing the Responsive UI, which allows the threading of sequential code, will enable you to continue working elsewhere in Eclipse. Without the Responsive UI support you would be locked out of performing any other actions when met with a slow operation.

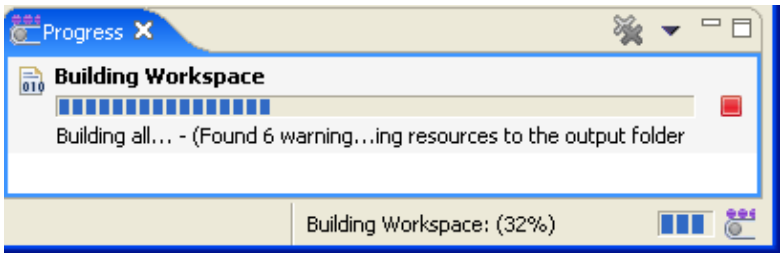
While some operations automatically run in the background (such as auto build), in many cases a dialog will be displayed providing you with the option to run an operation in the background. For example, building a project manually can sometimes take more than a few minutes, during which time you may wish to continue to use other functions in Eclipse.



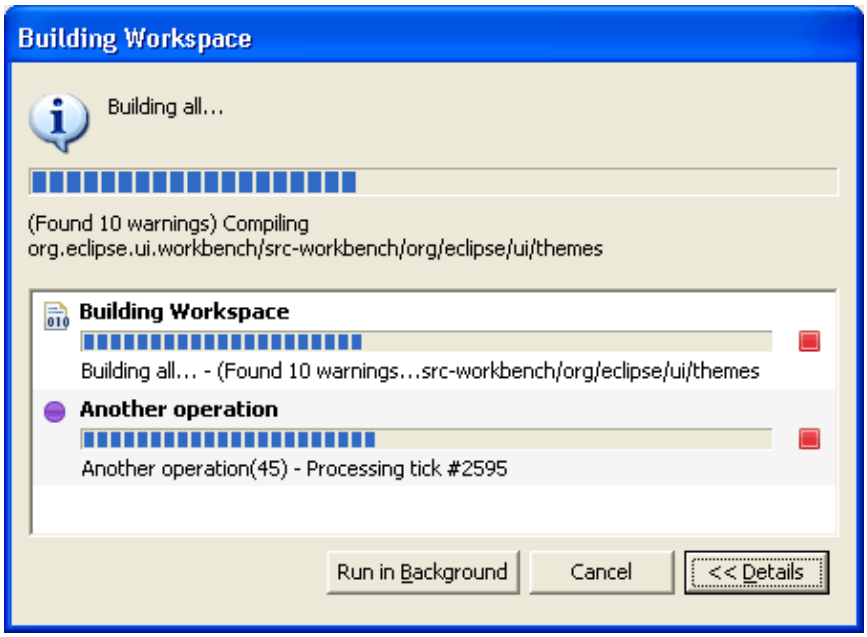
While the project is being built, select ***Run in Background*** from the ***Building Workspace*** dialog and the Responsive UI will allow you to carry on with other tasks in Eclipse.



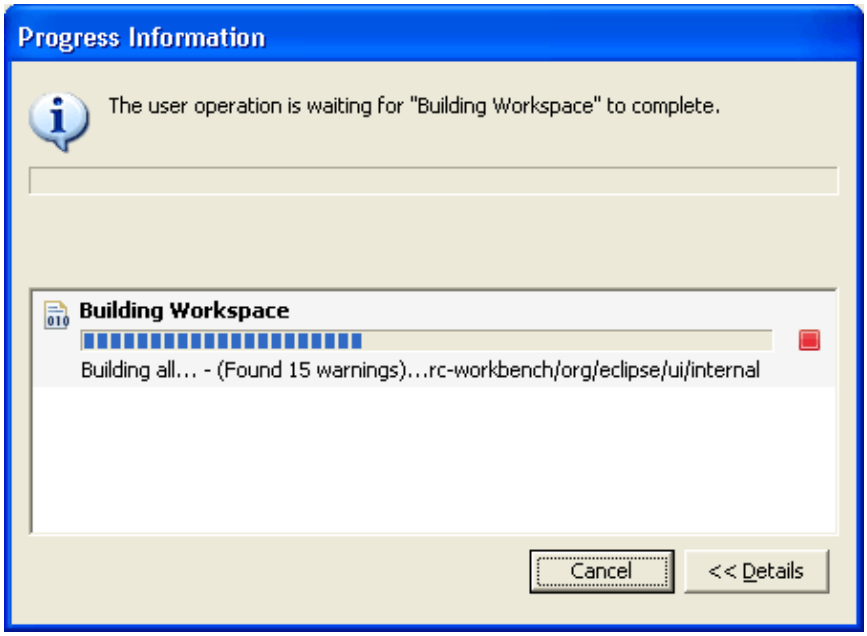
For information on the status of the action and additional operations that are currently running, click ***Details***.



The ***Details*** panel displays the status information of the operation at hand as well as any additional operations that may be running simultaneously.



The Progress Information dialog also indicates when one operation is being blocked by another.



To have operations running in the background set as the default, select ***Window*** > ***Preferences*** > ***General*** and check ***Always run in background.***

Exiting the Workbench

Each time the Workbench is exited, the Workbench is automatically saved, including all open perspectives and windows. The next time the Workbench is reopened, it will appear exactly as it was when it was last closed.

To exit the Workbench, select **File > Exit** from the menu bar or close the workbench with the window close button (x). When the latter option is used a prompt will ask if the user really wishes to exit the workbench.

Note: This dialog also presents an option to turn off the prompting. To turn it back on, select ***Window*** > ***Preferences*** from the main Workbench menu bar, in the Preferences Dialog select ***General*** > ***Startup and Shutdown*** and check Confirm exit when closing last window.

Team CVS tutorial

This chapter will explain how to use the CVS team capabilities built into the Workbench. The steps in this chapter are team oriented; it will be difficult to complete these tutorials if others are not simultaneously working through this chapter.

Instructions will be given explaining how to work on a project and then commit changes to the repository for others on the team to use. By continuing to work on the project alongside other users, this section will demonstrate how to commit resources that other users may be simultaneously working on, and how to update a workspace with changes others make in the project.

Remember, before getting started, entice at least one coworker into working through these steps as well.

Setting up a CVS repository

A repository is a persistent store that coordinates multi−user access to the resources being developed by a team. The Workbench comes with CVS support built−in. The CVS server is available at https://www.cvshome.org.

Refer to this site for information on how to install and configure a CVS repository including user access and passwords.

In order to continue with this tutorial, access to a CVS repository is required.

Starting offline

The team CVS tutorial will start by working offline and creating a simple project. Once the project is created, instructions will be given explaining how to commit it to the repository.

1. Create a new "Simple" project using **File > New > Project**. Use a unique name as the project name (e.g. JanesTeamProject).
2. Create a folder named folder1.
3. Create two text files (.txt) in folder1 called file1.txt and file2.txt. Their contents should be as follows:

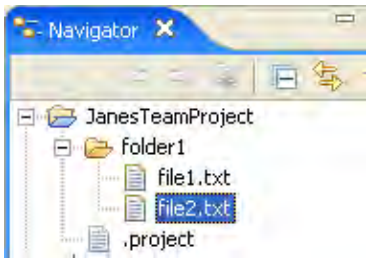
   file1.txt

   > *This is the contents*
   > *of file 1.*

   file2.txt

   > *File2 is a small file*
   > *with simple text.*

   The navigation view should now appear as follows:



The project can be continued within this mode but unless the project is committed into the repository, others on the team will be unable to work on the project as well.

Sharing the project

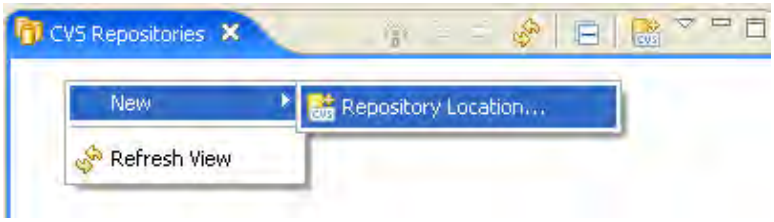
Now that a project has been created in the workspace you can make it available to other Team members. To do this the following steps will have to occur:

1. Create a CVS location identifying the team's shared CVS repository.
2. Share the project with the CVS location and commit the changes.

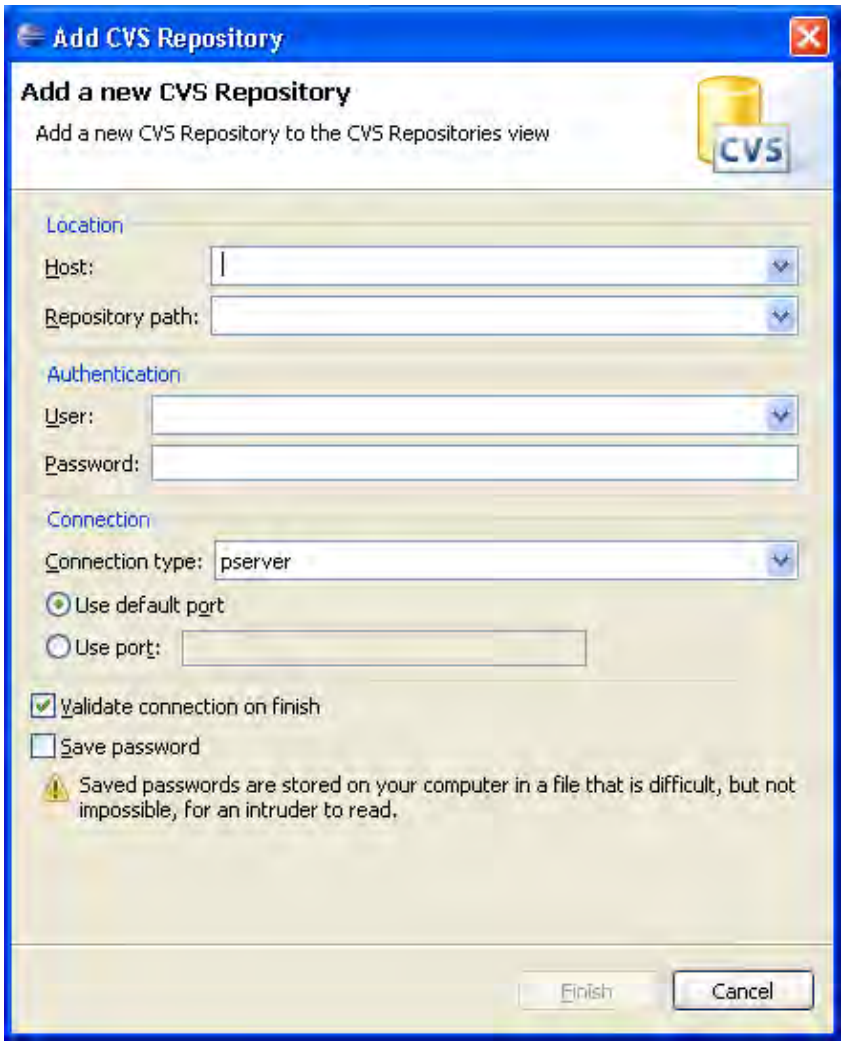
Specifying a repository location

Before it is possible to share the project with other users an available repository must first be specified.

1. Open the CVS Repository Exploring perspective. The top left view shows all of the CVS repositories that are currently being worked with. As can be seen, the view is empty, meaning a repository still needs to be specified.
2. In the context menu of the CVS Repositories view choose **New > Repository Location**.



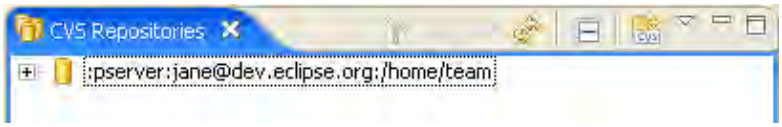
3. In the CVS Repository Location wizard, the location of the repository and the login information needs to be filled in. Assistance from a repository administrator may be required in order to fill in the necessary information.



4. In the **Host** field, type the address of the host (for example, "teamsamples.com").
5. In the **Repository path** field, type the path for the repository at the host address (for example, "/home/cvsroot/repositoryName").
6. In the **User** field, type the user name under which to connect.
7. In the **Password** field, type the password.
8. In the **Connection type** field, select the type of CVS connection for the repository (The default is pserver).
9. Leave **Use Default Port** enabled.
10. By default the **Validate Connection on Finish option** is checked.
11. Click **Finish** when done.

    Since **Validate location on finish** was checked, the wizard will now attempt to validate the information by connecting to the repository. In doing so it may prompt for a password. ***Note:*** The repository connection is only used to validate the information.

12. Observe that the CVS Repositories view now shows the new repository location.



Repository locations

Now that a new repository location has been added to the CVS Repositories view, the following will explain what a repository location is or, more importantly, what it isn't.

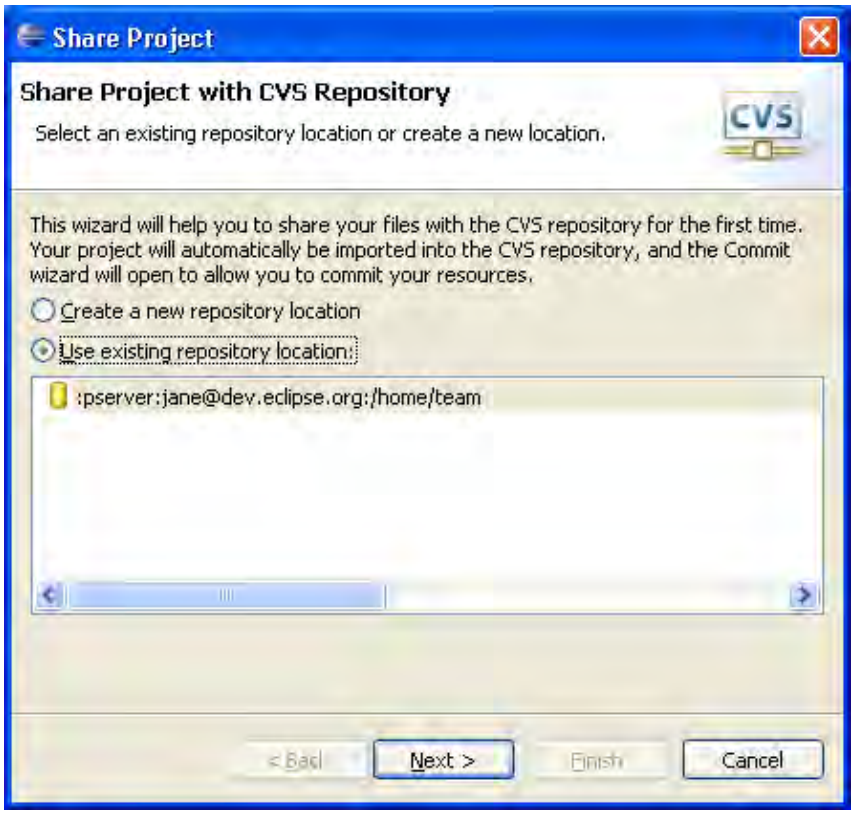
A repository location is not an actual live connection. Instead, it is a description of where the repository is located. At a later time, when instructions are given explaining how to commit work to the repository or update with work done by others, the Workbench will create a connection based on this location information. Connections will be opened and closed as required when performing CVS team operations, and those connections will be based on the information provided in the repository location.

When disconnected from the network the CVS Repositories view continues to display the list of known repository locations. In addition, the projects themselves will still know the repository location they are associated with.

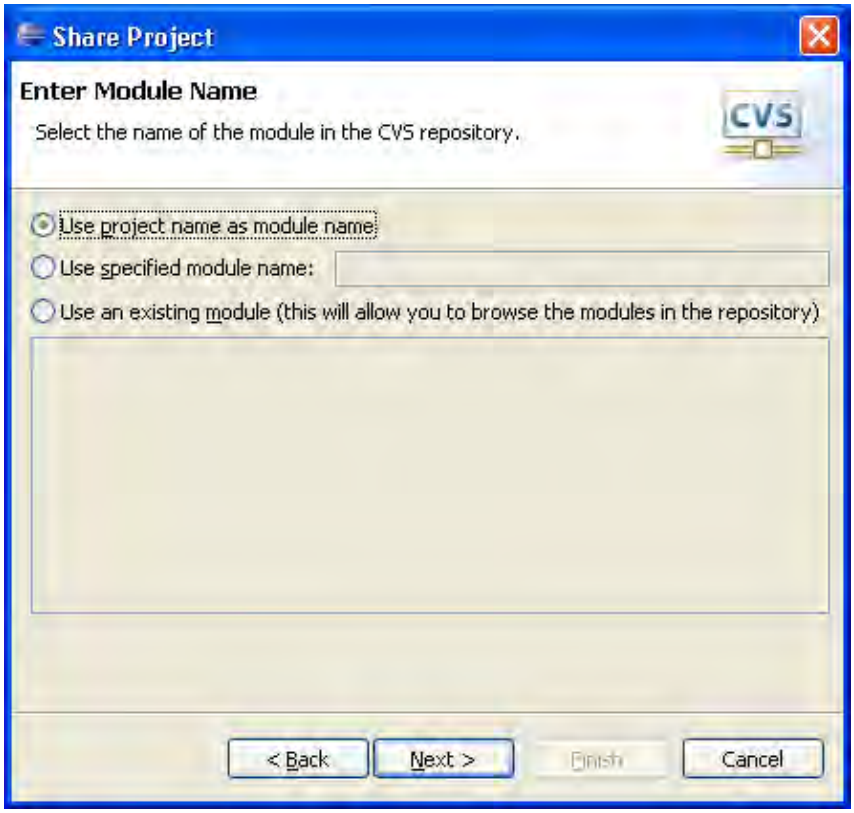
Sharing a project

Now that a project has been created and a repository location has been specified, the project can be made available to other team members.

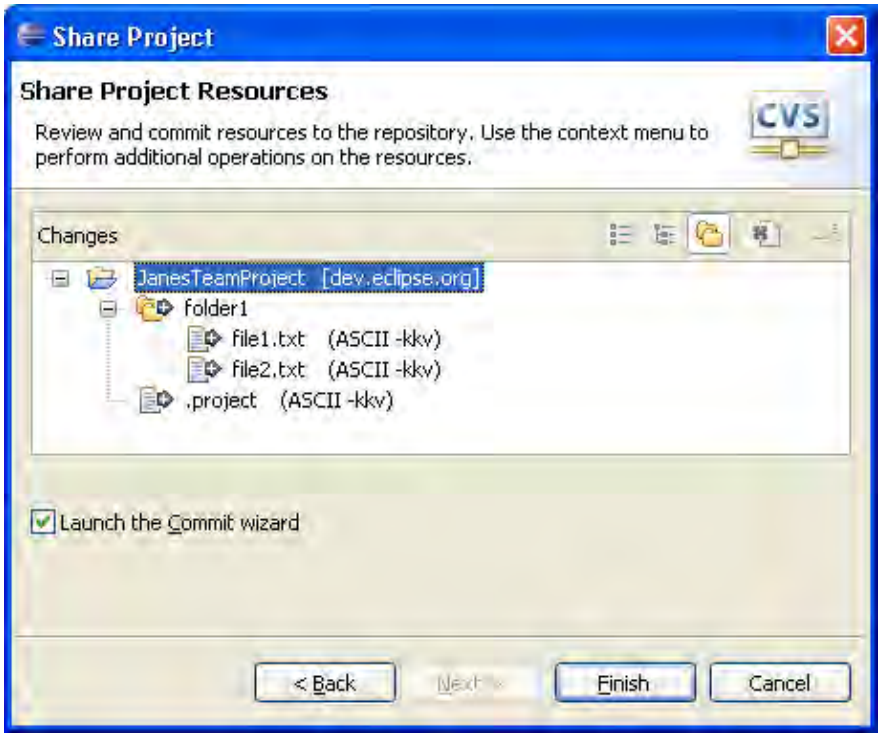
1. In one of the navigation views select the project JanesTeamProject.
2. From the project's context menu choose **Team > Share Project**. If more than one repository provider is installed, select ***CVS*** and select ***Next***.
3. In the sharing wizard page, select the location that was previously created.



4. The next page will ask you the module name to create on the server. Simply use the default value and use the name of the project you are sharing. Click Next.



5. The next page will allow you to see the files that are going to be shared with your team. The arrows with the plus sign show that the files are new outgoing additions. They have never existed on the server yet.



6. Press Finish and you will be prompted to commit the files. Enter a commit comment to describe the commit you are making. You can enter anything you like. Press ***Finish*** when you are done and the files will be committed.
7. Now you have shared the project and all the files have been committed to the CVS repository. Others can now see your files!

Working with another user

Instructions were given explaining how to create a project and commit it to the repository. First a repository location was specified. Next the project was shared with that location and the HEAD branch and the resources were added then committed.

The repository currently contains the project and the two files that were committed (file1.txt and file2.txt).

It's time to find that coworker (let's call him Fred) to work through some steps with you (for the remainder your name is Jane). This section will demonstrate how two people can work on the same project and simultaneously commit changes to the repository. Specifically, the following actions will be taken:

- Fred will import the project to his Workbench
- Fred will make changes to file1.txt and file2.txt.
- You will update these new changes and also add a new file called file3.txt.
- Fred will make more changes to some files while you change the same files. Fred will release his changes first.
- You will have to resolve the conflicts between files that you both changed.

Here is some terminology commonly used to describe changes:

incoming	Changes that are in the repository, which you have not yet updated to.
outgoing	Changes that you are committing.
conflict	Both you and the repository have modifications to the same resource.

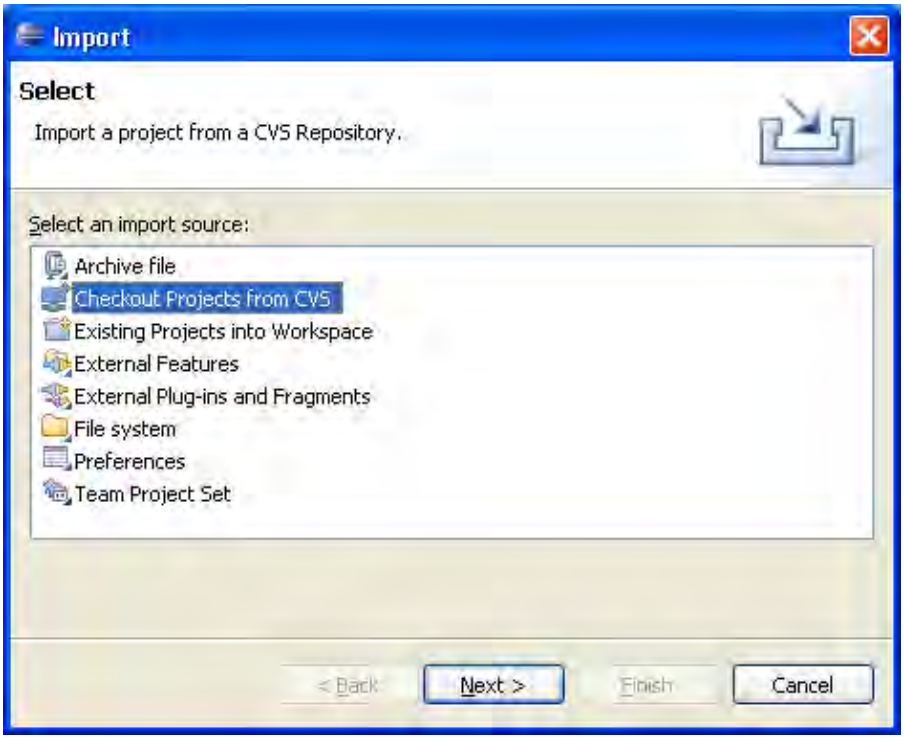
Checking out a project

Coworker Fred has several tasks in front of him:

- Fred will import the project that Jane committed to the CVS repository into his Workbench.
- Fred will make changes to file1.txt and file2.txt.
- Fred will synchronize and commit his outgoing changes to the two files.

Fred's first step is to import the project into his workspace as follows:

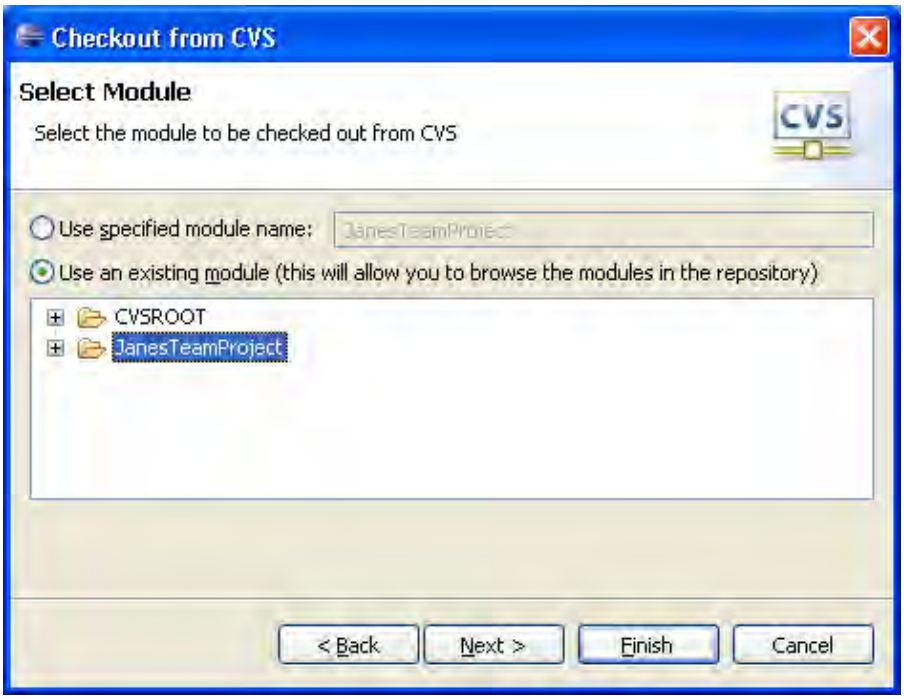
1. Run the import wizard via the File > Import menu item.



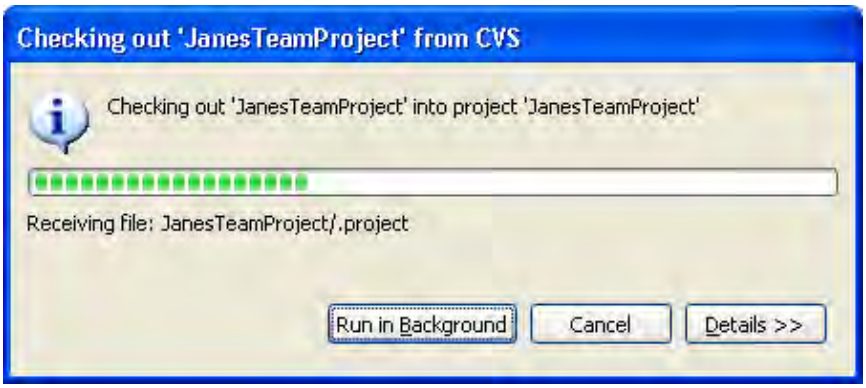
2. Select the Checkout Projects from CVS item and press Next.

3. Create a repository location as described when Jane created a repository.

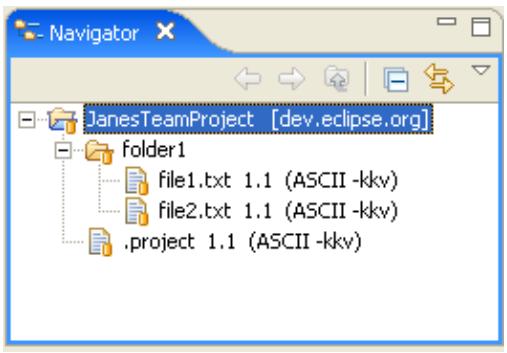
4. On the next page select the "Use an existing module" radio button and wait while the repository is contacted.



5. From the list of available projects, select JanesTeamProject and press Finish. A progress dialog will appear showing the progress of the import operation.



6. Open one of the navigation views and observe that it now includes the project JanesTeamProject. Notice that there are CVS decorators indicating the file revisions, the repository name, and the file types.



Another user making changes

Now that Fred has the project in his Workbench he needs to modify several of the files and synchronize with the repository in order to commit them.

- Fred will make changes to file1.txt and file2.txt
- Fred will synchronize and commit his outgoing changes to the two files.

Fred should proceed as follows:

1. Modify file1.txt as follows

   Original contents:

   This is the contents
   of file 1.

   New contents (changes shown in bold):

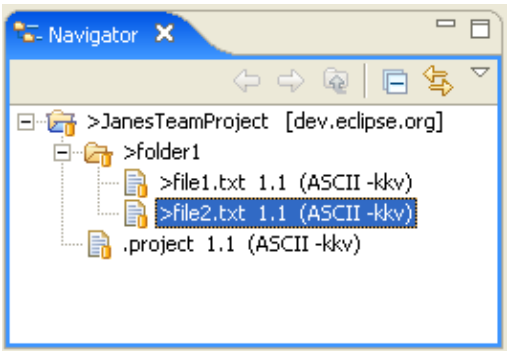
   This is the contents
   Fred−update
   of file 1.
2. Modify file2.txt as follows

   Original contents:

   File2 is a small file
   with simple text.

   New contents (changes shown in bold):

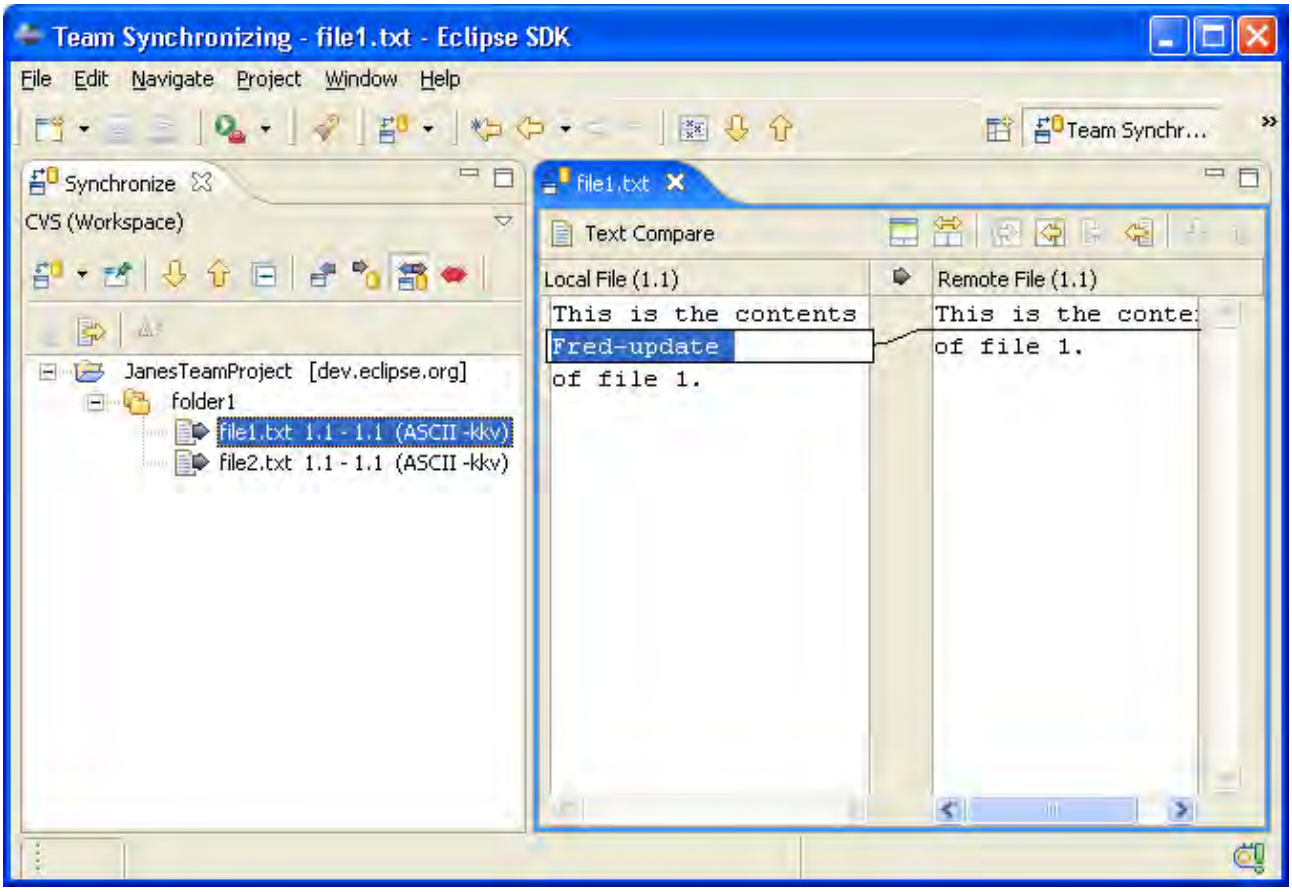
   *File2 is a (**Fred was here)** small file*
   with simple text.
3. Observe that the Navigator updates the CVS state of a resource. Notice that the two files which Fred has changed are preceded by ">".



4. To commit his changes to the repository Fred can either:
   - ♦ Select the two files and choose **Team > Synchronize with Repository** or,
   - ♦ Select the project and choose **Team > Synchronize with Repository**

   Often choosing the project is the easiest thing to do. Let's do that. Select the project and choose **Team > Synchronize with Repository** from its context menu. When you are asked to switch to the Team

Synchronizing perspective select Yes.



5. When the Synchronize View opens Fred can browse the changes he made and at the end should commit his changes to file1.txt and file2.txt by selecting JanesTeamProject in the Synchronize View and from the context menu select Commit. You will have to enter a commit comment before committing the changes.

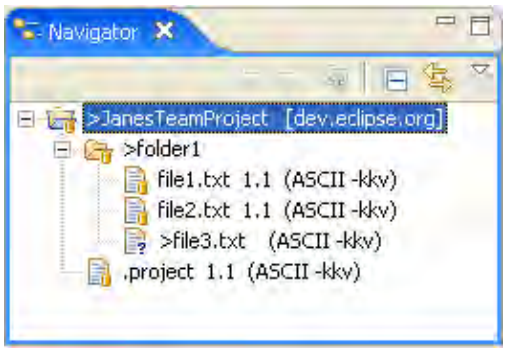
Making our own changes

Fred has made several changes to file1.txt and file2.txt and committed them to the repository. In this section additional changes will be made and then synchronized with the repository. When synchronizing, expect to see the changes made in this section along with changes that have been made by Fred.

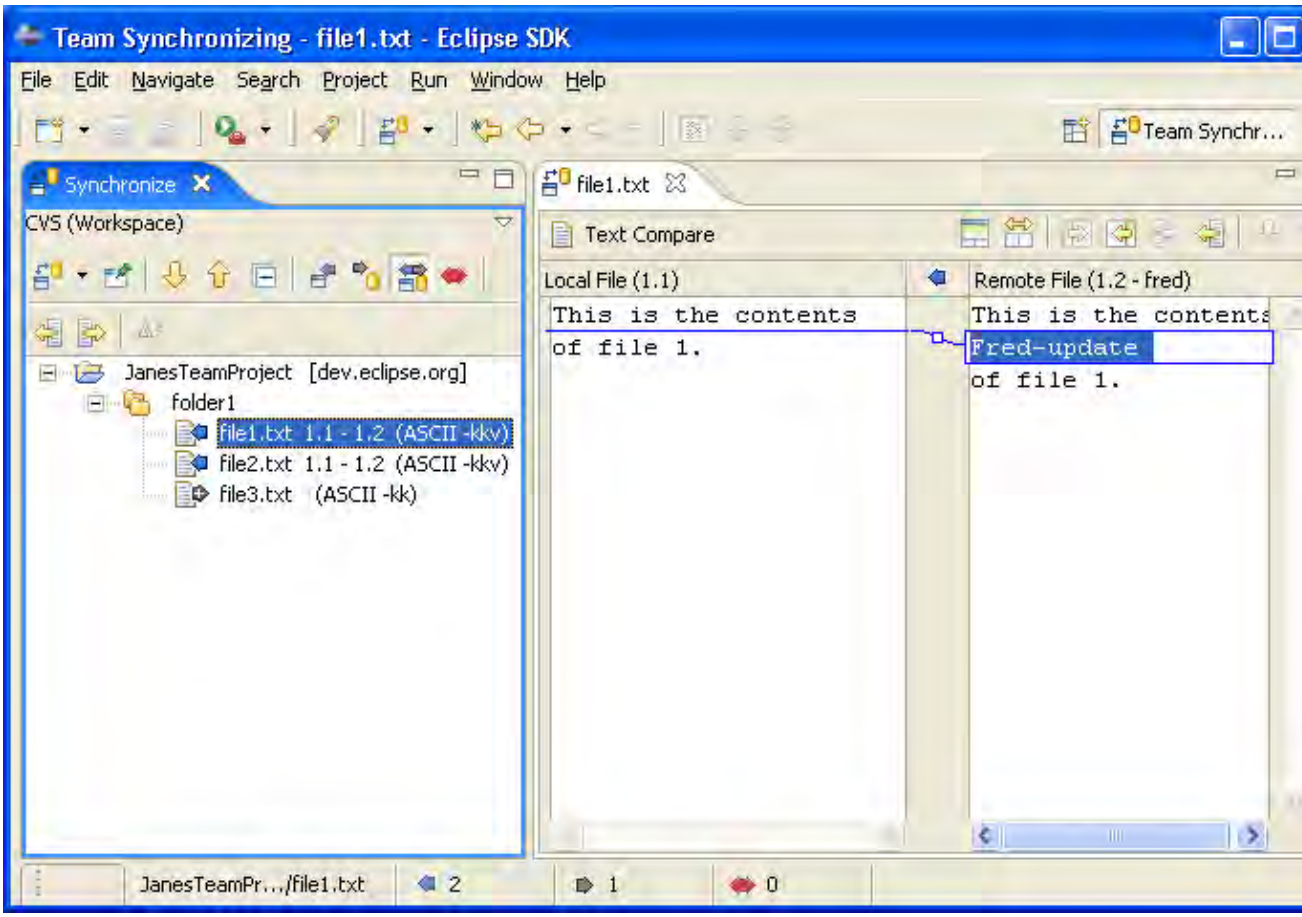
1. Add file3.txt as follows

   New contents:(changes shown in bold)

   This is the brief contents
   of file 3
2. Observe that the Navigator displays the CVS state of a resources. Notice that the new file added by Jane is preceded by ">".



3. Select the project JanesTeamProject and from the project's context menu, select **Team > Synchronize with Repository**. When asked to switch perspective select Yes. The Team Synchronizing perspective will open and you will see the files you have changed appear in the Synchronize View. Double click on file1.txt and you should see a compare editor open:

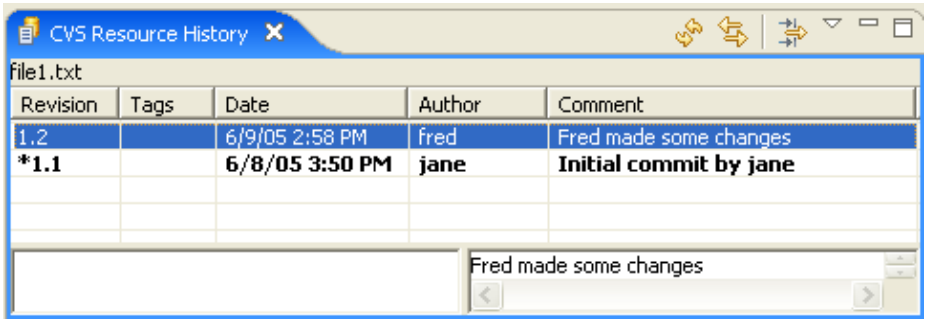


4. There are a couple of other things worth observing. First, the icon next to file1.txt (in the structured compare area) indicates that file1.txt has an incoming change. This means a change was released to the server which you need to take. Looking at file1.txt we can see the text that Fred has added to the file. Also, notice at the bottom of the window, in the status line, there are arrows with numbers beside them. These show the number of files you have incoming, outgoing, and in conflict. The first number beside a file is the revision you have in the workspace, and the other is the revision on the server when you last synchronized.

5. Normally you should update the changes made by others, then test your workspace with the changes, then commit your changes once you are sure that the new changes by others didn't break anything in

your workspace.

6. Before deciding to accept Fred's changes you may want to find out why he made the changes. Select file1.txt and from the context menu select Show in Resource History.



7. The row starting with a * indicates that this is the current revision loaded. In this case you can see the comment made by Fred when he released revision 1.2.
   Trick: You can select the 'link with editor' toolbar button in the history view to have the history automatically update when a new editor is opened. This allows for quick browsing of comments.

8. To update simply select JanesTeamProject in the Synchronize View and from the context menu select Update.

9. The Synchronize View will update to reflect the fact that file1.txt and file2.txt are no longer out−of−sync with the repository. You should only have file3.txt visible now.

10. Next you can commit file3.txt.

Working with conflicting changes

There are cases where two users are editing the same files and when the second to commit to the repository tries to commit their changes, the repository won't allow the commit to succeed because of the conflict. Let's simulate this by making Fred and Jane change the same files.

1. In Fred's workspace open one of the navigation views and edit file1.txt. Make the text the following:

   Fred line 1
   This is the contents
   Fred−update
   of file 1.

2. Fred will also change file2.txt with the following change:

   File2 is a (Fred was here again) small file
   with simple text.

3. Fred committed his changes to the repository.

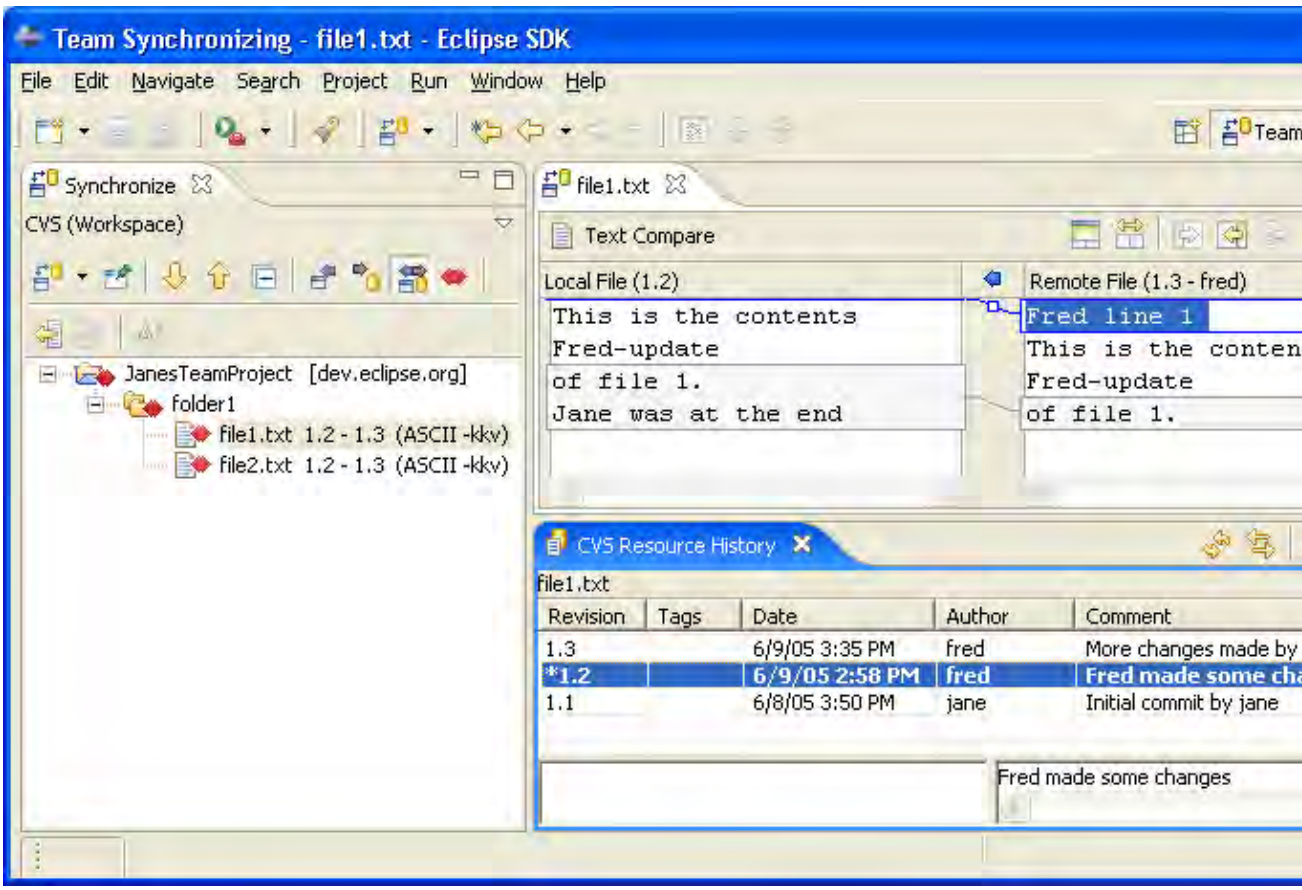
4. Next, at the same time Jane was making changes to file1.txt. She added the following line to the end of the file:

   This is the contents
   Fred−update
   of file 1.
   Jane was at the end

5. And finally, Jane changed file2.txt to the following:

   File2 is a (Jane was here) small file
   with simple text.

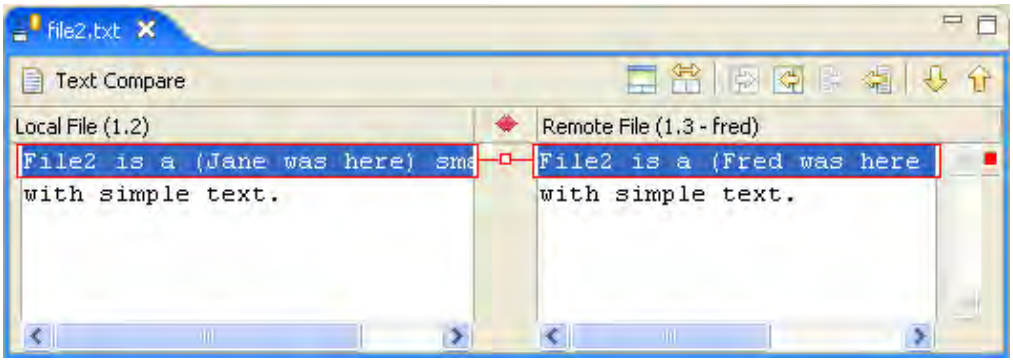
6. When Jane was finished making changes she synchronized the project and found the following appear in the Synchronize View:



7. Both file1.txt and file2.txt show with a red icon indicating that they have conflicting changes. You can't commit the files until the conflicts are resolved. Click on file1.txt and notice that Fred and Jane made changes to two different parts of the file. In this case, Jane can simply update the file and the lines added by Fred will be merged into Jane's local file. Select file1.txt and from the context menu select ***Update***.

8. Next, double on file2.txt to see the conflict. In this case you can see that both Jane and Fred changed the same line. For this type of conflicting change a regular update can't resolve the conflict. Here you

have three options (with the command to use in brackets): accept the changes from Fred (Override and Update), ignore Fred's changes (Mark as Merged), or manually merge the files within the compare editor.



9. For this example, let's say Jane updated file1.txt and selected override and update for file2.txt. After the operations are run the conflicts turn into outgoing changes. Jane can review the changes and commit them.

Replacing

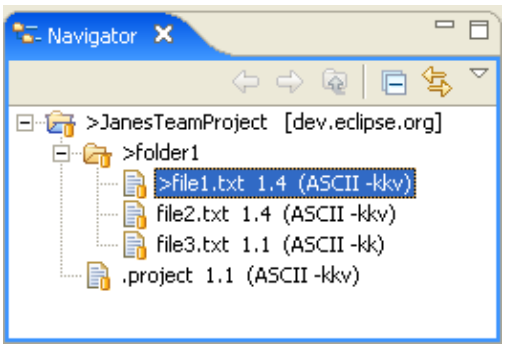
Suppose after further reflection, it becomes apparent that the revision of file1.txt that was just received is incorrect and an earlier revision is required. A Workbench resource can be replaced with an earlier revision of the resource from the repository. To roll back to an earlier revision:

1. In one of the navigation views select the file1.txt.

2. From the file's context menu, select **Replace With > Revision.**

3. In the replace with revision Compare editor that opens, choose the original (bottom) revision and, from the pop−up menu choose **Get Contents**.

4. Observe that the navigation view updates to show the CVS state of a resource. Notice that the modified file is preceded by ">" (see #1) indicating that file1.txt has changed (by replacing it with the earlier version).



5. Now assume that this older revision is not as good as it initially seemed and in fact the revision in the repository is better after all.

Instead of choosing **Team > Synchronize with Repository**, choose **Replace With > Latest from Repository.**

Observe that file1.txt is updated to be the contents from the repository, and that the leading indicator ">" has been removed since it is now the same revision as the repository.

Now that how to synchronize with the repository, replace with a revision or replace with the latest from the repository has been explained, these revisions/versions can also be compared with in a similar manner by choosing the **Compare With** menu operation from a resource's context menu in the navigation view.

Versioning a project

Now that the project is complete it is time to version it. While committing the file resources the repository automatically assigned them version numbers, or to be more accurate "revision numbers". As the files were committed the only information supplied was a commit comment.

To version a project proceed as follows:

1. Select the project JanesTeamProject.
2. In one of the navigation views select **Team > Tag as Version...**
3. When the Tag Resources dialog opens enter the version name A1 and click **OK**.
4. Select the CVS Repositories view.
5. Expand **Versions**, and JanesTeamProject. Observe that under JanesTeamProject there is now a version of this project whose version number is A1. Looking closely at the contents of folder1 note that the version numbers of the individual files match the version numbers in the Workbench.

A quick review

Here are some of the more important but subtle issues associated with working in a repository .

- The project was tagged as a version by versioning the project as it appeared in the Workbench. For this reason it is important to synchronize the project with the repository (that is the HEAD or the branch that is being worked in) prior to versioning it. Otherwise another user may have committed interesting changes to the project which have yet to be updated in the Workbench. By proceeding to version the project without updating, it will be versioned without these changes.
- The repository contains all projects in the repository. Individual users pick which projects they are interested in and check them out into the workspace. From that point on they are synchronizing those projects (only) with respect to the repository.
- The repository represents a large in−progress collection of all known projects. From the repository's perspective, everything in HEAD or on a branch is always open for change.
- The act of versioning a project effectively snapshots it and places it into the Versions section of the repository, however the repository branches are still open for change.
- It is important to first update to changes made to the repository, retest with those changes and the soon to be committed changes and then commit the changes. By first taking the latest changes in the branch, and retesting, it helps to ensure that the changes about to be committed will actually work with the current state of the branch.
- Each project is associated with a specific repository. Different projects can be associated with different repositories that may be on completely different servers.

Ant & external tools tutorial

This chapter covers Eclipse's external tools framework, and its integration with Ant, the build tool from the Apache Software Foundation. For more information on Ant, see the Apache Ant manual for documentation. This chapter assumes a basic knowledge of Ant.

This chapter is split into three main sections. First, the basics of working with Ant buildfiles in Eclipse are covered. Next, several Eclipse use cases will be worked through, in which Ant buildfiles can make life a little easier. Finally, the entire external tools framework and how to use non−Ant tools is considered.

Eclipse Ant basics

This section covers the basics of working with Ant buildfiles in Eclipse. Creating, editing, configuring and reusing Ant buildfiles will all be discussed in this section.

Creating Ant buildfiles

Ant buildfiles are just text files, so the most straightforward way to create an Ant buildfile in Eclipse is:

1. **File > New > File**.
2. Enter any name for the file, but make sure it has a .xml extension.
3. Click **Finish**.

As long as the file has a .xml extension, Eclipse will consider it to be a possible Ant buildfile, and will enable Ant−related actions when it is selected. Until a file has ant buildfile content, you will need to open it using **Open With > Ant Editor**.

As will be demonstrated in a later section, an Ant buildfile can be created for an Eclipse plug−in that contains predefined targets related useful for deploying the plug−in.

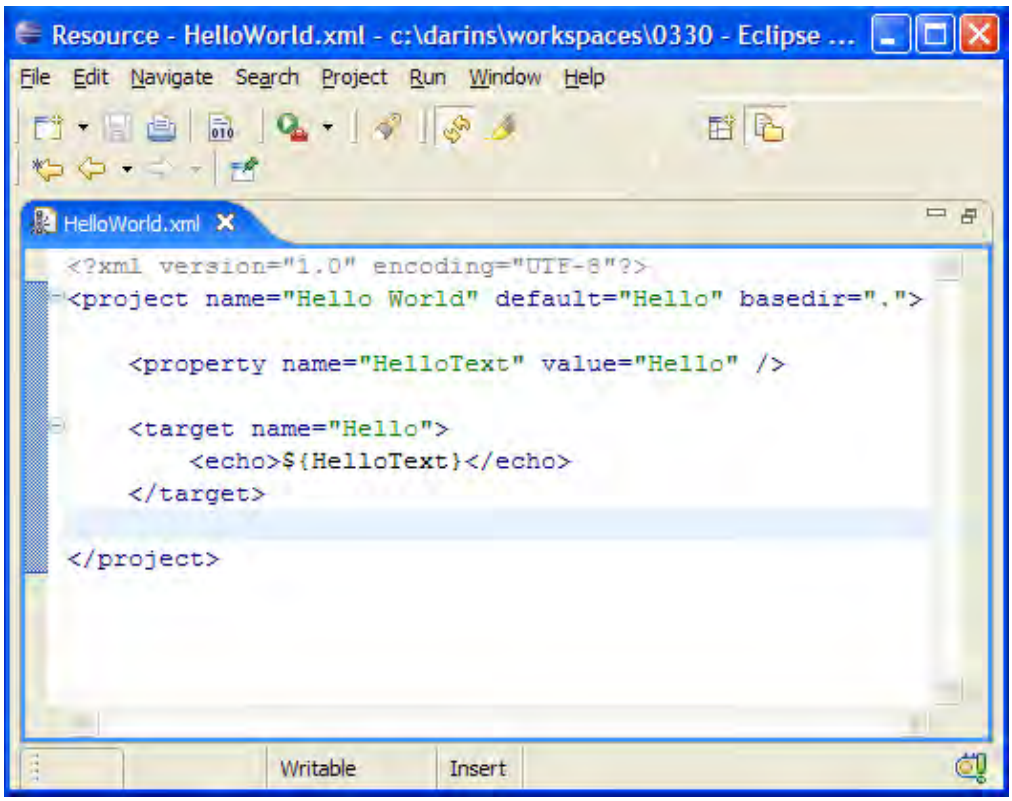
Editing Ant buildfiles

Because Ant buildfiles are simple text files, any text editor can be used to edit them. But there are several advantages to using the Eclipse Ant editor, including syntax coloring, content assist, navigation, occurrence marking and an outline view. To get familiar with the Eclipse Ant editor, create a simple Ant buildfile using this editor.

1. Create an Ant buildfile called HelloWorld.xml.
2. Open the Ant editor on the file by selecting **Open With > Ant Editor** from the file's context menu. ***Note:*** The default editor for a .xml file is a simple text editor until it has buildfile content, but this can be changed in the **Window > Preferences > General > File Associations**.
3. Enter the following content in the editor:

```
<?xml version="1.0" encoding="UTF−8"?>

<project name="Hello World" default="Hello" basedir=".">

        <property name="HelloText" value="Hello"/>
```

```
    <target name="Hello">
        <echo>${HelloText}</echo>
    </target>
</project>
```

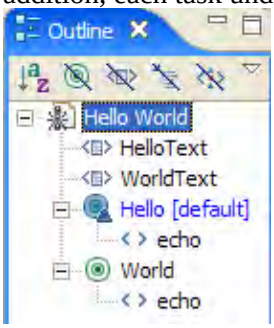


4. Notice the syntax coloring for property values.
5. Begin to enter a second target by typing '<tar', then hit ***Ctrl−Space*** to activate content assist. A list of valid completions is presented. Select the <target> completion and notice that the editor inserts both the opening and closing tags and leaves the cursor positioned to enter attributes for this tag.
6. Name this target 'World'.
7. Enter an 'echo' task for this target similar to that for the Hello target, but change the text to '${WorldText}'.
8. Add a dependency on the 'World' target from the 'Hello' target. Place the cursor after the definition of the name attribute, hit ***Ctrl−Space*** and select the depends attribute completion. Hit ***Ctrl−Space*** again to get the completion for the other target 'World' in the buildfile.
9. Save the buildfile contents.
10. The full content will now be:

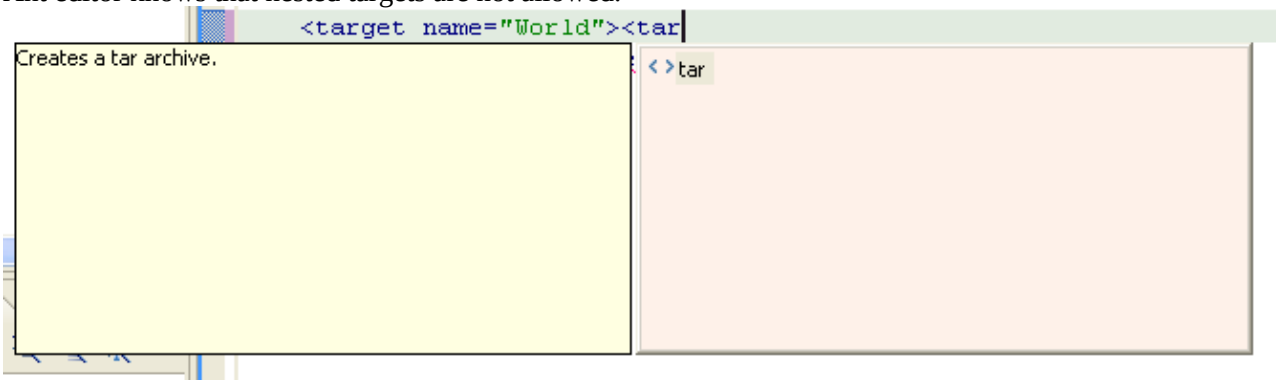
```
<?xml version="1.0" encoding="UTF−8"?>

<project name="Hello World" default="Hello" basedir=".">
```

```
<property name="HelloText" value="Hello"/>
<property name="WorldText" value="World"/>

<target name="Hello">

        <echo>${HelloText}</echo>

</target>

<target name="World">

        <echo>${WorldText}</echo>

</target>

</project>
```

11. Save the changes to HelloWorld.xml.
12. Make the Outline view visible and notice that there are entries for each property and each target. In addition, each task under a target has an entry.



13. Clicking on an entry in the Outline view will scroll the editor to that entry. In addition, when the Outline view has focus, typing a character will move the selection in the Outline view to the next visible entry beginning with that character.
14. Position the cursor just past the end of one of the '<target>' tags, type '<tar', then hit Ctrl−Space to activate content assist. Notice that now the only valid completion is the 'tar' tag. This is because the Ant editor knows that nested targets are not allowed.



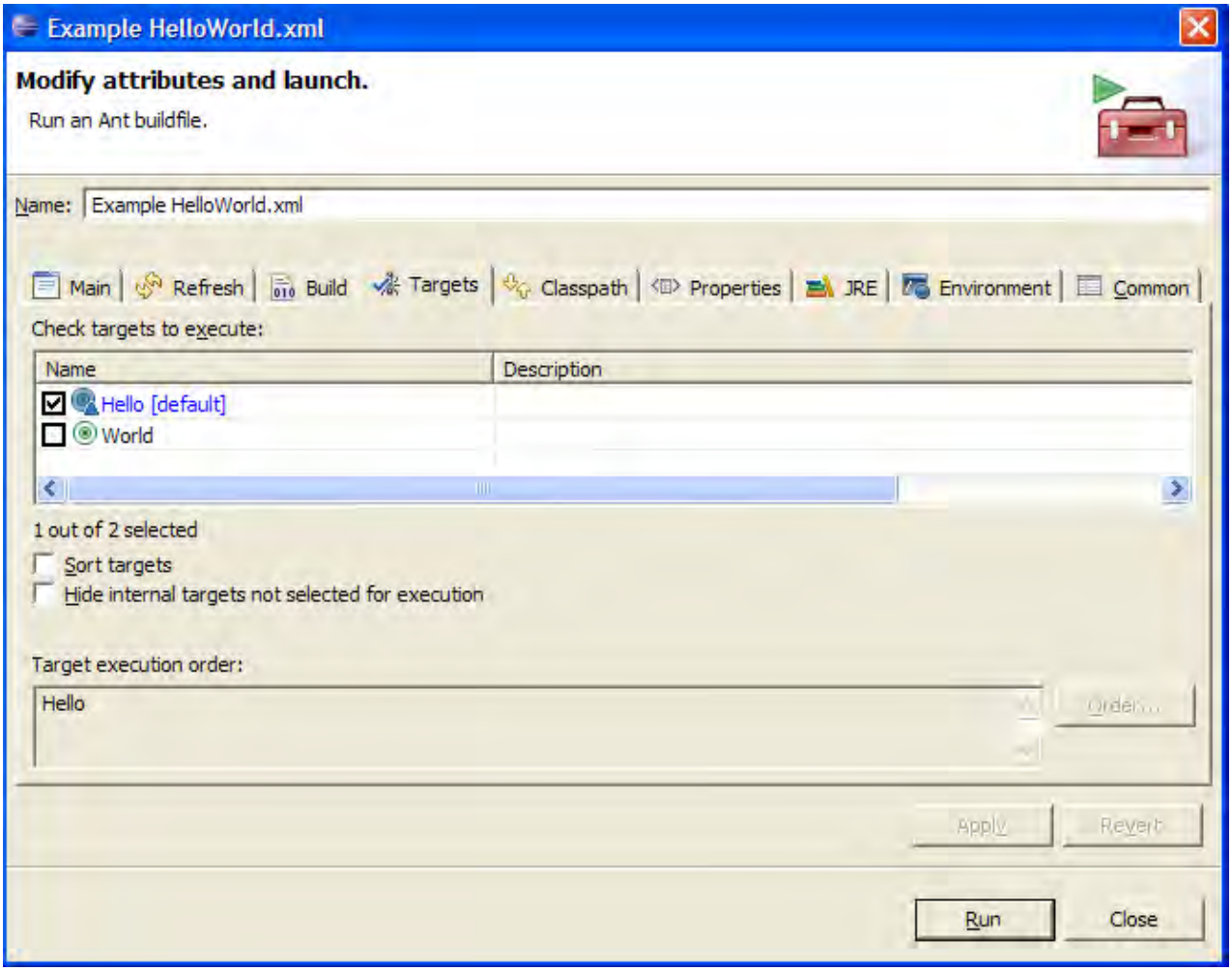
15. Close the editor and do not save changes.

Running Ant buildfiles

Any file with a .xml extension can be run as an Ant buildfile. Of course, not all such files really are Ant buildfiles, but no harm is done if an attempt is made to mistakenly run a non−Ant .xml file as an Ant buildfile.

In this section, the basic mechanisms for running Ant buildfiles in Eclipse will be covered, using the HelloWorld.xml file created in the last section.

1. Select HelloWorld.xml in one of the navigation views and choose **Run As > Ant Build...** from its context menu.
2. A launch configuration dialog is opened on a launch configuration for this Ant buildfile.



3. This dialog allows the configuration of many aspects of the way an Ant buildfile is run, but for now concentrate on the **Targets** tab which allows the selection of which Ant targets to run and their order. Select both targets and leave the order as the default.
4. Click **Run**.
5. The Ant buildfile is run, and the output is sent to the Console view.

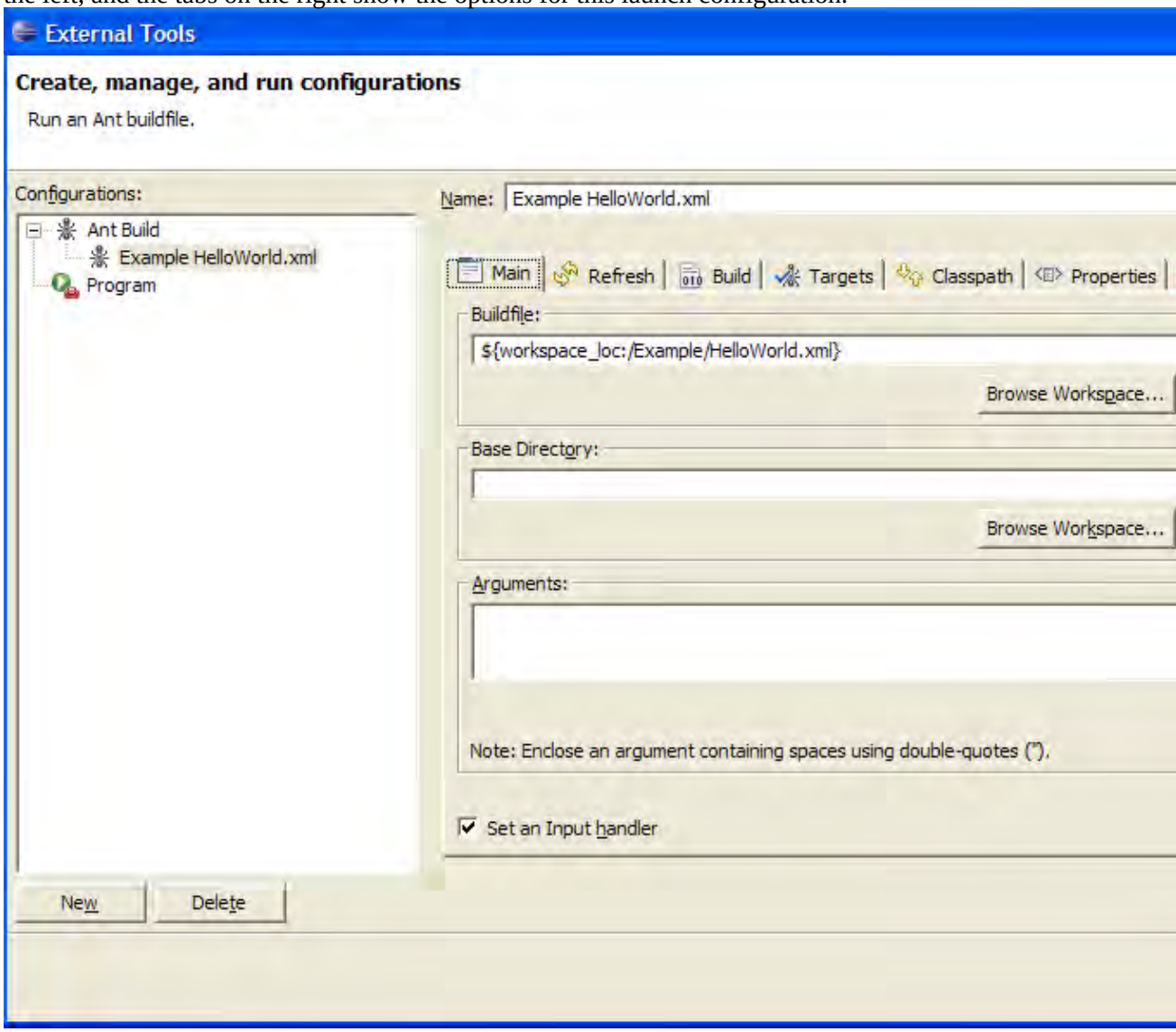
Fortunately, if the buildfile is run so that the Hello target is executed first and other times so that the World target is first, the dialog does not need to be brought up each time nor does the ordering need to be changed.

Saving & Reusing Ant options

When we ran the HelloWorld.xml Ant buildfile in the last section, the choice of targets, along with all other options in the Run Ant dialog were saved in an entity called a 'launch configuration'. Launch configurations contain all details necessary to run a single Ant buildfile in a particular way. It is perfectly valid to have multiple launch configurations associated with a single Ant buildfile. So, in addition to the launch

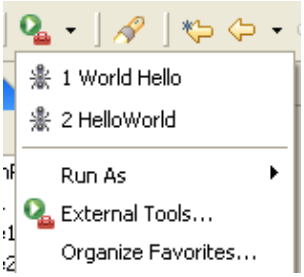
configuration that was created in the last step, specifying that our HelloWorld.xml buildfile should execute the targets Hello & World in that order, we could create a second launch configuration for this same buildfile specifying the same targets but in the reverse order. So far so good. But the really nice thing about launch configurations is that now you can quickly run your Ant buildfile in either configuration by simply specifying the corresponding launch configuration.

1. From the Workbench toolbar, select **Run > External Tools > External Tools...**.
2. The External Tools dialog opens and presents a choice of launch configurations to view and edit. The launch configuration we created when we ran the HelloWorld.xml buildfile is selected in the tree at the left, and the tabs on the right show the options for this launch configuration.



3. At the top of the dialog, change the Name to 'Hello World' and **Apply** the change.
4. In the tree at left, bring up on the context menu on the selected launch configuration and choose **Duplicate**. A copy of the launch configuration for the Hello World buildfile is created, '(1)' is appended to the name, and the new launch configuration is selected in the tree.
5. Rename the new configuration to 'World Hello'.
6. In the **Targets** tab, click the **Order...** button, change the order of the targets so that the World target executes first, and **Apply** the change.
7. Click **Run**.

8. As before, the HelloWorld.xml buildfile runs and sends its output to the Console view. This time however, because the targets were reversed, the output is reversed as well.
9. Go back to the External Tools drop down in the toolbar.



Notice now that there are two entries in the history, one for Hello World and one for World Hello. In order to rerun the Ant buildfile so that it outputs Hello World, just select this launch configuration in the history. To rerun the launch configuration that outputs World Hello, select this configuration in the history.
Note: The history is ordered so that the most frequently run configurations are at the top of the dropdown.

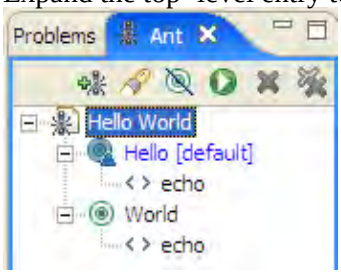
This concludes our quick look at the basics of Ant integration in Eclipse. In the next chapters, we consider several real−world use cases for running Ant buildfiles inside Eclipse.

Using the Ant view

Eclipse provides a standard view, the Ant view, that lets you work with your Ant buildfiles. This view is tree−structured, showing Ant buildfiles as top−level entries and targets & internal targets as children. The main advantage of this view is that you can work with all of your Ant buildfiles in one place, as opposed to hunting them down in one of the navigation views.

1. Open the Ant view from the workbench menu by selecting **Window > Show View > Ant**.
2. By default, the Ant view is empty. There are three ways to add Ant buildfiles to this view:
   - Click ***Add Buildfile*** . This brings up a dialog in which you explicitly select those Ant buildfiles you want to add
   - Click ***Add Buildfiles with Search*** . This brings up a search dialog in which you can specify a filename pattern for your Ant buildfiles and search within the entire workspace or a specified working set.
   - Drag and drop buildfiles from other views to the Ant view.

     Once added to the Ant view, Ant buildfile entries remain in the view across workbench invocations until explicitly removed or the associated project is closed.
3. Click ***Add Buildfiles with Search***. Suppose you only remember that the buildfile you want to work with starts with 'H'. Enter 'H*.xml' for the buildfile name. Make sure Workspace is selected for the scope, then click Search. The HelloWorld.xml file is found and placed in the Ant view.
4. Expand the top−level entry to see the default target Hello, and the internal target World.



5. Select the World internal target and click ***Run the Selected Target***  . Notice that just the World target gets executed.
6. Select the top−level HelloWorld buildfile and click ***Run the Default Target of the Selected Buildfile***  . Notice that just the default target, Hello, gets executed.
7. To edit your buildfile, bring up the context menu on the HelloWorld file and select **Open With > Ant Editor**.
8. To edit the default launch configuration, select ***Run As > Ant Build...*** from the context menu.
9. The Run Ant launch configuration dialog appears. Here you can modify the way in which the buildfile is run from the Ant view.
10. Select the HelloWorld file, then click the Remove button. The buildfile is removed from the view. ***Note***: This does not delete the file from the workspace.

Use cases for Ant in Eclipse

Now that we've seen how to work with Ant buildfiles in Eclipse, let's look at two ways that Ant buildfiles can be useful in Eclipse.

- Deploying Eclipse plug−ins
- Building projects

We'll start by using an Ant buildfile to handle various aspects of deploying Eclipse plug−ins.

Deploying Eclipse plug−ins

The first use of Ant within Eclipse discussed here will be as a plug−in deployment tool. When developing Eclipse plug−ins, it's frequently handy to build the jar files for your plug−in and test them as part of an Eclipse install. The Ant buildfile we'll look at in the next section will do this and much more.

Creating a HelloWorld plug−in

To work through this example, we need to create a simple 'Hello World' Eclipse plug−in. Creating Eclipse plug−ins is covered in more detail elsewhere in this documentation, so we will just create a simple HelloWorld plug−in as follows:

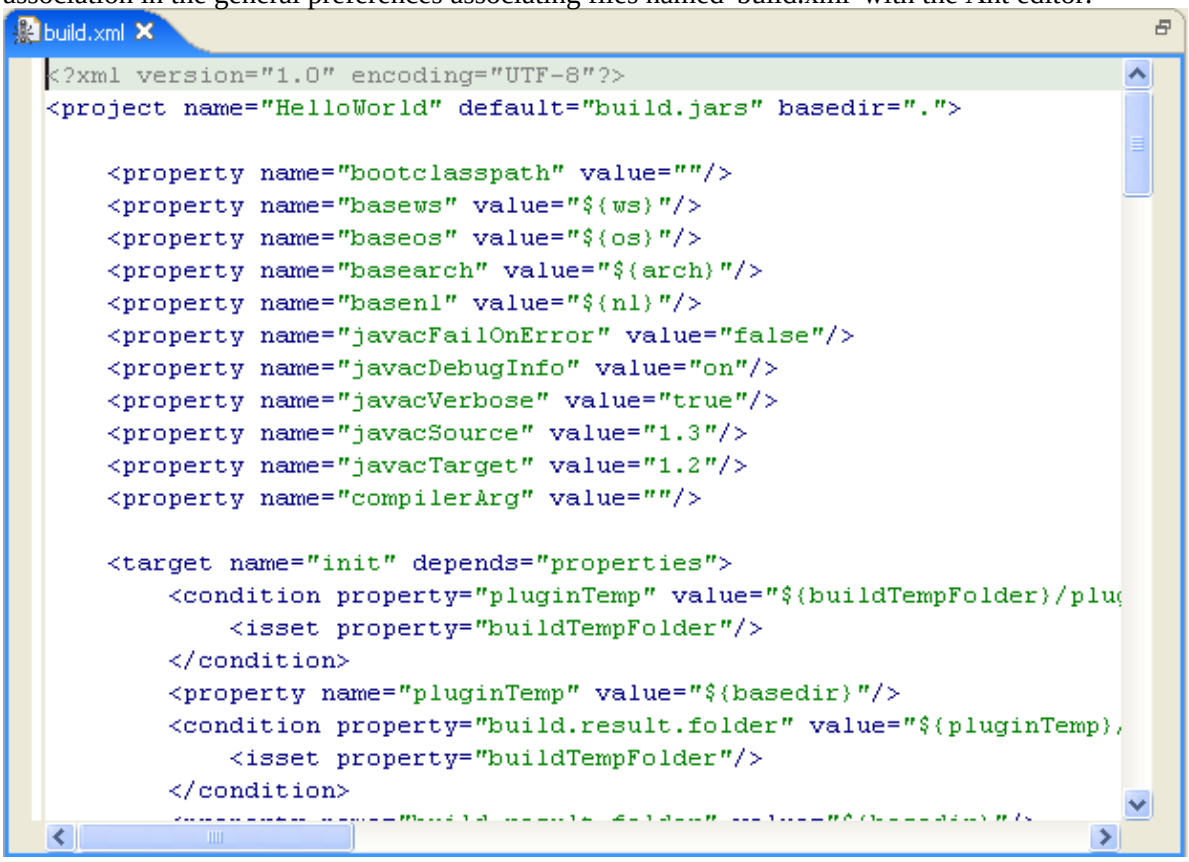
1. Create a new plug−in project by menuing to **File > New > Other...**
2. Select **Plug−in Development** in the left pane, and **Plug−in Project** in the right and click **Next**.
3. Enter a name for your project, then click **Next**.
4. On the next page, accept all defaults and click **Next**.
5. Make sure **Create a plug−in project using one of the templates** is selected, then select **Hello, World** from the list below and click **Next**.
6. Accept all the defaults on this page and click **Finish**.

Now that we have a plug−in project, we can create a build.xml file for it to manage plug−in deployment.

Generating the build.xml file

Open one of the navigation views and locate your new plug−in project.

1. Select the file 'plugin.xml' underneath your new project and right−click **PDE Tools > Create Ant Build File**.
2. A new file, 'build.xml' will be created underneath your project.
3. Open the Ant editor on the build.xml file by double clicking it in one of the navigation views. Notice that for this file, the Ant editor is the default editor. This is because by default, there is a file association in the general preferences associating files named 'build.xml' with the Ant editor.



The build.xml file is a default Ant buildfile with many targets useful for deploying your plug−in. For example, there are individual targets to build all necessary jars for your plug−in, create a .zip file containing everything in your plug−in, cleanup any .zip files that may have been created and so on.

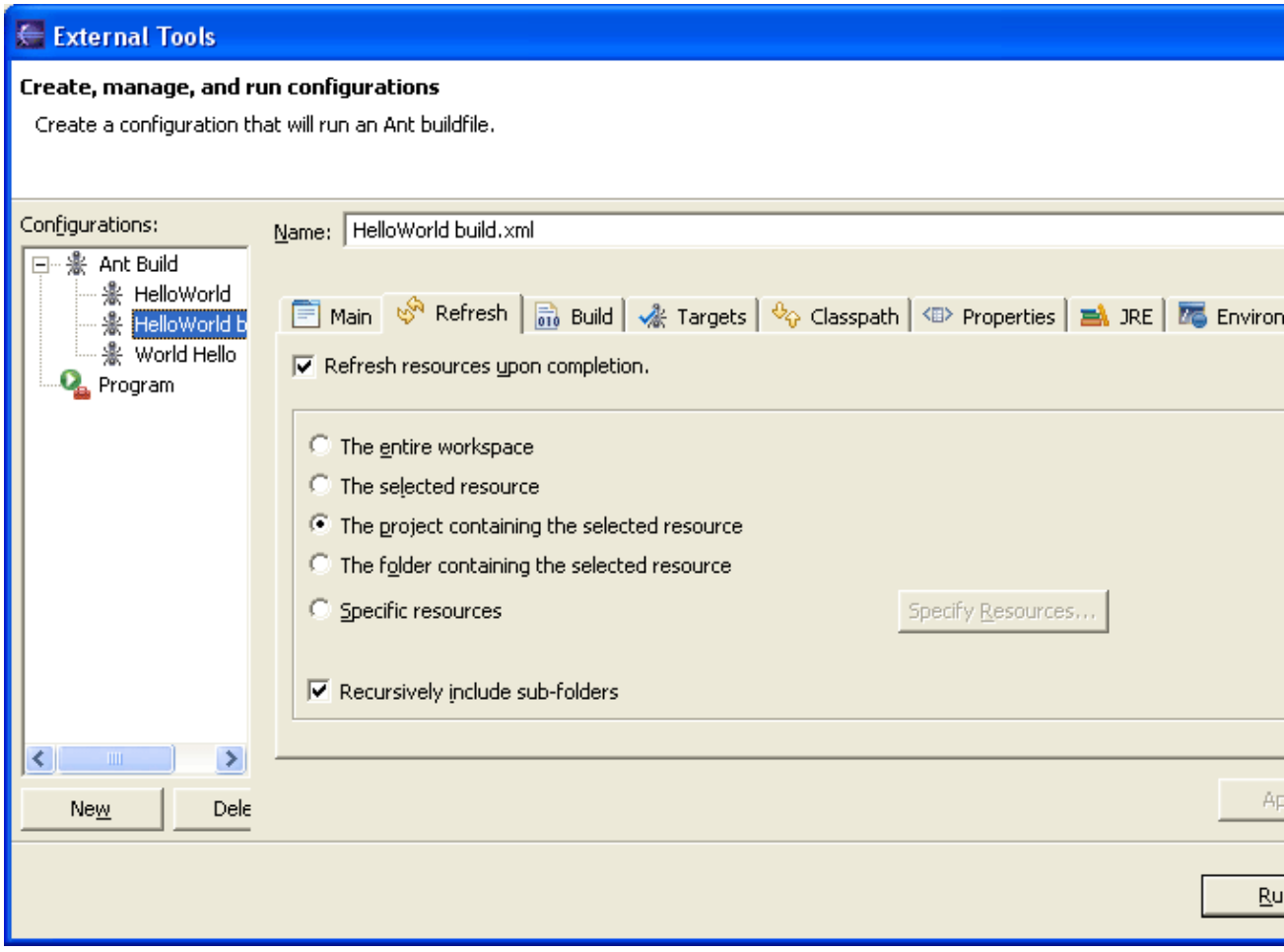
Now that we have the build.xml file, we will use it to build a .jar file for our plug−in.

Building a .jar file for the plug−in

Now we are ready to use the build.xml Ant buildfile that was created in the last section:

1. Select the build.xml file in one of the navigation views and choose **Run > External Tools** from the context menu.
2. In the dialog, the Targets tab should be visible. Make sure that 'build.jars' is the only item in the list selected.
3. Switch to the **Refresh** tab. The options on this tab control if and how the workspace looks for new, changed and deleted resources after the buildfile finishes execution. Because refreshing can be an expensive operation, you should refresh the smallest subset of the workspace necessary for your buildfile.
4. Check **Refresh resources upon completion**, then select **The project containing the selected**

resource.



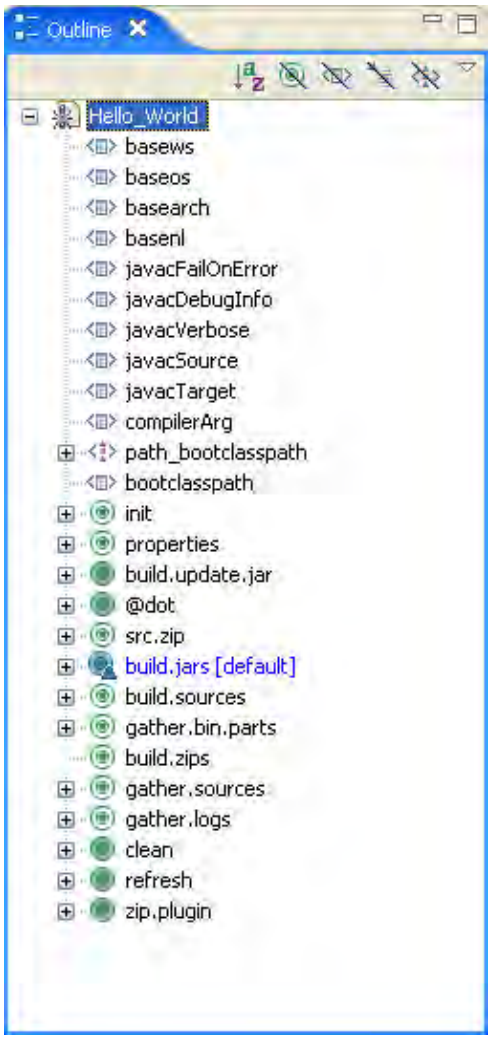
5. Click **Run**.
6. The Ant buildfile runs, executing the build.jars target. Output from the buildfile is visible in the Console view.
7. When the buildfile finishes, notice that there is a new .jar file under your project containing the two class files that comprise your plug−in. If we had chosen not to refresh the project, the .jar file would still have been created, but we would not see it in one of the navigation views.

We've run the default target for the build.xml Ant buildfile, but there are many other targets and options to be explored.

More plug−in deployment options

Now that we've run the default target for our build.xml buildfile, it's time to explore some more options.

As can be seen in the Outline view, the default build.xml Ant buildfile has a number of targets.



Solid green circles denote targets, while internal targets are represented by hollow green circles. The following are the useful top−level targets:

- **build.update.jar** This target builds jars for the plug−in, then zips them into a form suitable for use with the Eclipse Update Manager.
- **HelloWorldPlugin.jar** This target does the work of the building jars for the plug−in.
- **build.jars** This is the default target we ran in the last section. It defers to the HelloWorldPlugin.jar target to do its work.
- **clean** This target deletes all zips, jars and temporary directories that may have been created by other targets in this buildfile.
- **zip.plugin** This target builds executable and source jars, then zips everything into a single archive.

Remember that this build.xml file is simply the *default* deployment buildfile for an Eclipse plug−in. There will be times when you will want/have to modify this buildfile to handle your particular projects. For example, if you have a directory that contains resources necessary for your plug−in, you will want to update the build.xml file to include this directory in the jars and zips it creates.

This completes our look at using Ant buildfiles to deploy Eclipse plug−ins. The key points to remember are that you can easily create a default deployment buildfile for a plug−in that contains many useful targets, and that while this default buildfile is useful, you may have to modify it to suit your needs.

Ant buildfiles as project builders

Our second practical example of using Ant buildfiles in Eclipse is a 'project builder'. This is an Ant buildfile that has been designated to run whenever a project is built. The uses for such a buildfile are many:

- Generate a .jar file containing class files from your project
- Perform some type of pre or post build processing on source or binary files in your project. For example:
    - ♦ Pre−processing source files to instrument them for performance analysis
    - ♦ Obfuscating binary files to prevent reverse engineering
- Copy class files to some location (for instance, on a network)

For this example, we will create an Ant buildfile that creates a .jar archive of the class files in a project. Note that this is similar to the example in the last chapter, in which we used an Ant buildfile to generate .jar files for an Eclipse plug−in. This example differs in that it works for any Eclipse project, whether or not it is also an Eclipse plug−in.

Creating a project builder Ant buildfile

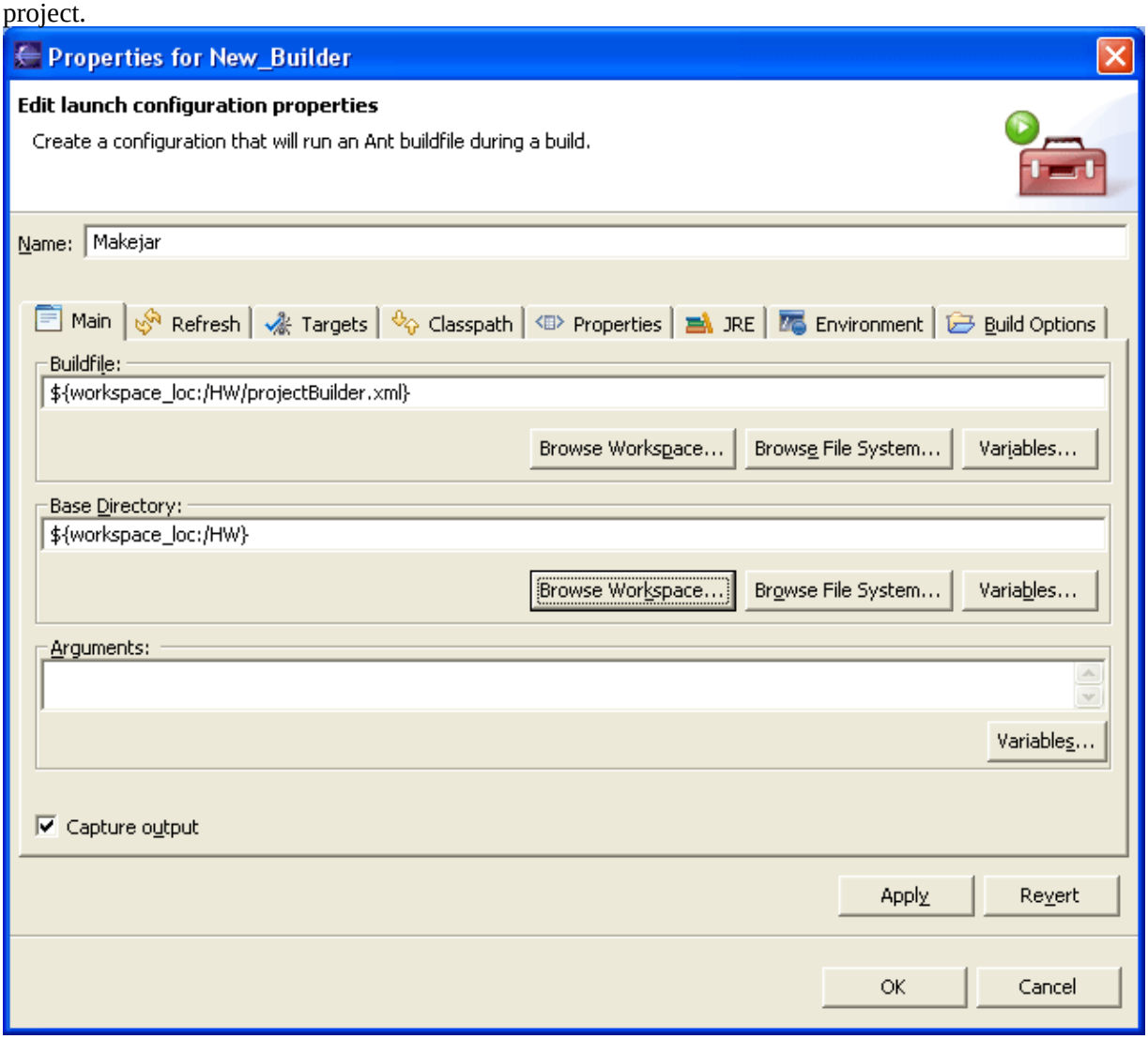
To see how project builders work, we will create a simple project with a single source file and an Ant buildfile that jars up the single class file. Though this example uses Java, it should be noted that project builders are available for all projects, Java or otherwise.

1. Create a Java project named 'HW'.
2. Create a Java source file named 'HelloWorld' with a main method.
3. Put a single 'System.out.println()' statement in the main method, and make it print a greeting of your choice.
4. Save changes.
5. Create a file named 'projectBuilder.xml', open the Ant editor on it, enter the following content, and save changes.

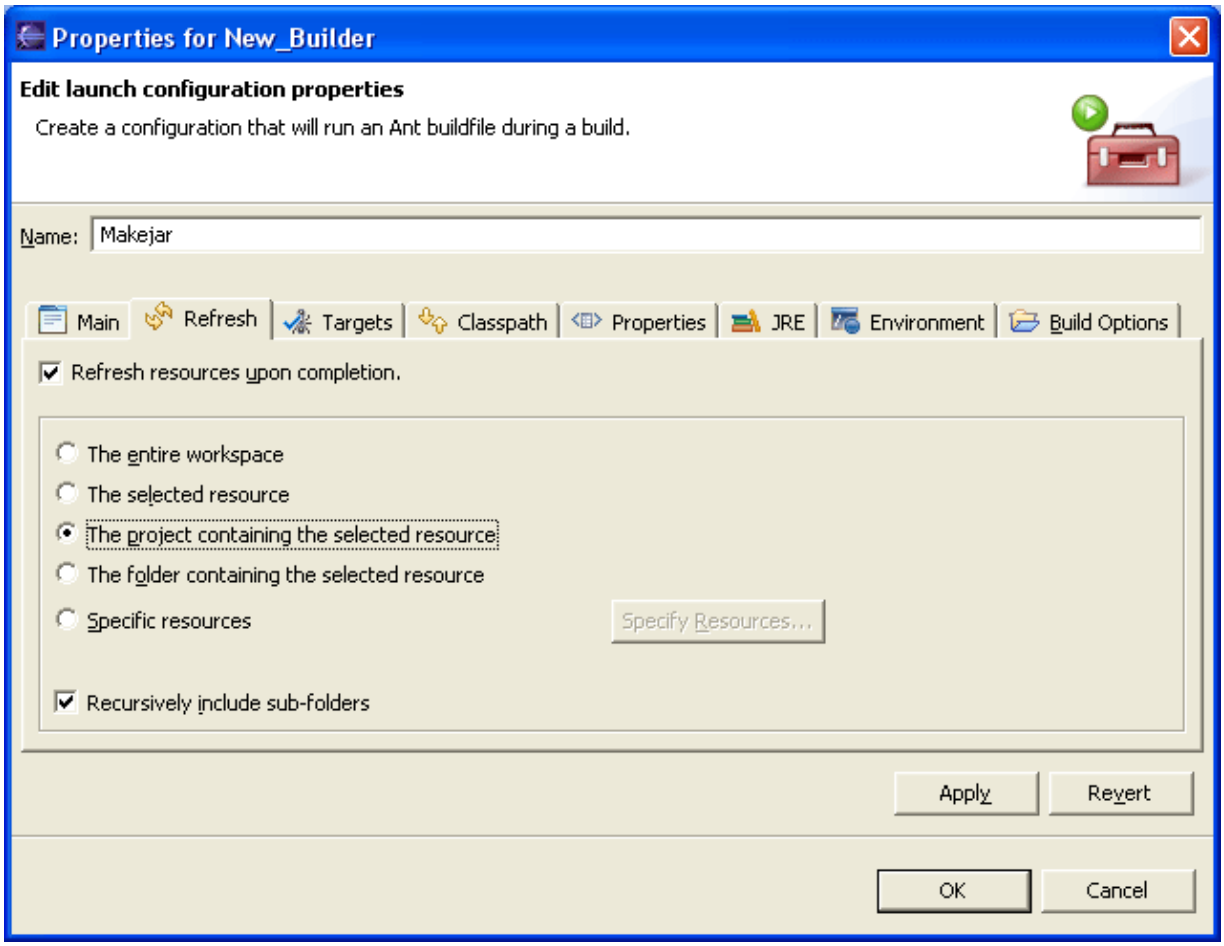
```
<?xml version="1.0" encoding="UTF−8"?>
<project name="HW.makejar" default="makejar" basedir=".">

      <target name ="makejar" description="Create a jar for the
      HW project">

            <jar jarfile="HelloWorld.jar"
            includes="*.class" basedir="."/>

      </target>

</project>
```

6. In one of the navigation views, select the HW project and choose **Properties** from its context menu.
7. In the project properties dialog, select **Builders**, then click **New...**.
8. In the ***Choose configuration type*** dialog, select ***Ant build***, and click **OK**.
9. The ***External Tools*** dialog appears. Set the name to 'Makejar'. In the Main tab, click the ***Buildfile*** **Browse Workspace...** and set the **Location** to be the projectBuilder.xml buildfile created above. Then click the ***Base Directory*** **Browse Workspace...** and set the Base Directory to be the HW

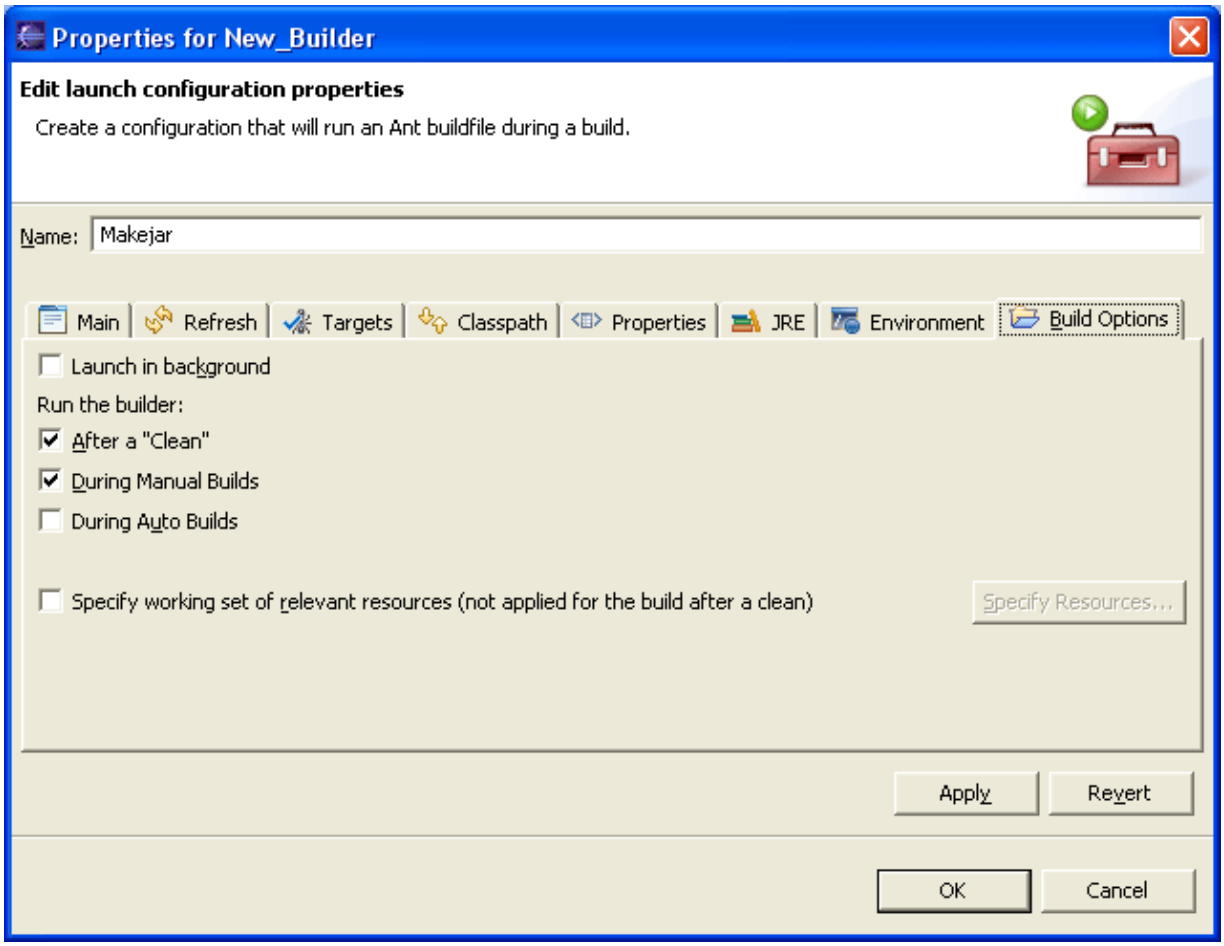
project.



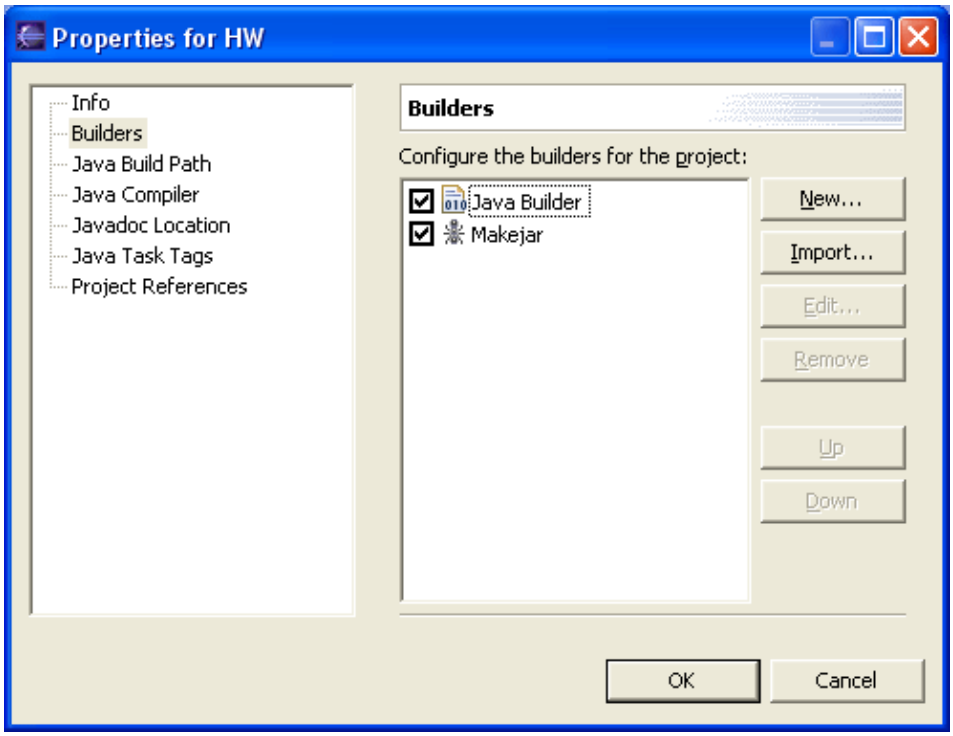
10. In the **Refresh** tab, we want to be sure that when our HelloWorld.jar is created, we see it in Eclipse. By default, no refreshing is done when a project builder finishes running, so check **Refresh resource upon completion**, then select **The project containing the selected resource** in the list of scope variables. Because refreshing can be expensive, you should in general refresh the smallest entity that contains all resources that will be affected by your buildfile.



11. In the **Targets** tab, the default target should be selected.
12. In the **Build Options** tab, you can specify when this project builder is executed. By default, this is set to ***After a "Clean"*** and ***During Manual Builds***. Running your project builder during auto builds is possible, though not recommended because of performance concerns.



13. Apply the changes and click **OK**.
14. Back in the project properties dialog, you will now see a project builder named 'Makejar' that is set to run after the default Java Builder. Click **OK** to save the project builder and close the dialog.
    Note: You can change the order so that your Ant buildfile runs before the Java builder, though that wouldn't make sense in this example.



The Java Builder runs the internal Eclipse Java compiler which in turn is responsible for indexing your source so that searching, refactoring and many other features are available. Thus it is not possible to replace the internal Eclipse Java compiler by using a project builder. You can disable the Java Builder and you can control when the Java Builder runs with respect to the project builders that you define.

Executing project builders

The whole point of project builders is that they are not explicitly run by the user, but instead are run any time a qualifying build takes place for the project that owns the buildfile. Remember that the builders are triggered as indicated on the ***Build Options*** tab in the External Tools dialog and can be any combination of after a clean, during a manual build or during auto builds. Let's see how this works.

1. Select the HW project in one of the navigation views. In the workbench menu bar, choose **Project > Clean...** Select **Clean selected projects** and click **Ok**
2. The project is rebuilt and the projectBuilder.xml buildfile is run. Notice the output from this buildfile in the Console view.
3. Make sure the Autobuild preference is turned on, then make some trivial change to HelloWorld.java and save the change. The save triggers an auto build, but the auto build does not trigger the project builder.
4. Suppose we don't want to see the buildfile output every time it runs. Go back to the External Tools Builders page of the project properties dialog on HW. Select the Makejar entry and click **Edit...**. On the **Main** tab, uncheck the **Capture Output** option, apply the change and exit back to the workbench.

This concludes our look at Ant buildfiles as project builders in Eclipse. It's worth repeating that though this example used a Java project, project builders are not tied to Java, and may be used for any type of project.

External tools

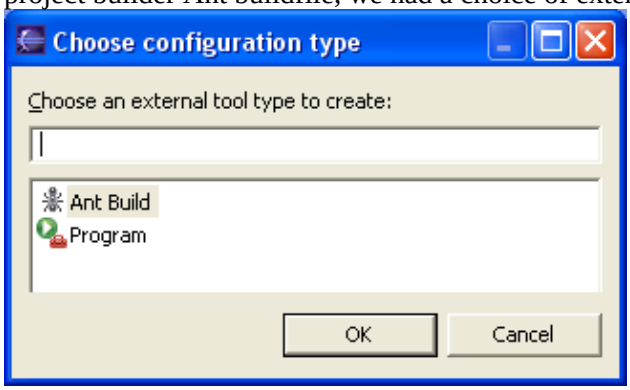
Up to this point in the tutorial, we've discussed Ant as if it were the only type of external tool available for use in Eclipse. The fact is that Eclipse's external tools framework handles two broad classes of external tools:

- Ant buildfiles
- Everything else

In this section, we look at ways to set up and use non−Ant external tools.

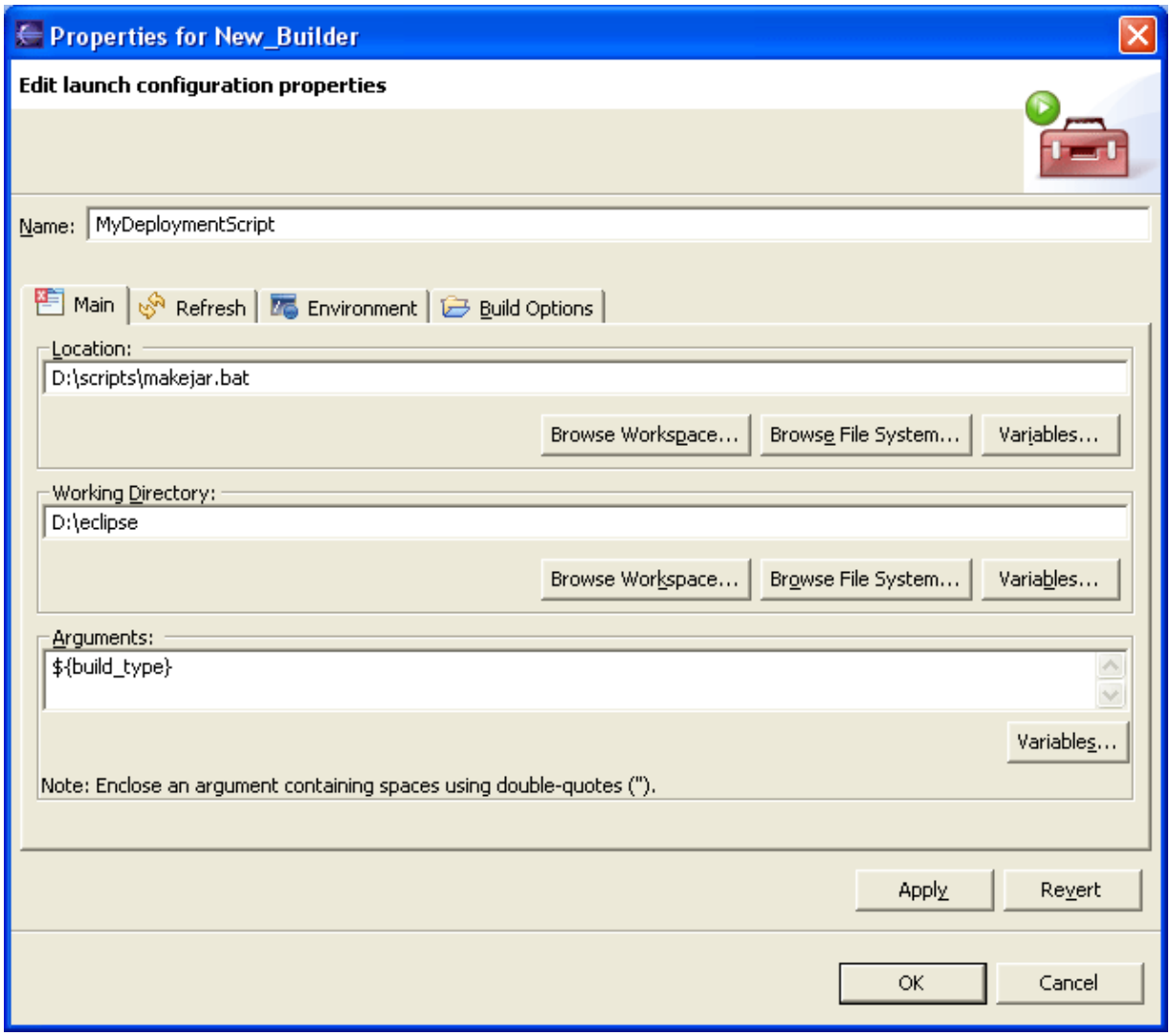
Non−Ant project builders

When we worked through our project builder example, you may have noticed that when we created our project builder Ant buildfile, we had a choice of external tool type:



The **Program** option is essentially a catch−all, allowing you to define an external tool for any executable file that is accessible on your local or network file system. Suppose that instead of Ant, you prefer to use your own shell scripts or Windows .bat files to jar up and deploy your Eclipse projects. You would then create a Program external tool that specified where and how to execute your script.

1. Create a script that performs your preferred deployment steps.
2. Select the project that you wish to build in one of the navigation views, and choose **Properties** from the context menu.
3. Select **Builders**, click **New...**, select **Program** and click **OK**.
4. The External Tools dialog appears, configured for Program type tools.
5. Enter the location of your script, its working directory, and any required arguments.



6. In this case, the script is a Windows .bat file, but it could be a Linux shell script, a Perl script or just about anything else that can be executed on your system.
7. The **Refresh** and **Build Options** tabs are identical to the tabs we saw for Ant project builders. In particular, the **Build Options** tab allows us to control what types of builds trigger our project builder buildfile.
8. Apply the changes, and click **OK**.
9. As with Ant project builders, we can control the ordering of this project builder with respect to other project builders (such as the default Java Builder for Java projects).
10. Rebuild your project. This will trigger your script to execute. Any output it generates will be sent to the Console view.

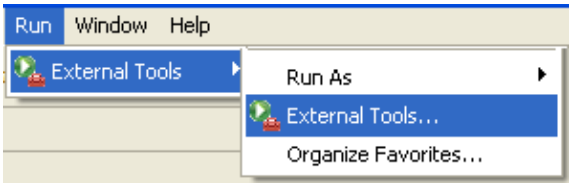
Ant is a popular tool for configuring and deploying projects. But if you prefer some other tool, or prefer to do it yourself, you can set up a Program external tool project builder. This allows you customize the deployment of your project as you see fit, while keeping the convenience of automatically running your script every time your project is built.

Stand−alone external tools

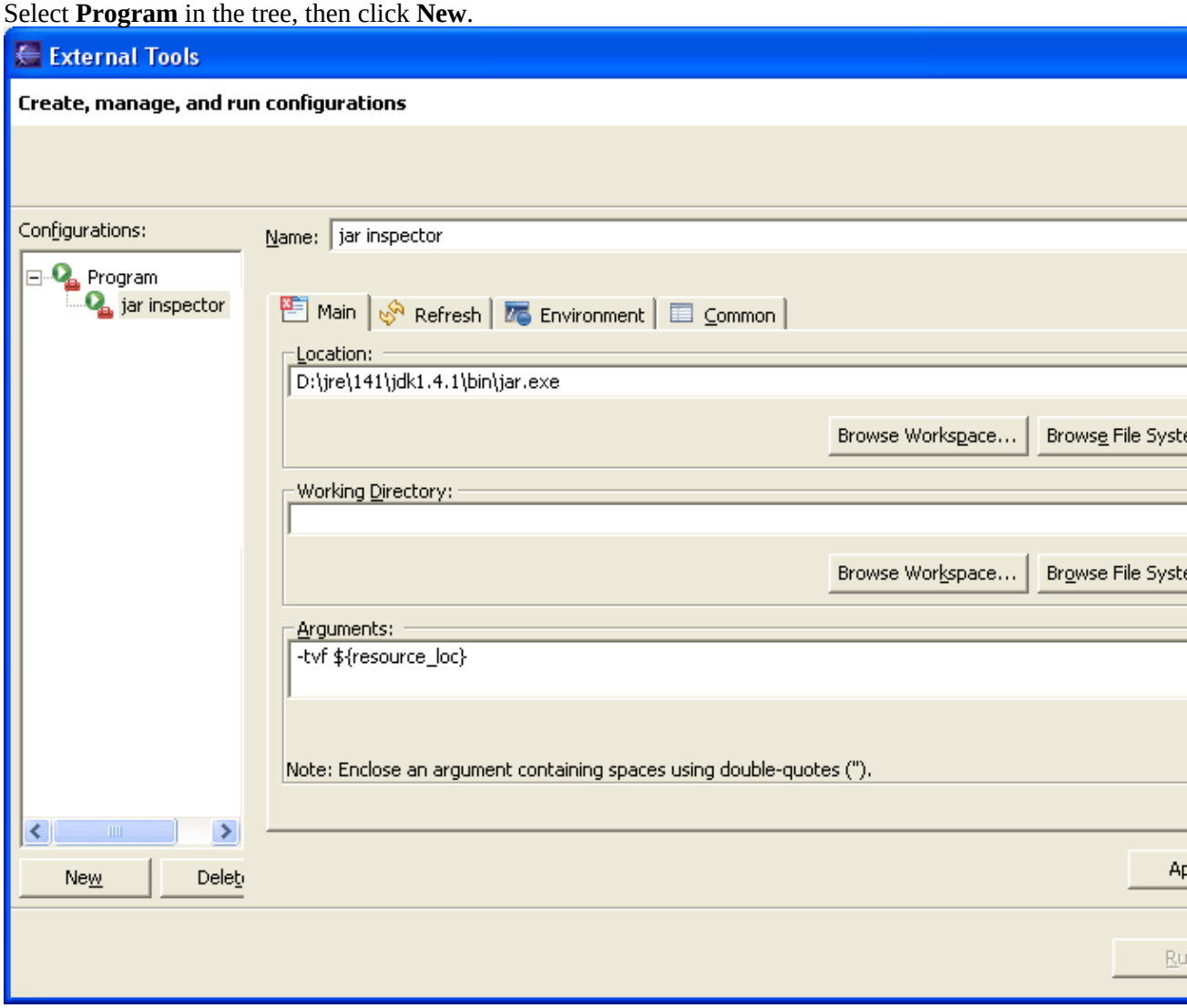
For the ultimate in external tool flexibility, create a 'stand−alone' external tool launch configuration. This is similar to the project builder launch configurations discussed in the last section, except that it need have

nothing to do with project building, and you can explicitly run it whenever you choose. Suppose you wanted to have a way to quickly see the contents of a .jar file in your workspace using the jar utility.

1. Select some .jar file in your workspace.
2. Select **Run > External Tools > External Tools...** from the workbench toolbar.



3. Select **Program** in the tree, then click **New**.



4. Name the launch configuration 'jar inspector'.
5. Use the first **Browse File System...** button to locate the jar executable.
6. In the **Arguments** field, type '−tvf' and a space, then click **Variables...**.
7. In the Select Variable dialog, you will see a number of variables you can pass as arguments to the program specified in Location. Select **resource_loc** and click **OK**.
8. When this buildfile is run, the absolute path of the resource selected in the workbench will be passed to the jar utility in the position specified.
9. Click **Run**.
10. Notice that the buildfile sends the jar utility output to the Console view.

11. Select a different .jar file in your workspace.
12. Click the External Tools button in the toolbar. Notice the contents of this jar are sent to the Console view as well. Now you have a quick and easy way to see the output of the jar utility for any .jar file in your workspace.

This example has only scratched the surface of what you can do with external tools. The important things to remember are that you can create an external tool for anything you can run on your system, and that you can pass arguments to the external tool related to the current workbench selection. In many cases, this allows you to loosely integrate tools that do not have corresponding Eclipse plug−ins.

Workbench

The term *Workbench* refers to the desktop development environment. The Workbench aims to achieve seamless tool integration and controlled openness by providing a common paradigm for the creation, management, and navigation of workspace resources.

Each Workbench window contains one or more perspectives. Perspectives contain views and editors and control what appears in certain menus and tool bars. More than one Workbench window can exist on the desktop at any given time.

Related concepts

Features
Resources
Perspectives
Views
Editors
CVS Repositories

Related tasks

Installing new features with the update manager
Opening perspectives
Opening views
Switching between perspectives
Configuring perspectives

Related reference

Toolbar buttons

Features

On disk, an Eclipse based product is structured as a collection of *plug−ins*. Each plug−in contains the code that provides some of the product's functionality. The code and other files for a plug−in are installed on the local computer, and get activated automatically as required. A product's plug−ins are grouped together into features. A *feature* is the smallest unit of separately downloadable and installable functionality. (The notion of a feature was new for Eclipse 2.0; it replaced the similar notion of component in Eclipse 1.0.)

The fundamentally modular nature of the Eclipse platform makes it easy to install additional features and plug−ins into an Eclipse based product, and to update the product's existing features and plug−ins. You can do this either by using traditional native installers running separately from Eclipse, or by using the Eclipse platform's own update manager. The Eclipse *update manager* can be used to discover, download, and install updated features and plug−ins from special web based Eclipse update sites.

The basic underlying mechanism of the update manager is simple: the files for a feature or plug−in are always stored in a sub−directory whose name includes a version identifier (e.g., "2.0.0"). Different versions of a feature or plug−in are always given different version identifiers, thereby ensuring that multiple versions of the same feature or plug−in can co−exist on disk. This means that installing or updating features and plug−ins requires adding more files, but never requires deleting or overwriting existing files. Once the files are installed on the local computer, the new feature and plug−in versions are available to be configured. The same installed base of files is therefore capable of supporting many different *configurations* simultaneously; installing and upgrading an existing product is reduced to formulating a configuration that is incrementally newer than the current one. Important configurations can be saved and restored to active service in the event of an unsuccessful upgrade.

Large Eclipse based products can organize their features into trees starting from the root feature that represents the entire product. This root feature then includes smaller units of functionality all the way down to leaf features that list one or more plug−ins and fragments. The capability to group feature hierarchically allows products to be stacked using a 'Russian doll' approach − a large product can build on top of a smaller one by including it and adding more features.

Some included features may be useful add−ons but not vital to the proper functioning of the overall product. Feature providers can elect to mark them as **optional**. When installing optional features, users are provided with a choice of whether they want them or not. If not installed right away, optional features can be added at a later date.

The **About** option on the **Help** menu provides information about installed features and plug−ins. The **Software Updates** submenu on the **Help** menu groups together items for updating existing features, and for finding, downloading, and installing new features.

Related concepts

Workbench

Related tasks

Inspecting the current configuration
Installing new features with the update manager
Configuration operations
Updating features with the update manager
Update policy

Restoring a saved configuration

Inspecting the current configuration

The About Eclipse Platform dialog (**Help > About Eclipse Platform**) shows most of the important information about the features and plug−in configured in the product. To find out further detailed information about the current configuration:

1. Open the update configuration manager by clicking **Help > Software Updates > Manage Configuration**. This opens the Product Configuration dialog.
2. In the Product Configuration dialog expand the root node (current configuration). The first tier of items are locations on the local computer where the files for features and plug−ins are stored. For a typical product, all features are installed in a single directory sub−tree. However, a feature that is installed as an extension of another product are usually located in a separate directory sub−tree.
3. Expand the location to see the feature versions installed there.
4. The nesting of the features indicates which child features are included in a given parent feature. Nested features act as a unit.
5. Select a feature version and click **Show Properties** to open the Properties dialog which presents more detailed information about the selected feature.
6. Click **Show Disabled Features** filter on the toolbar of the Product Configuration dialog. Any feature versions that are disabled are now shown (you might not have any). Disabled feature versions are ones that are installed on the local computer but excluded from the current configuration. At most one version of any given feature can be present in a given configuration; usually the ones with the lower version numbers will be in a disabled state.
7. In the Properties dialog of a feature, click **Status** to find out whether the selected feature is configured exactly as packaged. The status will show when a feature's plug−ins are present at different versions, or disabled due to missing prerequisite plug−ins that are expected to be supplied by some other feature.

In this dialog you can revert to a previous install configuration, view the installation history, or even add another eclipse product extension. A product extension is a local folder that has a similar file layout as the eclipse installation, but with a .eclipseextension marker (instead of .eclipseproduct). If you add a product extension, make sure there are no duplicate features in the extension you add.

When the install wizard is used to install features in a user selected location, that location automatically becomes an eclipse product extension.

Related concepts

Features

Related tasks

Installing new features with the update manager
Updating features with the update manager
Update Policy

Installing new features with the update manager

To locate and install a new feature into a product (ordinarily requires web access):

1. Open the Install Wizard by clicking **Help > Software Updates > Find and Install**. This opens the wizard.
2. Select the second button, "**Search for new features to install**" and click **Next**.
3. Create a bookmark for an update site where Eclipse features and plug−ins are published. In the sites to search list, select **Add Update Site** to add a remote site, or **Add Local Site** if the site is available on a local drive (including a CD), or **Add Archived Site**, when the site is available locally but is packaged as a jar or zip file.
4. In the add site dialog, give the site a name such as "CompanyA" and enter the URL such as "http://companyA.example.com/eclipseupdates".
5. After adding the site, expand it to the categories of feature versions available at that update site. This will contact the web site to discover what features are available.
6. Select the categories you want to search and click **Next**.
7. Wait for the search to finish and selecting the features to be added. You can view the description or more detailed properties for any feature by selecting the feature and pressing the "Properties" button.
8. Once you're decided which features to install, click **Next**.
9. Carefully review the license agreements for the features. If the terms of all these licenses are acceptable, check "I accept the terms in the license agreements". Do not proceed to download the features if the license terms are not acceptable.
10. If a feature selected for install include optional features, a page will show up allowing you to select whether you want them installed or not. Optional features typically carry functionality that is not essential for proper functioning of the main feature.
11. The Install Location page controls where the new feature's files are to be installed on the local computer. Select the directory into which the product is installed and hit **Next**. (If the product is installed in a directory to which you do not have write access, you should contact your system administrator and get them to install this feature so that it will be generally available. The other option is to click **Add** and point to a directory to which you do have write access.
12. Feature versions can be digitally signed by the company that provides them. This allows you to verify more easily that the features and plug−ins that are about to be downloaded and installed are coming from a trusted supplier.

    Warning: Because of the possibility of harmful or even malicious plug−ins, you should only download features from parties that you trust.

    Click **Install** to allow the downloading and installing to proceed.
13. Once the new feature and plug−ins have been downloaded successfully and their files installed into the product on the local computer, a new configuration that incorporates these features and plug−ins will be formulated. Click **Yes** when asked to exit and restart the workbench for the changes to take effect. To add other new features at the same time before restarting, click **No** and repeat. If this was a new feature and can be dynamically installed into the current configuration, you will also be presented with a **"Apply Now"** button. This will not restart the platform, but configures the feature and the plug−ins into the current configuration.

Related concepts

Features

Related tasks

Inspecting the current configuration
Updating features with the update manager
Restoring a saved configuration

Updating features with the update manager

To check to see whether there are updates for a product's existing features (requires Internet access):

1. Click **Help > Software Updates >Find and Install** and select the first choice (search for updates). This will contact the Web sites associated with the product's features to discover what versions of those features are available. The potential upgrades are presented in on the next page.
2. Select the feature versions that you wish to upgrade, and click **Next**.
3. Carefully review the license agreements for the upgraded features. If the terms of all these licenses are acceptable, check "I accept the terms in the license agreements." Do not proceed to download the features if the license terms are not acceptable.
4. Feature versions can be digitally signed by the company that provides them. This allows the user to verify more easily that the features and plug−ins that are about to be downloaded and installed are coming from a trusted supplier.

   Warning: Because of the possibility of harmful or even malicious plug−ins, you should only download features from parties that you trust.

   Click **Install** to allow the downloading and installing to proceed.
5. Once all the features and plug−ins have been downloaded successfully and their files installed into the product on the local computer, a new configuration that incorporates these features and plug−ins will be formulated. Click **Yes** when asked to exit and restart the Workbench for the changes to take effect.

Related concepts

Features

Related tasks

Inspecting the current configuration
Installing new features with the update manager
Enable, disable or uninstall a feature
Update policy control
Automatic update scheduler
Restoring a saved configuration

Configuration operations: enable, disable, or uninstall a feature

Current configuration can be browsed, but it can also be operated on to enable/disable/uninstall features, to revert to previous configurations, etc. Open the configuration manager by clicking **Help > Software Updates > Manage Configuration**.

1. **Disable a feature**: select the feature and click on the "Disable" feature action in the right pane of the dialog or in the context menu. The action is available only when the feature is currently enabled, and the feature is either an optional feature or a root feature (not included by other features).
2. **Enable a feature**: Turn on disabled features filtering from the dialog toolbar, then select a disabled optional or root feature and click on the "Enable" feature action in the right pane of the dialog or in the context menu.
3. **Uninstall** a feature: features installed by you using the update manager can be uninstalled, provided they are already disabled, or that they are optional or root features. If the feature is disabled, make sure you turn on the disabled features filtering from the dialog toolbar. Select the feature and click on the "Uninstall" feature action in the right page of the dialog or in the context menu.

Related concepts

Features

Related tasks

Installing new features with the update manager
Updating features with the update manager
Saving a configuration

Eclipse Update Policy Control

Eclipse Update allows users to search for updates to the currently installed features. For each installed feature, Update uses the embedded URL to connect to the remote server and search for new versions. If there are updates, Eclipse allows users to initiate the install procedure. After downloading, installing and restarting the platform, new feature version is ready for use.

In companies with many users of the same Eclipse−based product (typically a commercial one), several problems can arise from this model:

1. Updates for very large products (e.g. 500+ plug−ins) are also large. I/T support teams may not like the idea of hundreds of developers individually downloading 500MEG updates to their individual machines. In addition to the bandwidth hit, such a large download request may fail, leading to repeated attempts and increased developers' downtime.
2. Some companies explicitly don't want the developers downloading updates directly from the Internet. For example, they can set up a local support team that may not be ready to handle requests related to the version of the product already available from the provider's update site. They may want to restrict updates and fixes to the internally approved list. Ideally, they would do that by setting up 'proxy' update sites on the LAN (behind the firewall).
3. Once updates are set in the proxy sites as above, administrators need a way of letting users know that updates are available.

2. Update policy to the rescue

2.1 Support for creating local (proxy) update sites

First step for a product administrator would be to set up a local Eclipse update site on a server connected to the company's LAN (behind the firewall). The update site would be a subset of the product's update site on the Internet because it would contain only features and plug−ins related to the updates that the company wants applied at the moment. Technically, this site would be a regular Eclipse update site with site.xml, feature and plug−in archives.

Administrators would construct this site in two ways:

1. Product support teams would make a zip file of the update site readily available for this particular purpose. Administrators would simply need to download the zip file from the product support web page using the tool of their choice and unzip it in the local server. This approach is useful for very large zip files that require modern restartable downloading managers (those that can pick up where they left off in case of the connection problems).
2. Eclipse Update provides a tool to mirror remote update sites entirely or allow administrators to select updates and fixes to download. This mirroring capability would be fully automated and would greatly simplify administrator's task but it relies on Update network connection support.

2.2 Common update policy control

Since features have the update site URL embedded in the manifest, they are unaware of the local update sites set up by the administrators. It is therefore important to provide **redirection capability**. This and other update policy settings can be set for an Eclipse product by creating an update policy file and configuring Update to use that file when searching.

The file in question uses XML format and can have any name. The file can be set in **Preferences>Install/Update** in the **Update Policy** field. The text field is empty by default: users may set the URL of the update policy file. The file is managed by the local administrator and is shared for all the product installations. Sharing can be achieved in two ways:

- If users install the product: users are told to open the preference page and enter the provided URL
- If administrators install the product: administrators edit the file 'plugin_customization.ini' in the primary product feature and set the default value of the 'updatePolicyFile' property as follows:

```
org.eclipse.update.core/updatePolicyFile = <URL value>
```

  This will cause all the installations to have this file set by default.

The policy file must conform to the following DTD:

```
<?xml encoding="ISO-8859-1"?>

<!ELEMENT update-policy (url-map)*>
<!ATTLIST update-policy
>

<!ELEMENT url-map EMPTY>
<!ATTLIST url-map
    pattern    CDATA #REQUIRED
    url        CDATA #REQUIRED
>
```

url-map

- **pattern** - a string that represents prefix of a feature ID (up to and including a complete ID). A value of "*" matches all the features.
- **url** - a URL of the alternative update site that should be used if the feature ID begins with the pattern. If the string is empty, features matching pattern will not be updateable.

This element is used to override Update URLs embedded in feature manifests. When looking for new updates, Eclipse search will check the update policy (if present) and check if **url-map** for the matching feature prefix is specified. If a match is found, the mapped URL will be used **instead** of the embedded one. This way, administrators can configure Eclipse products to search for updates in the local server behind the firewall. Meanwhile, third-party features installed by Eclipse Update will continue to be updated using the default mechanism because they will not find matches in the policy.

Several **url-map** elements may exist in the file. Feature prefixes can be chosen to be less or more specific. For example, to redirect all Eclipse updates, the pattern attribute would be `"org.eclipse"`. Similarly, it is possible to use a complete feature ID as a pattern if redirection is required on a per-feature basis.

Patterns in the file may be chosen to progressively narrow the potential matches. This may result in multiple matches for a given feature. In this case, the **match with a longest pattern** will be used. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<update-policy>
```

```
        <url-map pattern="org.eclipse" url="URL1"/>
        <url-map pattern="org.eclipse.jdt" url="URL2"/>
</update-policy>
```

In the case above, all Eclipse features will be updated from URL1, except `org.eclipse.jdt` that will use URL2.

Update policy files do not contain translatable strings and therefore do not require special NL handling. In general, the files should use UTF-8 encoding.

2.3 Automatic discovery of updates

The third part of the overall solution is covered by another topic but is mentioned here because it is an integral part of the solution. Automatic updates will allow Eclipse to run update search on a specified schedule (on each startup (the default), once a day, once a week etc.).

3. Summary

Here is the complete sequence of steps that comprise the solution:

1. Administrator allocates a server on the company LAN for hosting local product updates. Initially it contains no update sites. The machine must have an HTTP server running.
2. Administrator sets up an update policy file on that server and instructs all users to set the update policy preference the provided URL.
3. As the product provider ships updates and fixes on their update sites, administrator downloads supported updates onto the local server.
4. Automatic update executed at the scheduled frequency when the client's product is up picks up the local updates and notifies the user
5. User chooses to install the discovered updates

Related tasks

Automatic Update Scheduler

Automatic Update Scheduler

Eclipse Update is able to search for updates to the installed features on a periodic basis without user assistance. The way this search is performed will be configurable by the user via the preference page. Users will be able to configure scheduling and downloading options.

Scheduling options

Each time an Eclipse−based product starts, update scheduler plug−in will activate based on the chosen scheduling. Users will be able to choose between:

- No automatic search
- On each startup (default)
- Every day at a specific time (i.e. 3:00PM) *
- On a scheduled day of the week at a specific time (i.e. Monday 8:00AM) *

* assuming the application is active. If the application is not active at a scheduled time and the search is past due, it will immediately start on the next startup

Downloading options

Scheduling options define occurrence of the automated search but do not define what is actually happening when the search is due. Downloading options control this behavior (this set of options is disabled if automatic search is turned off). Users can choose between:

- Search for updates and notify when they are available (default)
- Download new updates automatically and notify when ready to install them

In both cases, Update scheduler will create and initiate a search job that will execute in the background when it is due based on the scheduling options. There will be no messages if no updates were found. If there are updates, the behavior will depend on the selected downloading option.

Search and notify

If search only is selected, a message box will open, notifying user that new updates have been found:

New updates have been found for this product. Do you want to download them?

[Yes] [No]

If 'Yes' is pressed, a Update wizard will open, listing detected updates. The same wizard is used when search for updates has been manually initiated using Help>Software Updates>Find and Install.

If 'No' is pressed, dialog will close and the updates ignored.

Download new updates

If the second option has been selected (download updates automatically), the search job will be scheduled as before, but when new updates are found, the download will commence immediately. When all the features

have been successfully downloaded into the temp. space, a different message box will show up:

New updates have been found and downloaded. Do you want to install them?

[Yes] [No]

If 'No' is pressed, dialog will close and all the data in the temporary space will be deleted. If 'Yes' is pressed, installation will commence (only a progress monitor will be shown − no wizards).

If you choose not to install the downloaded features, but later launch the Find and Install wizard for updating, the downloaded files will be reused (they're cached), unless there are newer versions on the server or you have restarted eclipse.

Related tasks

Updating features with the update manager

Restoring a saved configuration

A previously saved configuration (or any configuration recorded in the history) can be restored, thereby backing out of the results of an unsuccessful upgrade. Restoring a saved configuration does not delete the files for feature and plug−in versions from the local computer; it merely ignores them and acts as it did when they were not present. To restore a saved configuration to be the current configuration:

1. Select **Help > Software Updates > Manage Configuration...** to open the Manage Configuration dialog.
2. In the Product Configuration view, click the Installation History icon and select the desired configuration. Click **Restore**.
3. Click **Yes** when asked to exit and restart the Workbench for the changes to take effect.
4. Expand the Current Configuration (and children) to confirm that the current configuration now matches the configuration saved earlier.

Related concepts

Features

Related tasks

Inspecting the current configuration
Update policy control

Resources

Resources is a collective term for the projects, folders, and files that exist in the Workbench. The navigation views provide a hierarchical view of resources and allows you to open them for editing. Other tools may display and handle these resources differently.

There are three basic types of resources that exist in the Workbench:

Files
: Comparable to files as you see them in the file system.

Folders
: Comparable to directories on a file system. In the Workbench, folders are contained in projects or other folders. Folders can contain files and other folders.

Projects
: Contain folders and files. Projects are used for builds, version management, sharing, and resource organization. Like folders, projects map to directories in the file system. (When you create a project, you specify a location for it in the file system.)
A project is either open or closed. When a project is closed, it cannot be changed in the Workbench. The resources of a closed project will not appear in the Workbench, but the resources still reside on the local file system. Closed projects require less memory. Since they are not examined during builds, closing a project can improve build time.
When a project is open, the structure of the project can be changed and you will see the contents.

Folders and files directly below projects can be linked to locations in the file system outside of the project's location. These special folders and files are called linked resources.

Different tools that plug into the Workbench use their own specialized types of projects, folders, and files.

Related concepts

Workbench
Navigator view
Resource hierarchies
Linked resources
Builds

Related tasks

Importing resources into the Workbench
Creating a project
Creating a folder
Creating a file
Creating linked resources
Closing projects
Viewing resource properties
Finding a resource quickly
Copying resources
Moving resources
Comparing resources

Linked resources

Linked resources are files and folders that are stored in locations in the file system outside of the project's location. These special resources must have a project as their parent resource. Linked folders and files can be used to add resources to your project that for some reason must be stored in a certain place outside of your project. For example, a linked folder can be used to store build output separately from your source files.

You can even use linked resources to overlap other resources in the workspace, so resources from one project can appear in another project. If you do want to have overlapping resources in your workspace, do so with caution. Keep in mind that this means changing a resource in one place will cause simultaneous changes in the duplicate resource. Deleting one duplicate resource will delete both!

Special rules apply when manipulating linked resources. Since they must be located directly below a project, you cannot copy or move linked resources into other folders. Deleting a linked resource will *not* cause the corresponding resource in the file system to be deleted. However, deleting child resources of linked folders *will* cause them to be removed from the file system.

Some plug−ins built on top of the Eclipse platform are not compatible with linked resources. If this is the case, you can completely disable the linked resource feature to prevent them from being created in your workspace. Linked resources can be disabled from the **General > Workspace > Linked Resources** preference page. Certain types of projects or team repository providers may also disallow linked resources from being created in some projects.

Related concepts

Workbench
Navigator view
Resources
Resource hierarchies

Related tasks

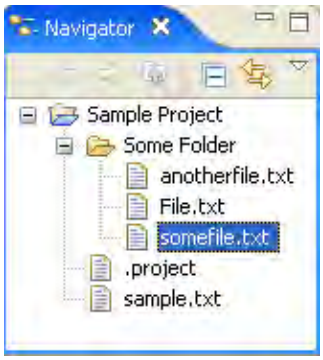
Creating linked resources
Deleting resources
Viewing resource properties

Related reference

Linked resources

Navigator view

The Navigator view provides a hierarchical view of the resources in the Workbench. From here, you can open files for editing or select resources for operations such as exporting.



Right−click on any resource in the Navigator view to open a pop−up menu that allows you to perform operations such as copying, moving, creating new resources, comparing resources with each other, or performing team operations. To see a description of what each menu item does, move selection highlight to that menu item and press the context−sensitive help key (e.g., F1 on Microsoft Windows).

By default, the Navigator view is included in the Resources perspective. To add it to the current perspective, click **Window > Show View > Navigator**.

Toolbar

The toolbar of the Navigator view contains the following buttons:

Back
: Displays the hierarchy that was displayed immediately prior to the current display.

Forward
: Displays the hierarchy that was displayed immediately after the current display.

Up
: Displays the hierarchy of the parent of the current highest level resource.

Collapse All
: Collapses the tree expansion state of all resources in the view.

Link with Editor
: Toggles whether the Navigator view selection is linked to the active editor. When this option is selected, changing the active editor will automatically update the Navigator selection to the resource being edited.

Menu
: Provides menu items that allow you to sort or filter the contents of the Navigator view as well as select a working set.

Icons

The following icons can appear in the Navigator view.

Icon	~~Description~~

	Project (open)
	Folder
	File

Related concepts

Resources
Views

Related tasks

Narrowing the scope of the Navigator view
Sorting resources in the Navigator view
Showing or hiding files in the Navigator view
Opening views
Moving and docking views
Creating fast views

Related reference

Navigator view

Views

Views support editors and provide alternative presentations as well as ways to navigate the information in your Workbench. For example, the Navigator and other navigation views display projects and other resources that you are working with.

Views also have their own menus. To open the menu for a view, click the icon at the left end of the view's title bar. Some views also have their own toolbars. The actions represented by buttons on view toolbars only affect the items within that view.

A view might appear by itself, or stacked with other views in a tabbed notebook. You can change the layout of a perspective by opening and closing views and by docking them in different positions in the Workbench window.

Related concepts

Perspectives
Fast views
Editors

Related tasks

Opening views
Moving and docking views
Creating fast views
Maximizing a view or editor
Saving a user defined perspective
Resetting perspectives

Perspectives

Each Workbench window contains one or more perspectives. A perspective defines the initial set and layout of views in the Workbench window. Within the window, each perspective shares the same set of editors. Each perspective provides a set of functionality aimed at accomplishing a specific type of task or works with specific types of resources. For example, the Java perspective combines views that you would commonly use while editing Java source files, while the Debug perspective contains the views that you would use while debugging Java programs. As you work in the Workbench, you will probably switch perspectives frequently.

Perspectives control what appears in certain menus and toolbars. They define visible *action sets*, which you can change to customize a perspective. You can save a perspective that you build in this manner, making your own custom perspective that you can open again later.

You can set your General preferences to open perspectives in the same window or in a new window.

Related concepts

Workbench
Editors
Views

Related tasks

Opening perspectives
Opening views
Changing where perspectives open
Specifying the default perspective
Switching between perspectives
Configuring perspectives
Saving a user defined perspective
Resetting perspectives

Editors

Most perspectives in the Workbench are comprised of an editor area and one or more views.

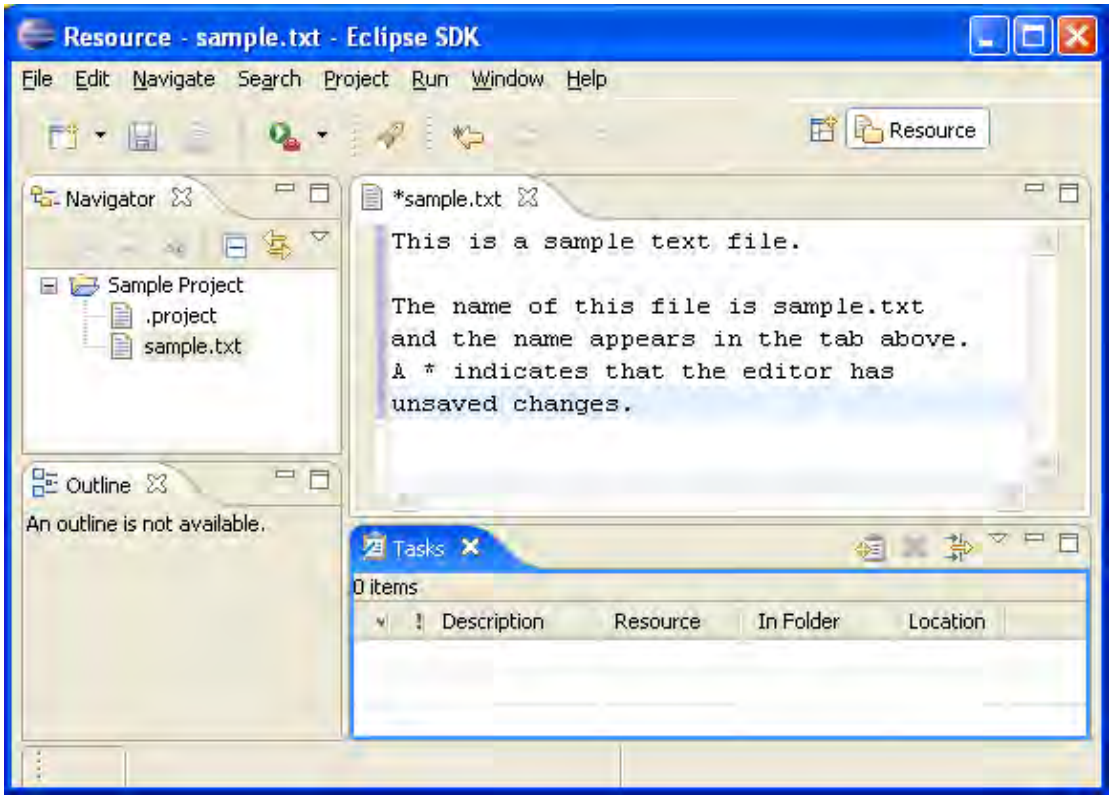
You can associate different editors with different types of files. For example, when you open a file for editing by double−clicking it in one of the navigation views, the associated editor opens in the Workbench. If there is no associated editor for a resource, the Workbench attempts to launch an *external editor* outside the Workbench. (On Windows, the Workbench will first attempt to launch the editor in place as an OLE document. This type of editor is referred to as an *embedded editor*. For example, if you have a .doc file in the Workbench and Microsoft Word is registered as the editor for .doc files in your operating system, then opening the file will launch Word as an OLE document within the Workbench editor area. The Workbench menu bar and toolbar will be updated with options for Microsoft Word.)

Any number of editors can be open at once, but only one can be active at a time. The main menu bar and toolbar for the Workbench window contain operations that are applicable to the active editor.

Tabs in the editor area indicate the names of resources that are currently open for editing. An asterisk (*) indicates that an editor has unsaved changes.

By default, editors are stacked in the editor area, but you can choose to tile them in order to view source files simultaneously.

Here is an example of a text editor in the Workbench:



The gray border at the left margin of the editor area may contain icons that flag errors, warnings, or problems detected by the system. Icons also appear if you have created bookmarks, added breakpoints for debugging, or

recorded notes in the Tasks view. You can view details for any icons in the left margin of the editor by moving the mouse cursor over them.

Related concepts

Workbench
External editors
Bookmarks
Tasks view
Navigator view

Related tasks

Opening files for editing
Associating editors with file types
Editing files outside the Workbench

External editors

Sometimes you may need to use an external program to edit a file in the Workbench. This can occur, for example, when the Workbench has no editor for that file type.

The external program will be used if that program is registered as the system default editor for that file type and no other editor is registered for that file type in the Workbench. For example, on most systems the default "editor" for HTML files is a Web browser. If there is no other editor associated with .htm and .html files in the Workbench, then opening an HTML file from the Workbench would cause the file to be opened in a separate browser session.

Related concepts

Editors

Related tasks

Opening files for editing
Associating editors with file types
Editing files outside the Workbench

Opening files for editing

You can launch an editor for a given file in several ways.

- By right−clicking the file in one of the navigation views and then selecting **Open** from the pop−up menu.
- By double−clicking the file in one of the navigation views.
- By double−clicking a bookmark that is associated with that file, in the Bookmarks view.
- By double−clicking an error or warning, or task record that is associated with that file, in the Problems view.

All of the above alternatives open the file in the default editor for that type of file. To open it in a different editor, select **Open With** from the file's pop−up menu.

Related concepts

Editors
External editors

Related tasks

Associating editors with file types
Editing files outside the Workbench
Linking the Navigator view to the active editor
Tiling editors
Changing the placement of editor tabs
Comparing resources

Associating editors with file types

To associate editors with various file types in the Workbench:

1. From the main menu, select **Window > Preferences**.
2. In the Preferences window, select the **General > Editors > File Associations** page .
3. Select the file type from the **File types** list, or click **Add** to add a type that is not already on the list.
4. In the **Associated editors** list, select the editor that you want to associate with that file type. To add an editor to the list:
    a. Click **Add**. The Editor Selection dialog box opens.
    b. Select **Internal Editors** or **External Programs**, depending on whether the editor that you want was built for the Workbench or runs outside the Workbench.
    c. If you select **External Programs**, you can click the Browse button to browse the file system.
    d. Select the editor from the list and click **OK**.
5. Click **OK** to finish associating the editor with the selected file type.

When you associate an internal editor with a file type, that editor opens in the editor area of the Workbench. For example, if you double−click a file in the Navigator or an entry in the Bookmarks or Tasks view it opens in the editor area.

WIN Editors that support OLE document mode can also run in the editor area of the Workbench.

Tip: You can choose to override your default editor selections by selecting **Open With** from the pop−up menu for any resource in one of the navigation views.

Related concepts

Editors
External editors

Related tasks

Opening files for editing
Editing files outside the Workbench
Linking the Navigator view to the active editor

Editing files outside the Workbench

To edit a Workbench resource outside the Workbench:

1. Navigate in the file system to the Workbench's installation directory. Go into the workspace directory and open the file that you want to edit with the external editor.
2. Edit the file as needed. Save and close it as usual.
3. *Important*: Go back to the Workbench, right−click the edited file in one of the navigation views, and select **Refresh** from the pop−up menu. The Workbench will perform any necessary build or update operations to process the changes that you made outside the Workbench.

Tip: If you work with external editors regularly, you may want to enable auto−refresh.This can be done by going to the preferences dialog (by choosing the ***Window >Preferences*** menu item), drilling down to the ***General > Workspace*** preference page, and checking the ***Refresh automatically*** option. When this option is enabled, any external changes will be automatically discovered by the Workbench. Depending on the platform this may not happen immediately.

Related concepts

Editors
External editors

Related tasks

Opening files for editing
Associating editors with file types

Linking the Navigator view to the active editor

When you have multiple files open for editing, you can configure one of the navigation views to automatically bring an open file to the foreground (make its editor session the active editor) every time you select that open file in one of the navigation views. There are two ways to set this.

1. From the Navigator menu, select **Link With Editor.**
2. Click on the Link With Editor icon  on one of the navigation bars.

Related concepts

Editors

Related tasks

Opening files for editing
Associating editors with file types

Tiling editors

The Workbench allows you to have multiple files open in multiple editors. Unlike views, editors cannot be dragged outside the Workbench to create new windows. However, you can tile editor sessions within the editor area, in order to view source files side by side.

1. With two or more files open in the editor area, select one of the editor tabs.
2. Holding down the left mouse button, drag that editor over the left, right, top or bottom border of the editor area. Notice that the mouse pointer changes to a "drop cursor" that indicates where the editor session will be moved when you release the mouse button.
3. (Optional) Drag the borders of the editor area or each editor, to resize as desired.

This is a similar operation to moving and docking views inside the Workbench, except that all editor sessions must be contained within the editor area.

Related concepts

Editors

Related tasks

Opening files for editing
Changing the placement of the tabs
Comparing resources

Changing the placement of the tabs

You can change the placement of the tabs. The tabs for stacked views or editors can appear at the top or bottom of the area which contains them.

1. On the main menu bar, click **Window > Preferences**.
2. Expand the **General** category and select **Appearance**.
3. Select from the choices displayed in the **Editor tab positions** group or **View tab positions** group to control whether you want the tabs at the top or the bottom.
4. Click **Apply** or **OK**.
5. The tabs will immediately move to their new locations.

Related concepts

Views
Editors

Related tasks

Opening files for editing
Tiling editors

Comparing resources

When a comparison is performed, comparison editors appear in the editor area. The differences between files are highlighted in the comparison editors, allowing you to browse and copy changes between the compared resources.

To compare resources:

1. Select one or more resources in one of the navigation views.
2. From the resource's pop−up menu, select **Compare With.**

A tool for viewing differences is opened in the editor area.

Here are some of the comparisons that you can perform.

Compare With > Latest from HEAD
: compares the selected resource with the version of the same resource that is currently committed to the active branch. This option is available for CVS. This ,or a similar option, may or may not be available for other repository types.

Compare With > Another Branch or Version
: compares the selected resource with a version that has been committed to the repository or to the latest version in a particular branch. This requires you to choose a version of the resource or a branch from a list. This option is available for CVS. This, or a similar option, may or may not be available for other repository types.

Compare With > Each Other
: compares two or three selected resources with each other

Compare With > Revision
: compares the selected resource with a version of the same resource that was committed to the active branch. This requires you to choose a revision of the resource from a list. This option is available for CVS. This, or a similar option, may or may not be available for other repository types.

Compare With > Local History
: compares the selected resource to one that is in the local history, which is maintained when you save changes. (*Tip:* to configure the size and depth of your local history, look under **Window > Preferences > General > Workspace > Local History**.)

Tip: You can customize the behavior of the comparison editor by selecting **Window > Preferences > General > Compare/Patch** from the main menu bar. You can select to "lock scroll" the pane contents, as well as setting options for showing conflicts with other team members and changing the default font.

Related concepts

Synchronizing with a CVS repository
Three way comparisons
Local history

Related tasks

Synchronizing with the repository
Merging changes in the Compare editor
Resolving conflicts
Setting preferences for comparing files

Comparing resources with repository versions
Understanding the comparison
Working with patches
Tiling editors

Related reference

Compare editor
CVS Synchronize view

Synchronizing with a CVS repository

In the CVS team programming environment, there are two distinct processes involved in synchronizing resources: *updating* with the latest changes from a branch and *committing* to the branch.

When you make changes in the Workbench, the resources are saved locally. Eventually you will want to commit your changes to the branch so others can have access to them. Meanwhile, others may have committed changes to the branch. You will want to update your Workbench resources with their changes.

Important!: It is preferable to update *before* committing, in case there are conflicts with the resources in your Workbench and the resources currently in the branch.

The synchronize view contains filters to control whether you want to view only i*ncoming changes* or *outgoing changes*. Incoming changes come from the branch. If accepted, they will update the Workbench resource to the latest version currently committed into the branch. Outgoing changes come from the Workbench. If committed, they will change the branch resources to match those currently present in the Workbench.

Regardless of which mode (filter) you select, the Synchronize view always shows you conflicts that arise when you have locally modified a resource for which a more recent version is available in the branch. In this situation you can choose to do one of three things: update the resource from the branch, commit your version of the resource to the branch, or merge your work with the changes in the branch resource. Typically you will want to merge, as the other two options will result in loss of work.

Related concepts

Team programming with CVS
Branches

Related tasks

Synchronizing with the repository
Updating
Resolving conflicts
Merging from a branch
Committing

Related reference

CVS
CVS Synchronize view

Team programming with CVS

In the Concurrent Versions System (CVS) team programming environment, team members do all of their work in their own Workbenches, isolated from others. Eventually they will want to share their work. They do this via a CVS Repository.

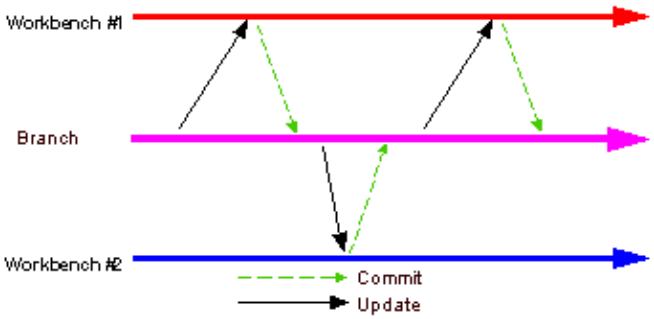
Branches

CVS uses a branch model to support multiple courses of work that are somewhat isolated from each other but still highly interdependent. Branches are where a development team shares and integrates ongoing work. A branch can be thought of as a shared workspace that is updated by team members as they make changes to the project. This model allows individuals to work on a CVS team project, share their work with others as changes are made, and access the work of others as the project evolves. A special branch, referred to as HEAD, represents the main course of work in the repository (HEAD is often referred to as the trunk).

Sharing work

As team members produce new work, they share this work by *committing* those changes to the branch. Similarly, when they wish to get the latest available work, they *update* their local workspaces to the changes on the branch. Thus the branch is constantly changing, moving forward as team members submit new work.

The branch effectively represents the current state of the project. At any point a team member can update their workspaces from the branch and know they are up to date.



Optimistic team model

CVS provides two important features required for working in a team:

- A history of the work submitted by the team
- A way to coordinate and integrate this work

Maintaining history is important so that one can compare the current work against previous drafts, revert to older work that is better, and so on. Coordinating the work is critical so that there exists one definition of the current project state, containing the integrated work of the team. This coordination is provided via the branch model.

An optimistic model is one where any member of the team can make changes to any resource he or she has access to. Because two team members can commit to the branch changes to the same resource, conflicts can

occur and must be dealt with. This model is termed *optimistic* because it is assumed that conflicts are rare.

Recommended work flow

Usually resources do not exist in isolation, they typically contain implicit or explicit dependencies on other resources. For example, Web pages have links to other Web pages, and source code has references to artifacts described in other source code resources. No resource is an island.

As resources are committed to the branch, these dependencies can be affected. Ensuring the integrity of the dependencies is important because the branch represents the current project state: at any point a team member could take the branch contents as a basis for new work.

The ideal work flow therefore is one in which the branch integrity is preserved.

Ideal flow enumerated

The ideal work flow proceeds as follows:

1. Start fresh. Before starting work, update the resources in the workspace with the current branch state. If you are sure that you have no local work that you care about, the fastest way to get caught up is to select the projects you are interested in from the branch (or HEAD) and select **Checkout** (or **Replace with > Latest from Repository** if the projects already exist locally). This will overwrite your local resources with those from the branch.
2. Make changes. Work locally in your Workbench, creating new resources, modifying existing ones, saving locally as you go.
3. Synchronize. When you are ready to commit your work, synchronize with the repository.

    a. Update. Examine incoming changes and add them to your local Workbench. This allows you to determine if there are changes which might affect the integrity of what you are about to commit. Resolve conflicts. Retest, run integrity checkers (for example, check for broken hypertext links, ensure your code compiles, and so on).
    b. Commit. Now that you are confident that your changes are well integrated with the latest branch contents, commit your changes to the branch. To be prudent, you may repeat the previous step if there are new incoming changes.

Of course this is an *ideal* workflow. Under certain conditions you may be confident that the incoming changes do not affect you and choose to commit without updating. However, in general team members should make an effort to follow a flow similar to the above in order to ensure that the branch integrity is not accidentally compromised.

You can find more information on CVS at https://www.cvshome.org.

Related concepts

CVS Repositories
Branches
Versions
Synchronizing with a CVS repository

Related tasks

Creating a CVS repository location
Checking out a project from a CVS repository
Replacing resources in the Workbench
Sharing a new project using CVS
Synchronizing with the repository
Updating
Resolving conflicts
Merging from a branch
Committing

Related reference

CVS

CVS Repositories

A Concurrent Versions System (CVS) repository is a persistent data store that coordinates multi−user access to projects and their contents. Projects in a repository can be of two forms: immutable (a project version) or modifiable (a project in a branch). Communication between the repository and Workbench clients is possible over local or wide area networks. Currently the Workbench supports two internal authentication protocols for CVS: pserver and ssh. Authentication protocols provided by external tools can also be used by CVS.

You can find more information on CVS at https://www.cvshome.org.

Related concepts

Team programming with CVS
Branches
Versions
Local history

Related tasks

Creating a CVS repository location

Related reference

CVS
CVS Repositories view

Branches

In CVS, teams share and integrate their ongoing work in *branches*. Think of a branch as a shared work area that can be updated at any time by team members. In this way, individuals can work on a team project, share their work with others on the team, and access the work of others during all stages of the project. The branch effectively represents the current shared state of the project.

Resources can be changed in the Workbench without affecting the branch. Individuals must explicitly provide their changed resources to the branch.

Every CVS repository has at least one branch, referred to as HEAD. Under certain conditions, more than one branch may exist in a repository. For example, one branch may be for ongoing work, and another branch may be for maintenance work.

As you make changes locally in your Workbench, you are working alone. When you are ready to make your local resource changes available to other team members, you'll need to *commit* your work to the branch. All such changes are classified as outgoing changes when you do a synchronization.

Ideally, you should update your local workspace with any changes others have made in a branch before committing to it. This ensures that you have the very latest work from the other team members. After you have updated from the branch, merged any conflicting changes in your local Workbench, and tested your changes locally, you can more easily commit your Workbench's changes to the branch.

When you commit changes to the branch, your changes are copied from your local Workbench to the branch. As a result, these changes are then seen as incoming changes when other developers update from the branch later.

Related concepts

Team programming with CVS
CVS Repositories
Synchronizing with a CVS repository

Related tasks

Checking out a project from a CVS repository
Sharing a new project using CVS
Branching
Synchronizing with the repository
Updating
Committing
Resolving conflicts

Related reference

CVS
CVS Repositories view

Creating a CVS repository location

Prerequisite: A CVS server must already be configured on the host machine to create a valid repository location in the Workbench.

To create a new repository location:

1. Open the CVS Repositories view by selecting **Window > Show View > Other...** on the main menu bar and then selecting **CVS > CVS Repositories** from the Show View dialog. Alternatively, the CVS Repositories view is also shown in the CVS Repository Exploring perspective.
2. On the toolbar, click on ***Add CVS Repository*** (or from the context menu of the CVS Repositories View, select **New > Repository Location**). The Add CVS Repository wizard opens.
3. Enter the information required to identify and connect to the repository location:
    a. In the **Host** field, type the address of the host. (For example: `mymachine.com`).
    b. In the **Repository path** field, type the path to the repository on the host (for example `/home/repo, d:/repo.`)
    c. In the **User** field, type the user name under which you want to connect to the repository.
    d. In the **Password** field, type the password for the above user name.
    e. From the **Connection Type** list, select the authentication protocol of the CVS server. There are three connection methods that come with the Eclipse CVS client:
        ◊ **pserver** − a CVS specific connection method.
        ◊ **extssh** − an SSH 2.0 client included with Eclipse
        ◊ **ext** − the CVS ext connection method which uses an external tool such as SSH to connect to the repository. The tool used by ext is configured in the **Team > CVS > EXT Connection Method** preference page.
    f. If the host uses a custom port, enable **Use Port** and enter the port number.
4. (Optional) Select **Validate Connection on Finish** if you want to authenticate the specified user to the specified host when you close this wizard. (If you do not select this option, the user name will be authenticated later, when you try to access the contents of the repository.)
5. (Optional) Select ***Save Password*** if you want to save the password in the Eclipse keyring file so you do not have to enter the password again the next time you start Eclipse. The keyring file is stored on your local drive and does not use strong encryption so you should not enable this option for sensitive passwords.
6. Click **Finish**. The repository location is created.

Note: Prior to Eclipse 3.0, the ***extssh*** connection method only supported SSH 1.0. In 3.0, this connection method was upgraded to support SSH 2.0 as well.

Tip: If you want to use the ***extssh*** connection method but want to keep your local workspace (sandbox) compatible with the CVS command line client, you can use the ***ext*** connection method and configure it to use ***extssh*** from inside ***Eclipse on the Team>CVS>EXT Connection Method*** preference page.

Related concepts

Team programming with CVS
CVS Repositories

Related tasks

Discarding a CVS repository location
Refreshing the CVS repositories view

Checking out a project from a CVS repository
Discovering branch and version tags
Changing the properties of a CVS Repository Location

Related reference

CVS
CVS Repositories view

Discarding a CVS repository location

Note: You cannot discard a location if you have projects in the Workbench that are shared with that location.

To discard a CVS repository location:

1. Switch to the CVS Repository Exploring perspective or add the CVS Repositories view to the current perspective.
2. In the CVS Repositories view, select the repository location that you want to delete from the Workbench.
3. Select **Discard Location** from the pop−up menu or press the DELETE key.

Related concepts

Team programming with CVS
CVS Repositories

Related tasks

Creating a CVS repository location

Related reference

CVS
CVS Repositories view

CVS

The Workbench is shipped with a built in client for the Concurrent Versions System (CVS). With this client you can access CVS repositories.

You can find more information on CVS at https://www.cvshome.org.

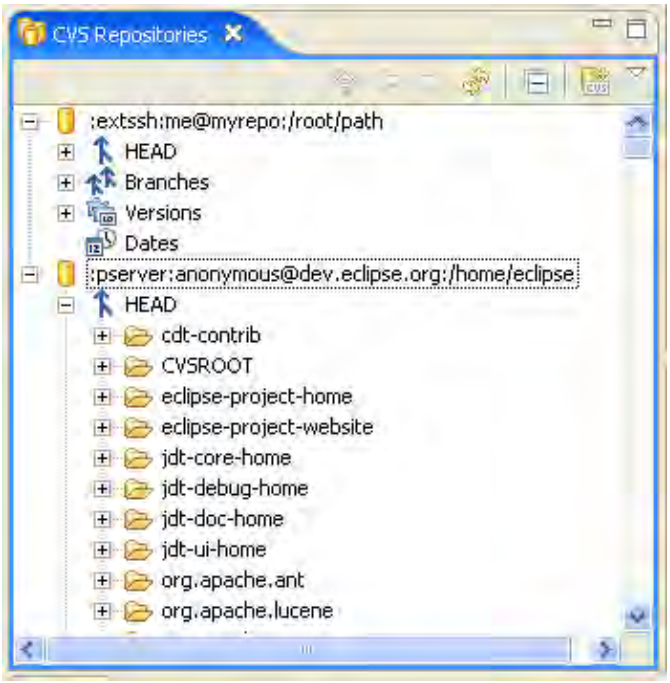
You can also visit the Eclipse CVS FAQ.

CVS Repositories view

The CVS Repositories view, which is part of the CVS Repository Exploring perspective, shows the CVS repository locations that you have added to your Workbench. Expanding a location reveals the main trunk (HEAD), project versions and branches in that repository. You can further expand the project versions and branches to reveal the folders and files contained within them.

The pop−up menu for this view also allows you to specify new repository locations. Use the CVS Repositories view to check out resources from the repository to the Workbench, configure the branches and versions shown by the view, view resource history, and compare resource versions.

Here is what the CVS Repositories view looks like:



There are two types of folders displayed in the repositories view. For the most part, they can be treated in the same fashion. That is, they can be checked out from the Repositories view, etc. However, there are some cases where modules cannot be expanded in the Repositories view even though they can still be checked out.

	A CVS module that maps to a folder on the CVS server.
	A CVS module defined in the CVSROOT/modules file of the repository.

Like other views in the Workbench, the CVS Repositories view has its own toolbar. Toolbar buttons are provided for refreshing the view, navigating and creating new repository locations. The toolbar also contains a drop−down menu that allows the user to filter the view by working set.

Toolbar

Go Home

This command redraws the view, showing the repository locations as the roots displayed by the view.

Go Back

This command redraws the view, showing the roots that where displayed immediately prior to the previous issued Go Into command.

Go Into

This command redraws the view, making the children of the selected entry the roots displayed by the view.

Refresh View

This command refreshes the contents of the Repositories view.

Collapse All

This command collapses all expanded entries in the Repositories view.

Add CVS Repository

This command brings up the wizard to help you specify a new repository location.

Drop−Down Menu

The drop down menu in the title bar allows the repositories view to be filtered by a working set.

Context menu

From the context menu of the Repositories view you can perform a number of interesting operations.

New > Repository Location

This command brings up the wizard to help you specify a new repository location (same as Add CVS Repository toolbar item).

New > Date Tag

This command, available on the repository location and Date category entries, brings up the dialog which allows the specification of a date tag that is to be added to the Date category of the location entry in the Repositories view.

Check Out

This command checks the selected CVS modules out into projects in the Workbench with the same names as the remote modules (for remote modules that are folders on the server) or into projects with the name specified with the module (if the module is defined in the CVSROOT/modules file on the server).

Check Out As...

This command open the Check Out wizard in order to allow the configuration of how the selected remote modules are checked out into the Workbench.

Tag as Version...

This command versions the selected resource based on the current branch contents.

Tag with Existing...

This command versions the selected resource based on the current branch contents, moving the tag from previously tagged resources if required.

Compare

This command will compare two selected resources.

Compare With...

This command will compare the selected folder with a branch or version of the same folder.

Configure Branches and Versions

This command brings up the wizard to help you discover the branch and version tags that exist in the repository for the selected folder so they can be added to the repositories view to allow the resources that have these tags can be browsed.

Refresh Branches and Versions

This command, available on repository location entries, allows you to refresh the list of known branches and versions that are displayed in the repositories view for selected projects. This operation makes use of the defined auto−refresh files for each project. If the operation fails for a particular project, use Configure Branches and Versions on the project to select one or more appropriate refresh files.

Add to Branch List...

This command adds the selected project to the list of projects that are displayed under the specified branch in the repositories view. This command only modifies the repositories view and does not effect the repository in any way. If you want to add the project to a branch, you can perform a ***Tag with Existing*** after performing this operation.

Open

This command opens the selected file in an editor. Since file revisions in the repository are immutable, the editor opens in a read−only state, so it is not modifiable.

Show Annotation

This command shows the contents of file with annotations identifying the author of each line of code in the file.

Show in Resource History

This command shows the revision history of the selected file in the CVS Resource History view.

Properties

Available on repository location entries, this command allows the modification of any of the properties of the location. It also allows the assignment of a display name and a server encoding which is used to translate file paths and commit comments between the client and server (but does not affect file contents).

Related concepts

Team programming with CVS
CVS Repositories
Branches
Versions
Local history

Related tasks

Creating a CVS repository location
Discarding a CVS repository location
Refreshing the CVS repositories view
Changing the properties of a CVS Repository Location
Checking out a project from a CVS repository
Replacing resources in the Workbench
Sharing a new project using CVS
Viewing a file's revision history

Related reference

CVS
CVS Resource History View

Versions

Resources are versioned in order to capture a snapshot of the current state of the resources at one specific point in time. Resources in CVS are versioned by tagging them with a version label. When a resource is versioned, it means that a non−modifiable copy of it can be retrieved from the repository.

Versioning a project saves the line up of all resource versions in the project. Resources other than projects (files and folders) can be versioned. However, it is more common to version entire projects together as the resources contained in a project are often highly interdependent. Projects can be versioned from the workspace or from the branch (including HEAD) in the CVS Repositories view. The difference between these two approaches is in deciding which child resource versions should be part of the project version.

When tagging a project as a version from the *Workbench*, the base revisions of the files in the Workbench are tagged as belonging to that version. This is the preferred method of versioning a project because you know exactly which file revisions will be in the version. This operation is allowed if you have outgoing changes or uncommitted changes. Uncommitted changes are simply ignored and resources with outgoing changes can still have their base revisions be part of the version. Versioning a project with uncommitted or outgoing changes is handy if you have to split the project at the point where you started making changes to the resources and commit the resources to another branch.

When tagging a project as a version from a *branch* in the CVS Repositories view, you are versioning whatever the latest resource versions are in the branch at that moment in time. You should not version your projects from the branch if you do not know what is committed in the branch. For this reason, versioning from the Workbench is often preferable.

Related concepts

CVS Repositories
Branches
Local history
Resources

Related tasks

Creating a version of a project
Versioning projects in the repository
Enabling the CVS resource decorations
Moving version tags

Related reference

CVS

Local history

A local edit history of a file is maintained when you create or modify a file. Each time you edit and save the file, a copy is saved so that you can replace the current file with a previous edit or even restore a deleted file. You can also compare the contents of all the local edits. Each edit in the local history is uniquely represented by the date and time the file was saved.

Only files have local history; projects and folders do not.

Related concepts

Versions

Related tasks

Comparing resources with local history
Replacing a resource with local history
Restoring deleted resources from local history
Setting local history preferences

Creating a version of a project

Comparing resources with the local history

To compare a Workbench resource with a state in the local history:

1. In one of the navigation views, select the resource that you want to compare with a local history state.
2. From the resource's pop−up menu, select **Compare With** > **Local History**. The Compare with Local History page opens.
3. Select a state in the **Local History** list. The Text Compare editor opens.
4. Click the **Select Next Change** and **Select Previous Change** buttons to browse the changes made between the state in the local history and the Workbench resource.
5. Click **OK** when you are finished.

Related concepts

Local history
Versions

Related tasks

Replacing a resource with local history
Restoring deleted resources from local history
Setting local history preferences
Comparing resources

Replacing a resource with local history

To replace a Workbench resource with a state in the local history:

1. In one of the navigation views, select the resource that you want to replace with a local history state.
2. From the resources pop−up menu, select **Replace with** > **Local History**. The Replace from Local History page opens.
3. Select a state from the **Local History** list. The Text Compare editor opens.
4. Click the **Select Next Change** and **Select Previous Change** buttons to browse through the changes made between states in the local history and the Workbench resource.
5. Select the state you want to replace, and click **Replace**.

Tip: You can configure your general preferences to specify how many days to keep files, or how many entries per file you want to keep, or the maximum file size for files to be kept. Select **Window > Preferences > General > Local History**.

Related concepts

Local history
Resources

Related tasks

Comparing resources with the local history
Restoring deleted resources from the local history
Setting local history preferences

Restoring deleted resources from local history

To restore a deleted Workbench resource with a state from the local history:

1. In one of the navigation views, select the folder or project into which you want to restore a local history state.
2. From the resource's pop−up menu, select **Restore from Local History...**. The Restore From Local History dialog opens showing all files that were previously contained in the selected folder or project and all of their sub−folders.
3. Check the files that you want to restore
4. If you don't want to restore just the last state of a file you can select any other state of the file from the **Local History** list on the right hand side of the dialog. The bottom pane of the dialog shows the contents of the state.
5. If you are done with all files click **Restore**.

Tip: You can configure your Workbench preferences to specify how many days to keep files, or how many entries per file you want to keep, or the maximum file size for files to be kept. Select **Window > Preferences > General > Local History**.

Related concepts

Local history
Resources

Related tasks

Comparing resources with the local history
Replacing resources with the local history
Setting local history preferences

Setting local history preferences

To indicate the level of local history that should be kept for each resource in the Workbench:

1. Click **Window > Preferences**.
2. Select the **General > Workspace > Local History** category in the left pane. The Local History Preferences page opens.
3. In the **Days to keep files** field, type the number of days that you want to keep records for any one Workbench resource. For example, if you type 7, then a history of saved states from the last seven days will be kept.
4. In the **Entries per file** field, type the number of states to keep for any one Workbench resource. Note that when you exceed the number of entries per file, the oldest changes are discarded to make room for the newer changes.
5. In the **Maximum file size (MB)** field, type the maximum file size (in MB) of a resource for which a local history should be kept. If the size of the resource exceeds the maximum amount of file size allocated, no local history is kept for that resource.
6. Click **OK** to set your preferences and close the Local History Preferences page.

Related concepts

Local history
Versions
Resources

Related tasks

Comparing resources with the local history
Replacing a resource with local history
Restoring deleted resources from local history

Creating a version of a project

Prerequisite: The project that you want to version must first be in the workspace.

Tip: It is often preferable to synchronize resources with the CVS repository before versioning. This will ensure that there are no outstanding incoming or outgoing changes that might be accidentally excluded from the version.

To make a version of a project:

1. In one of the navigation views, select the project that you want to version.
2. From the project's context menu, select **Team > Tag as Version**.
3. Enter the name of the version you wish to create.
4. Click **OK** to close the Tag Resources dialog and create the version.

Tip: You can browse existing versions of your project by clicking on the **Details** button in the Tag Resources dialog. Clicking **Refresh from Repository** should populate the Existing Versions list. It is sometimes helpful to see existing versions before naming your new version.

Related concepts

Versions
Synchronizing with a CVS repository
Branches
Local history

Related tasks

Sharing a new project using CVS
Versioning projects in the repository
Synchronizing with the repository
Enabling the CVS resource decorations

Related reference

CVS

Sharing a new project using CVS

There are several scenarios that can occur when sharing a project using CVS. The most common scenario is sharing a new project using CVS when the project does not already exist remotely.

To share a new project using CVS:

1. In one of the navigation views, select the project to be shared.
2. Select **Team > Share Project...** from the project's pop−up menu. The Share Project wizard opens.
3. From the list of available repository providers, choose **CVS** and click **Next** to go to the next page. **(Note**: If CVS is not present in the list, it may be disabled. Clicking on the ***Show All Wizards*** button should make it visible.)
4. Select the target repository from the list of known repositories or, if the target repository is not in this list, choose to create a new repository location and click **Next**.
5. If entering a new repository location, enter the repository information and click **Next** when completed. **(Note**: this page is the same format as the Creating a CVS repository location wizard.)
6. Either choose to use the name of the local project as the remote project name or enter another name and click ***Next***.
7. The final page shows you the resources of the new project as outgoing additions. You can choose to commit or ignore resources from this page. **(Note**: If the project already exists remotely, the page will show conflicts on any files that exist both locally and remotely. Only those files whose contents do not match are shown.)
8. Click **Finish** to share the project with the repository. You will be prompted to commit any resources in the new project that have not yet been committed or ignored. Choosing to do so will run the commit operation in the background.

Consequences for Linked Resources

As mentioned in Linked resources, different providers may handle linked resources differently. For Team CVS, linked resources are allowed but ignored. Specifically, projects which contain linked resources can be shared with CVS, but the linked resources themselves cannot be version controlled.

Related concepts

Team programming with CVS
Branches
Synchronizing with a CVS repository

Related tasks

Project checked out with another CVS tool
Checking out a project from a CVS repository
Replacing resources in the Workbench

Related reference

CVS

Project checked out with another CVS tool

It is possible to use a CVS project checked out using another CVS tool into Eclipse. A common concern for those migrating to Eclipse is that they already have their CVS project checked out on their local machine. You *don't* have to check out the contents again to create a project in Eclipse. When such a project is imported into Eclipse, the CVS plug−in may detect that CVS folders exist and auto−share the project. However, there are some instance were this does not occur and the project will need to be shared manually within Eclipse in order to enable the CVS Team menu operations.

To share an CVS project that was already checked out:

1. In one of the navigation views, select the project to be shared.
2. Select **Team > Share Project...** from the project's pop−up menu. The Share Project wizard opens.
3. From the list of available repository providers, choose **CVS** and click **Next** to go to the next page. (**Note**: If CVS is not present in the list, it may be disabled. Clicking on the ***Show All Wizards*** button should make it visible.)
4. The next page displays the sharing information stored in the CVS folder of the project. Click ***Finish*** to share the project and enable the CVS Team menu operations.

Related concepts

Team programming with CVS
Synchronizing with a CVS repository

Related tasks

Sharing a new project using CVS

Related reference

CVS

Checking out a project from a CVS repository

Projects can be checked out from a CVS repository into the Workbench using the Checkout wizard or from the CVS Repositories view.

Checking out using the Checkout Wizard

It is available from the Import menu, the New > Project menu and the toolbar of the CVS Repository Exploring perspective. It is also opened when performing a Checkout As from the CVS Repositories view.

1. Launch the Checkout wizard. It is available from the ***Import*** menu, the ***New*** **>** ***Project*** menu and the toolbar of the CVS Repository Exploring perspective. It is also opened when performing a Checkout As from the CVS Repositories view. (***Note***: If the CVS capability is disabled, the Checkout wizard may only be available from the ***Import*** menu).
2. Select the desired repository from the list of known repositories or, if the desired repository is not in this list, choose to create a new repository location and click **Next**.
3. If entering a new repository location, enter the repository information and click **Next** when completed. **(Note**: this page is the same format as the Creating a CVS repository location wizard.)
4. Either select one or more modules from the list of existing modules or type in the name of the module to be checked out. Entering a name is supported for those situations where obtaining the list of existing modules from the server is impractical or unsupported. You can also enter a dot (.) as the module name if the repository is configured to host a single project at its root. Click ***Next*** when the module or modules to be checked out are specified.
5. (Optional) Choose whether to check out one or more selected modules as a new projects or into an existing project or folder. If one module is selected and it does not contain a .project file (the Workbench project configuration file), you will also have the option to configure the project using the New Project wizard (in order to make the project a Java project, for instance). Click ***Next***.
6. (Optional) For a single project, you can configure the location of the project to be either the default location or a custom location outside the workspace. For multiple projects, you can configure the parent folder where all the projects should be located. Click ***Next***.
7. (Optional) Select the tag to check out from and click ***Finish***.

Checking out from the CVS Repositories view

To check out a project from the CVS repositories view to the Workbench:

1. Switch to the CVS Repository Exploring perspective or add the CVS Repositories view to the current perspective.
2. In the CVS Repositories view, expand the repository location.
3. Expand **HEAD** and select the folders that you want to add as projects to the Workbench. If you are looking for a folder in a particular version:
    a. Expand the **Versions** category and find the folder name that you want to add to the Workbench.
    b. Expand the folder to reveal its versions.

   If you are looking for the latest folder in a branch:

    a. Expand the **Branches** category.
    b. Expand the branch that holds the folder that you want to add to the Workbench.
4. From the pop−up menu for the selected folders, select one of the following:

   a. **Check Out** to check out each of the selected folders as a project in the local workspace with the same name as the folder in the repository.
   b. **Check Out As...** to check out the selected folders into a custom configured project in the local workspace. *Note:* When multiple folders are selected, this operations only allows the specification of a custom parent location for the new projects.
5. If **Check Out As...** was chosen on a single project, one of two possible dialogs is displayed depending on whether the folder in the repository contains a .project file or not.
   a. If there is a .project file, the dialog will accept a custom project name and location.
   b. Otherwise, the dialog will be the New Project wizard which allows full customization of the project (e.g. Java project).

Tip: Any folder, including non−root folders, can be checked out from a CVS repository.

Related concepts

Team programming with CVS
CVS Repositories

Related tasks

Checking out a module from a CVS repository
Checking out a folder into an existing project
Creating a CVS repository location
Replacing resources in the Workbench
Discovering branch and version tags
Sharing a new project using CVS
Synchronizing with the repository

Related reference

CVS
CVS Repositories view
CVS Checkout wizard

Checking out a module from a CVS repository

Modules can be defined in the CVSROOT/modules file of a CVS repository. To check out such a module from a CVS repository to the Workbench:

1. Switch to the CVS Repository Exploring perspective or add the CVS Repositories view to the current perspective.
2. In the CVS Repositories view, expand the repository location.
3. Expand **HEAD** and select the modules that you want to add to the Workbench. *Note*: Modules appear as root folders of the repository but are associated with the  icon.
4. From the pop−up menu select **Check Out**. This will checkout the module into one or more projects in the local workspace, depending on how the module is defined in the repository.

Note: Defined modules can also be checkout out using the Checkout Wizard.

Related concepts

Team programming with CVS
CVS Repositories

Related tasks

Checking out a project from a CVS repository
Checking out a folder into an existing project
Creating a CVS repository location
Replacing resources in the Workbench
Discovering branch and version tags
Sharing a new project using CVS
Synchronizing with the repository

Related reference

CVS
CVS Repositories view
CVS Checkout wizard

Checking out a folder into an existing project

To check out a folder from a CVS repository into an existing project in the Workbench, you can use either the ***Checkout*** wizard or ***Check Out As*** from the CVS Repositories view. Follow these steps to use the Checkout wizard.

1. Launch the CVS Checkout wizard.
2. Select the desired repository from the list of known repositories or, if the desired repository is not in this list, choose to create a new repository location and click **Next**. If entering a new repository location, enter the repository information and click **Next** when completed.
3. Either select one or more modules from the list of existing modules or type in the name of the module to be checked out and click ***Next***.
4. Choose whether to check out one or more selected modules into an existing project or folder. you can provide the following information:
    1. ***Target folder name***: (only when checking out a single module) The name of a local folder in which the contents of the remote folder will be placed. *Note*: If the local folder exists, its contents will be deleted and replaced by the contents of the remote folder.
    2. ***Parent of target folder***: The existing local folder into which the target folders will be created.
    3. ***Checkout sub−folders***: The option to indicate whether the sub−folders of the remote folder should be checked out as children of the target folder.
5. (Optional) Select the tag to check out from and click ***Finish***.

Note: Only folders within non−shared projects or projects shared with the same CVS repository as the selected remote folder are valid targets for the *Checkout Into* operation. Also, if the target project of the operation is an unshared project, the project will be connected to the CVS repository (i.e. the project will become a shared CVS project) but any pre−existing content will be ignored.

Related concepts

Team programming with CVS
CVS Repositories

Related tasks

Checking out a project from a CVS repository
Checking out a module from a CVS repository
Creating a CVS repository location
Replacing resources in the Workbench
Discovering branch and version tags
Sharing a new project using CVS
Synchronizing with the repository

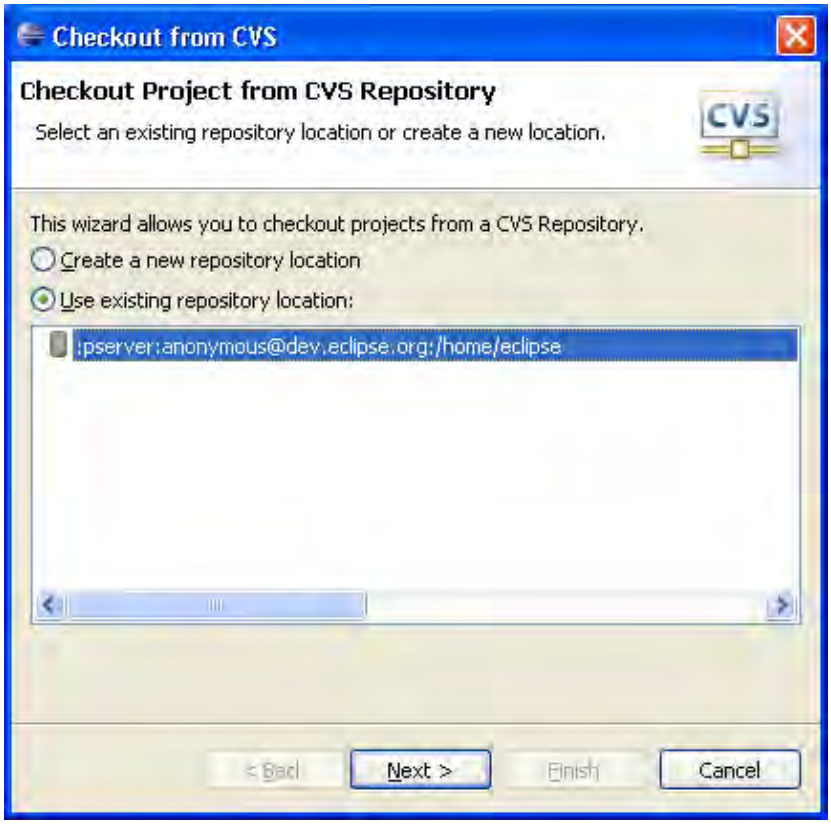
Related reference

CVS
CVS Repositories view
CVS Checkout wizard

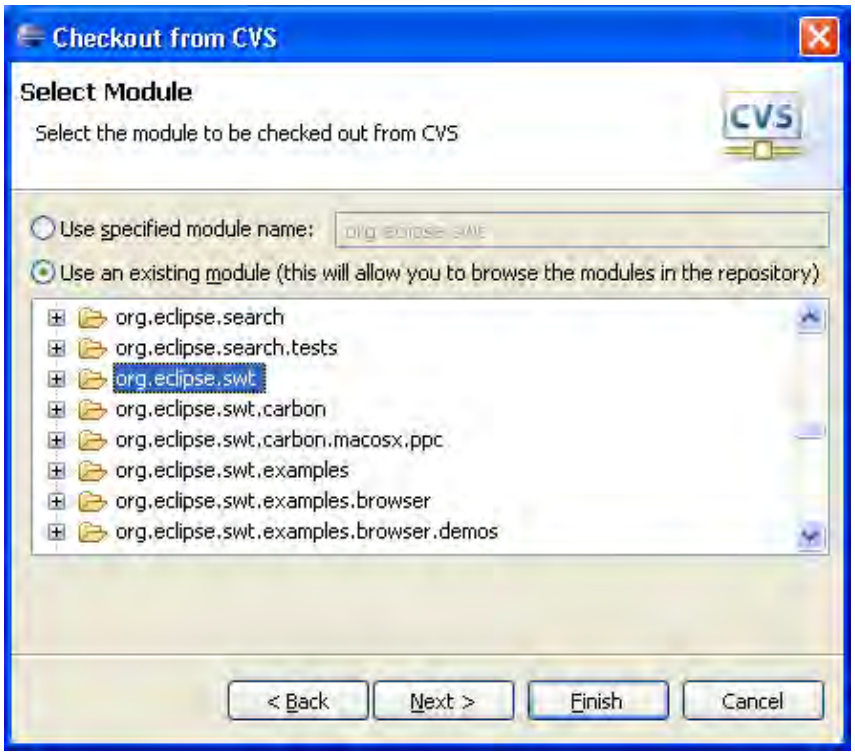
CVS Checkout wizard

This wizard helps you check out one or more projects from a CVS repository. It is available from the Import menu, the New > Project menu and the toolbar of the CVS Repository Exploring perspective. It is also opened when performing a Checkout As from the CVS Repositories view.

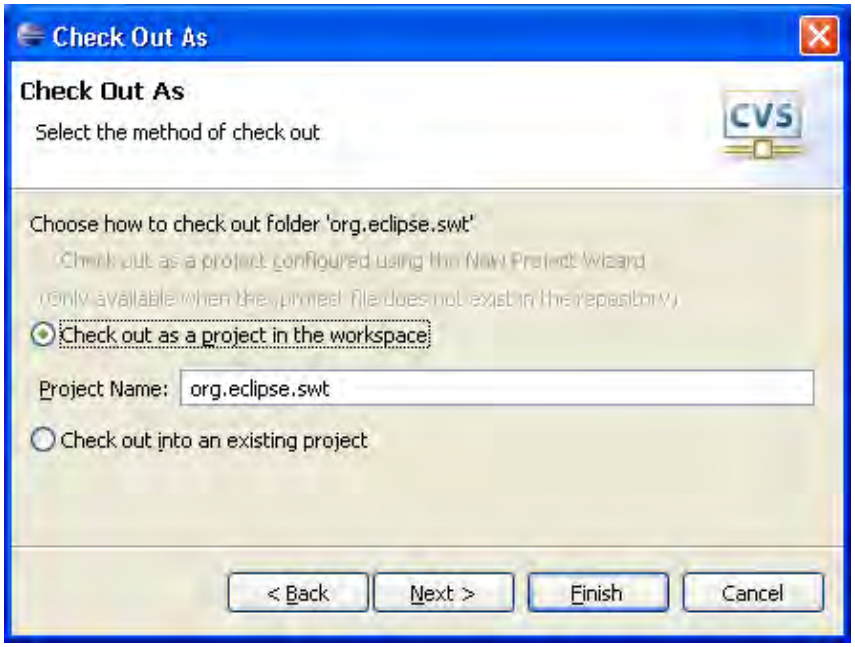
The first page of the checkout wizard allows you to choose an existing repository location or create a new one. If you choose to create a new location, the page from the New Repository Location wizard is shown.



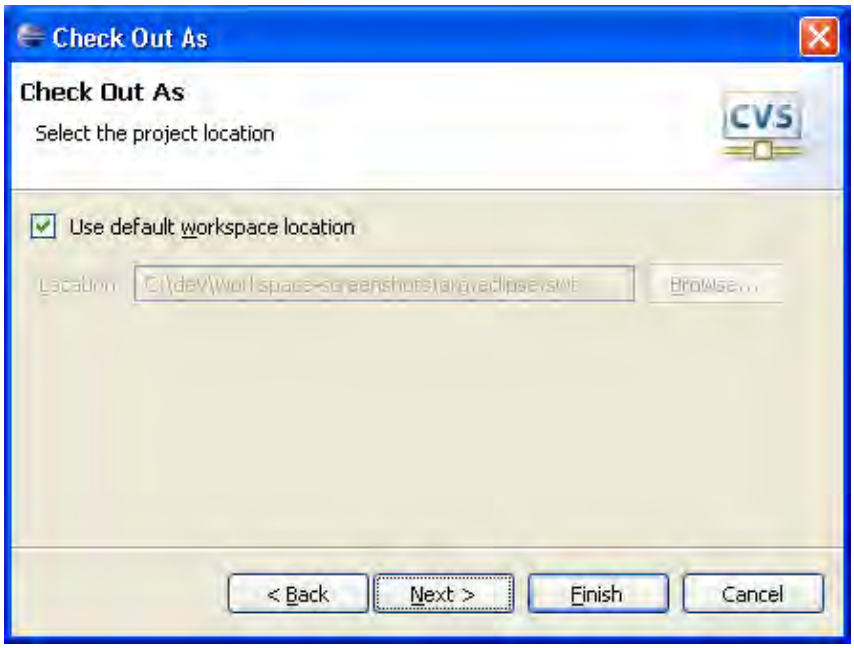
After a repository location is selected, you can now choose one or more modules to check out. You can either enter a module path or select one or more from the list of existing modules. A dot (.) can be entered to checkout the entire contents of the repository for those rare cases where the repository represents a single project.



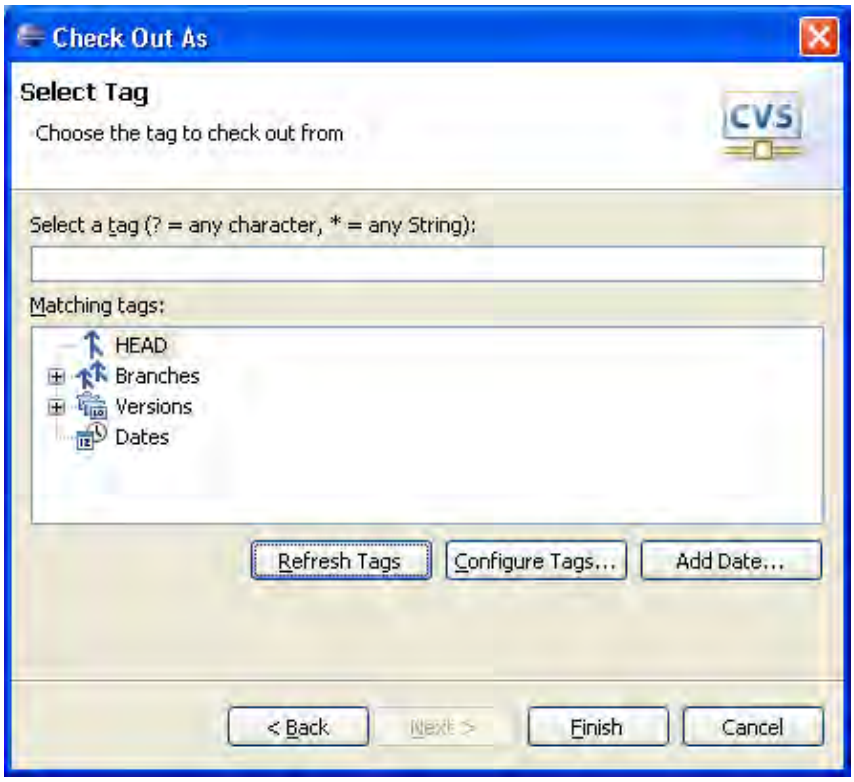
Once one or more modules are selected, you can choose to check out one or more selected modules as a new projects or into an existing project or folder. If one module is selected and it does not contain a .project file (the Workbench project configuration file), you will also have the option to configure the project using the new project wizard (in order to make the project a Java project, for instance).



For a single project, you can configure the location of the project to be either the default location or a custom location outside the workspace. For multiple projects, you can configure the parent folder where all the projects should be located.



Finally, you can choose the tag to be used for the check out.



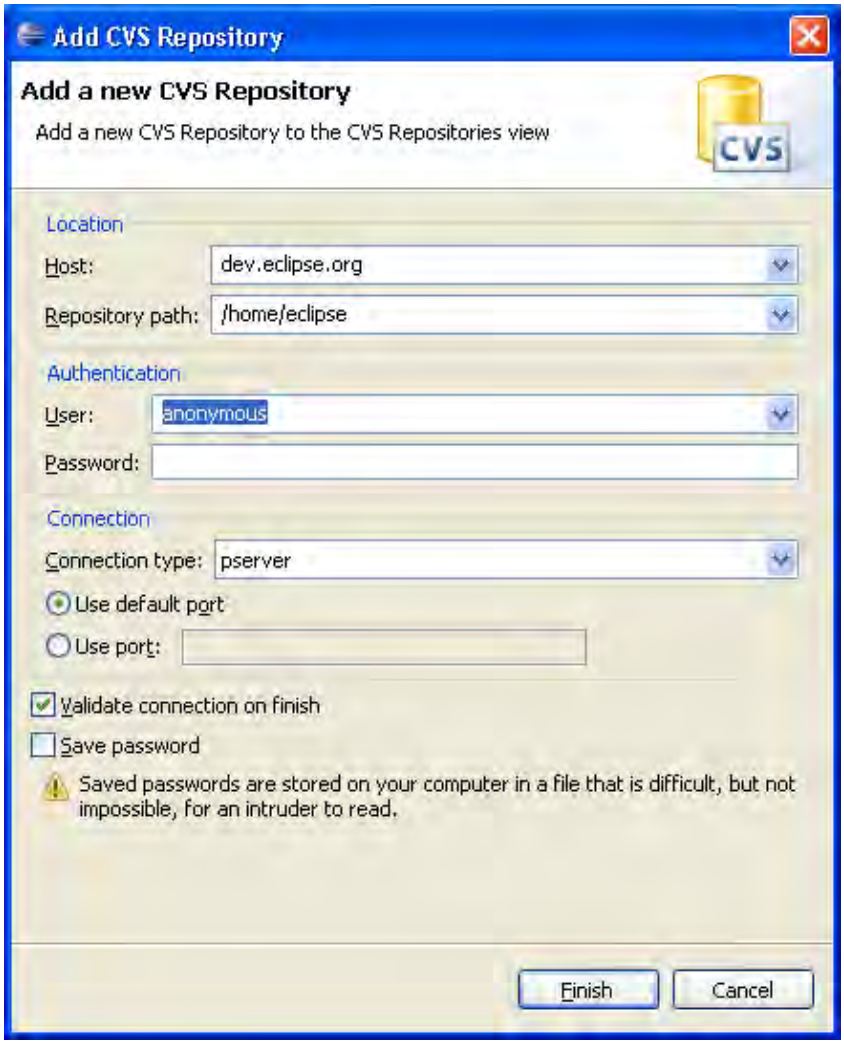
Related reference

CVS Repositories View
Add CVS Repository wizard

Add CVS Repository wizard

This wizard helps you create a repository location.

Here is what the New CVS Repository Location wizard looks like:



Fields

The following are the editable fields for a new CVS repository location:

Option	Description	Default
Host	The address of the host (e.g., "hostmachine.com").	<blank>
Repository Path	The path to the repository on the host (e.g., "/x/y/z", "d:/myrepo").	<blank>
User Name	The user name under which you want to connect to the server.	<blank>
Password	The password required for the above user name to access the above host.	

Connection Type	The type of CVS connection for the repository, either "pserver", "extssh" or "ext".	pserver
Port	Option to use a custom port for the connection.	Use Default Port
Validate connection on finish	Check this option if you want to attempt a connection to the host upon completion of the wizard to validate that the entered information is correct. If this option is not enabled, you will not know whether the information entered is correct until you try to access the contents of a repository for the first time.	On
Save Password	Check this option if you want the password in the Workbench keyring file. The file is stored on your local harddrive and is obfuscated but not strongly encrypted. Only do this if your password is not sensitive or you are sure that no one can access the keyring file on your local disk.	Off

Related reference

CVS Repositories View

Replacing resources in the Workbench

To replace Workbench resources with versions in the repository:

1. Select a resource in one of the navigation views.
2. From the resource's pop−up menu, select one of the following menu items:
    - **Replace With > Latest From <Branch Name>** or **<*Version*>** − replace the Workbench resource with the latest resources currently committed to the branch that the local project is shared with or if the local project is checked out as a version then replace with the same version.
    - **Replace With > Another Branch or Version** − replace the Workbench resource with a specific version or branch that you select in the repository. When you select this option, another window opens, so that you can browse through the branch and version tags in the repository and select the one that you want.
    - **Replace With > Revision** − (available only on a file) replace the Workbench file with another revision of the file. When you select this option, a dialog opens to allow you to select a revision from the list of available ones. A test window displays the comparison between the local revision and the selected remote revision.
    - ***Team > Switch to Another Branch or Version*** − this update will replace Workbench resources with those on the tag specified in the Update dialog. The behavior differs from the above replace operations in that uncommitted modifications in the Workbench are not replaced but instead are "moved" to the resources from the selected tag. For more details on the specifics of this operation, see the CVS documentation at www.cvshome.org relating to the use of the *−r* option with *cvs update*.

Tip: When replacing with a branch or version or when updating, you can specify a particular date instead of a a version or branch tag. This can be done by right clicking on the ***Dates*** category in the tag selection list and choosing ***Add Date***. A Date Tag dialog is displayed that allows you to specify a date and optionally a time. After you click ***Ok***, the date tag will appear in the tag selection list.

Related concepts

Team programming with CVS
Branches
Versions
Synchronizing with a CVS repository

Related tasks

Checking out a project from a CVS repository
Sharing a new project using CVS
Viewing a file's revision history
Branching
Synchronizing with the repository

Viewing a file's revision history

Every time you commit a file to the CVS repository, a new revision is created. Each revision is identified by a number (for example, 1.1 or 1.11.3.5). Along with each revision is stored the author, the date, and a commit comment.

To view the resource history for a file:

1. Select the file in one of the navigation views. From the context menu, choose **Team > Show in Resource History**.
2. The CVS Resource History view will open and show a history for the selected file.

Note: The CVS Resource History view also shows you all version and branch tags that are associated with the file.

Tip: You can drag and drop a file from one of the navigation views into the CVS Resource History view instead of choosing the menu item.

Tip: By clicking on column headings in the CVS Resource History view you can change the sorting order of the items.

Tip: You can have the CVS Resource History view update automatically to the history of the file in the active editor by enabling the ***Link with Editor*** item in the view's toolbar.

Related concepts

Team programming with CVS
Branches
Synchronizing with a CVS repository

Related tasks

Synchronizing with the repository

Related reference

CVS
CVS Resource History view

Synchronizing with the repository

Method 1: Using the context menu

To synchronize resources in the Workbench with those in the repository:

1. In one of the navigation views, select the resources that you want to synchronize.
2. Right click and select **Team > Synchronize with Repository**. The Synchronize view opens.

Method 2: Using the synchronize action

1. From the Team Synchronizing perspective select the ***Synchronize...*** action from the Synchronize  button drop down.
2. Select ***CVS*** from the list of possible synchronization types and click ***Next***.
3. Select the resource scope for the synchronize by either selecting Workspace, Selected Resources or Working Set. Then select ***Finish.***
4. The Synchronize view will open.

Note: The synchronize action is not enabled by default in other perspectives. You can enable the action to appear in your current perspective by selecting ***Window > Customize Perspective***. Then click on the ***Commands*** tab and check off ***Team***.

Tip: If you choose the Workspace scope for your synchronization, you can then use method 3 below to always have an up−to−date view of your local modifications available in the Synchronize view without re−running the synchronize operation.

Method 3: Using a pinned CVS Workspace Synchronization in the Synchronize view

Once you have a CVS workspace synchronization in the Synchronize view, you can pin it. This will prevent it from being replaced by the next CVS workspace synchronization that is launched using one of the previous 2 methods. Here are some of the advantages of using a pinned synchronization.

- The local modification state of the resources being synchronized is kept up−to−date. This means that, if you modify a file locally that is within the scope of a synchronization that appears in the Synchronize view, the resource will appear automatically in the view, if it is not already there.
- The remote state of all of the resources being synchronized can be refreshed using the Synchronize  button in the toolbar of the Synchronize view.
- The remote state of a selection of resources can be refreshed using the Synchronize command from the context menu in the Synchronize view.
- You can schedule the refreshing of the remote state to happen at particular intervals (each hour, for example)

The implications of this is that you can see you outgoing resources without re−fetching the remote state from the server (a potentially long running operation). Also, the fetching of the remote state is run in the background so you can do other things (e.g. inspect changes) while the remote state is fetched.

From within the synchronize view

1. Use the toolbar buttons to switch modes for this view. There are four modes:
   - Incoming mode − shows incoming changes only (resources in the repository that differ from what is in the Workbench).
   - Outgoing mode − shows outgoing changes only (resources modified in the Workbench).
   - Incoming/Outgoing mode − shows both incoming and outgoing changes.
   - Conflicts mode = shows only conflicting resources.

   Important: It is preferable to update resources in the Workbench first, resolve any conflicts that exist by merging, then commit Workbench resources to the repository.
2. It is possible that someone has committed a new revision of your file since you started working on it. This will result in a *conflict*, and care must be taken to resolve this. For this reason, conflicts are shown in all modes of the Synchronize view.

Related concepts

Team programming with CVS
Branches
Synchronizing with a CVS repository
Three way comparisons

Related tasks

Committing
Updating
Resolving conflicts
Comparing resources
Merging changes in the Compare editor
Merging from a branch
Version control life cycle: adding and ignoring resources
Replacing resources in the Workbench

Related reference

CVS
Synchronize view

Three way comparisons

Three way comparisons show the differences between three different versions of a resource. This feature is most useful when merging resources or when there is a conflict during synchronization. Conflicts occur when two developers add a version from the same branch to their Workbench, then each developer modifies it, then one developer attempts to commit the resource after the other developer has already committed it.

When this situation arises, you can view the differences between three resource versions: the resource in the Workbench, the version of the resource that is committed in the branch, and the *common ancestor* from which the two conflicting versions are based. If a common ancestor cannot be determined, for example because a resource with the same name and path was created and committed by two different developers, the comparison becomes a two−way comparison.

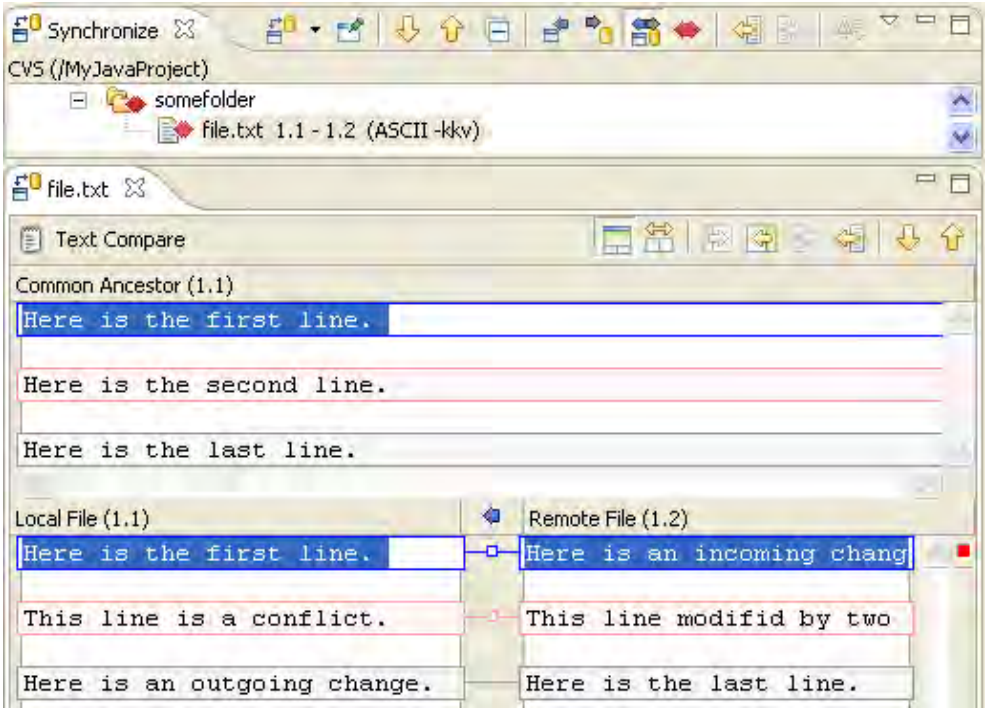
Interpreting compare results

The Synchronize view allows you to view the differences between two or three files. If a common ancestor is available, the sync view performs a three way comparison. It is possible that a common ancestor for two conflicting resource versions cannot be determined, (e.g. a resource with the same name and path is created and committed by two different developers). In this case the compare becomes a regular two way compare.

In a three way compare the Workbench shows you:

- what has been changed in the first child in comparison to the common ancestor.
- what has been changed in the second child in comparison to the common ancestor.

In the picture below, the common ancestor is displayed in the top pane of the text compare pane. The differences that you see highlighted are what has changed in the Workbench resource as compared to the common ancestor, and what has been changed in the branch resource as compared to the common ancestor. The sections that differ in all three files are highlighted as differences. Conflicts are shown in red, incoming changes in blue, and outgoing changes in gray.



Related concepts

Synchronizing with a CVS repository

Related tasks

Comparing resources
Synchronizing with a repository
Merging changes in the compare editor
Resolving conflicts
Setting preferences for comparing files
Comparing resources with repository versions

Related reference

Compare editor
CVS Synchronize view

Merging changes in the compare editor

The toolbar buttons in the compare editor allows you to merge changes from the left file to the right file and vice versa. There are four types of merges you can perform:

- Copy All from Left to Right
- Copy All from Right to Left
- Copy Current Change from Left to Right
- Copy Current Change from Right to Left

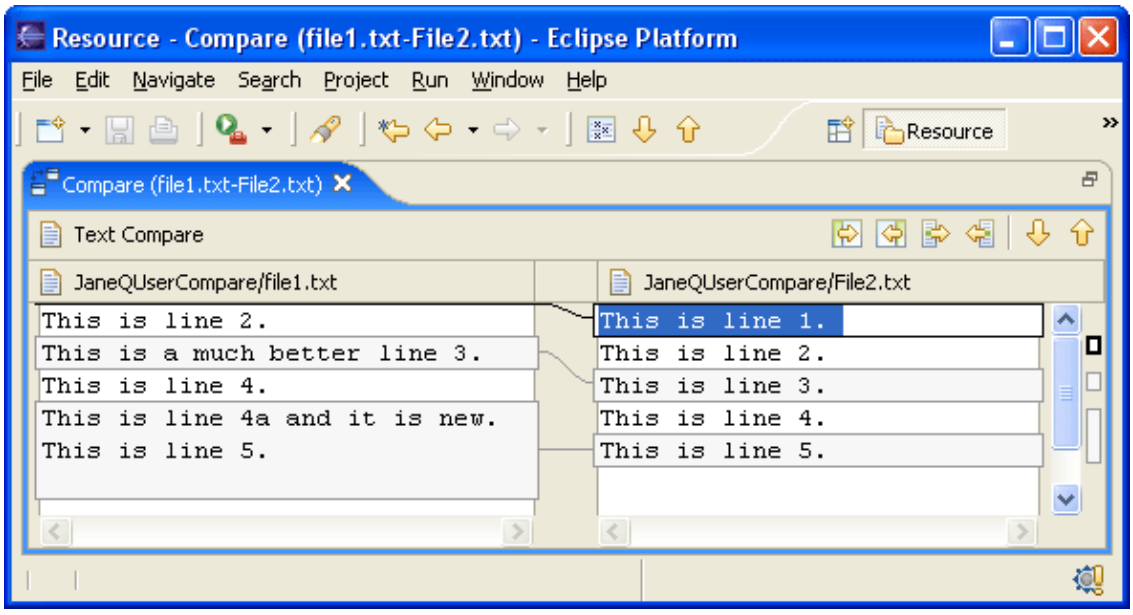
.

The **Copy All from Left to Right** and **Copy All from Right to Left** actions completely replace the contents of one resource with the other resource.

To merge a single change:

1. Select the highlighted difference that you want to merge.
2. Depending on what you want to do, click either the **Copy Current Change from Right to Left** or the **Copy Current Change from Left to Right** toolbar button. The selected text is copied from one file to the other.
3. Right click to get the resource's pop−up menu, and select **Save**.

The image below gives an example of two files (file1.txt and file2.txt) compared.



Related concepts

Three way comparisons
Synchronizing with a CVS repository

Related tasks

Comparing resources

Understanding the comparison
Synchronizing with the repository
Resolving conflicts
Setting preferences for comparing files

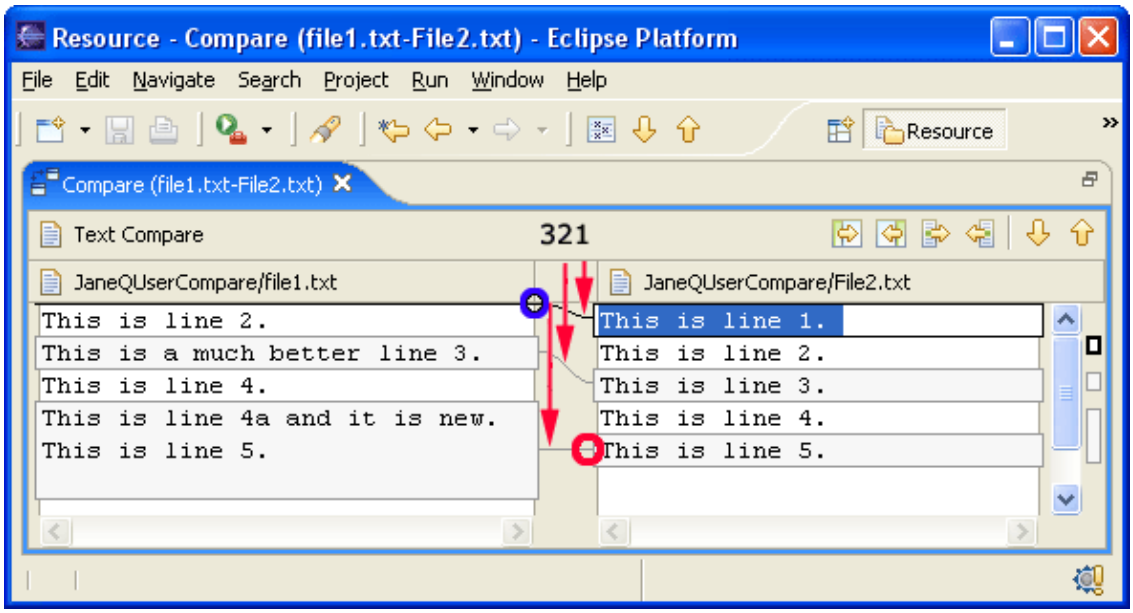
Related reference

Compare editor
CVS Workspace Synchronization

Understanding the comparison

If you compare two files (file1.txt and file2.txt), the following results are shown in the compare editor. The left side shows the contents of file1.txt and the right side shows the contents of file2.txt. The lines connecting the left and right panes indicate the differences between the files.

You can double click on the editor tab to maximize the editor.



Looking at the numbered changes in the above image:

- Starting with the top line (in the left pane) we can see that the difference bar (in the area of the blue circle) indicates something is missing from the very top of the left file. If we follow the difference band (see #1) to the right file we can see that it contains "This is line 1" .
- The next line "This is line 2." is white indicating it matches the right file.
- Moving onto the next line (colored in gray) we can see that the left file and right file have different contents for this line (see #2).
- The next line ("This is line 4") is once again in white, so we can skip it, since the contents are the same in both.
- The next line exists in the left file but since it is gray we follow its difference bar to the right (see #3) and notice that the right file does not contain the line (see red circle).

Hint: On the right−hand side of the comparison, to the right of the scrollbar, there is a column which shows a graphical representation of all differences between the resources. You can click on any of the segments displayed there to quickly scroll to that difference.

Related concepts

Synchronizing with a CVS repository
Three way comparisons
Local history

Related tasks

Comparing resources
Synchronizing with the repository
Merging changes in the Compare editor
Resolving conflicts
Setting preferences for comparing files
Comparing resources with repository versions
Tiling editors

Related reference

Compare editor
CVS Workspace Synchronization

Resolving conflicts

When updating or committing you may encounter conflicts. A conflict occurs when you have locally modified a resource for which a more recent revision is available in the branch in the repository. Specifically, the branch will contain a revision newer than the base revision of your resource. In this situation you can choose to do one of the following:

- You can take the change from the branch, discarding your local work. This could occur if you have made unintended changes locally, or if you realize that the revision in the repository is better than yours. Overwriting your local changes should be done with caution since you are in essence throwing work out.
- You can commit your change, subsuming the revision in the repository. This should be done with great caution since you are in essence throwing away someone else's work. In particular, the change you are overwriting may have other dependencies in the branch (for example there may be other incoming changes which depend on the conflict).
- You can merge your work and the repository resource, saving locally the merged resource. You may then later choose to commit this merged result.

Typically, you will want to take the third option, that is to merge, because of the loss of work issues with the other two choices.

Manually merging changes

The Synchronize View indicates those resources which are in conflict with the branch. For a given resource in conflict, typically you will want to merge your changes with changes in the branch's resource.

For example, let us assume that both you and another team member have modified the same html page. Opening that resource from the Synchronize view will display a comparison of the local resource and the branch revision. By cycling through and merging the individual changes, you can decide for each change whether to accept the incoming change, reject it, or merge it with your local changes. When you are finished merging, you save your changes. This overwrites your local resource with the result of the merge. You can subsequently commit this merged resource.

Tip: When merging changes, it is often convenient to be able to distinguish which files you have completed merging. When you're done merging a file, you can pick **Mark as Merged** from the context menu. This will change the status of the file from being a conflict to being an outgoing change.

You can merge differences in the Synchronization view on two levels:

- **Whole Document** − In the Synchronize view, open the resource that you want to merge in a Compare editor. In the Compare editor, click the Copy all non−conflicting Changes from Right to Left button to copy all non−conflicting changes. Conflicting changes will have to copied individually.
- **Current Change** − In the Text Compare editor, either use the Go to Next Difference and Go to Previous Difference buttons to navigate to the change that you want to merge, or simply click in either source pane somewhere in the desired change difference fragment. Click the Copy current change from right to left button as needed to overwrite the highlighted fragment either with the corresponding modification in the branch.

Auto merging changes

It is also possible to have your changes automatically merged for you. For any resource marked as ASCII, performing a **Team > Update** will automatically merge into your local resource differences with the branch resource. This works fine provided there are no lines with conflicting changes. If there are, CVS inserts special markup in the file to indicate those lines which could not be merged.

Updating from within the Synchronization view works a bit differently. In the case of a conflict, ***Update*** will only process files whose contents contain no conflicts. Files that have content conflicts will be skipped and left in the Synchronize view as conflicts.

Related concepts

Team programming with CVS
Synchronizing with a CVS repository
Three way comparisons

Related tasks

Synchronizing with the repository
Updating
Committing
Merging from a branch
Comparing resources
Merging changes in the Compare editor

Related reference

www.cvshome.org: Bringing a file up to date
www.cvshome.org: Conflicts example
Compare editor

Updating

While you are working on a project in the Workbench, other members of your team may be committing changes to the copy of the project in the repository. To get these changes, you may "update" your Workbench to match the state of the branch. The changes you will see will be specific to the branch that your Workbench project is configured to share. You control when you choose to update.

The update command can be issued from two places: the **Team > Update** menu, or the **Synchronize** view. In order to understand the difference between these two commands, it is important to know about the three different kinds of incoming changes.

- A *non−conflicting* change occurs when a file has been changed remotely but has not been modified locally.
- An *auto−mergeable conflicting* change occurs when an ASCII file has been changed both remotely and locally (i.e. has non−committed local changes) but the changes are on different lines.
- A *non−auto−mergeable conflicting* change occurs when one or more of the same lines of an ASCII file or when a binary file has been changed both remotely and locally (binary files are never auto−mergeable).

When you select **Team > Update**, the contents of the local resources will be updated with incoming changes of all of the above three types. For non−conflicting and auto−mergeable conflicts, there is no additional action required (for auto−mergeable conflicts, the changed local resource is moved to a file prefixed with ".#" just in case the auto−merge wasn't what the user wanted). However, for non−auto−mergeable conflicts, the conflicts are either merged into the local resource using special CVS specific markup text (for ASCII files) or the changed local resource is moved to a file prefixed with ".#" (for binary files). This matches the CVS command line behavior but can be problematic when combined with the Eclipse auto−build mechanism. Also, it is often desirable to know what incoming changes there are before updating any local resources. These issues are addressed by the Synchronize view.

To open the Synchronize view in incoming mode:

1. In one of the navigation views, select the resources which you want to update.
2. From the pop−up menu for the selected resources, select **Team > Synchronize with Repository**. The Synchronize view will open.
3. On the toolbar of the Synchronize View, click the **Incoming mode** button to filter out any modified Workbench resources (outgoing changes) that you may have.

In incoming mode, you will see changes that have been committed to the branch since you last updated. The view will indicate the type of each incoming change. There are two update commands (available from the context menu of any resource in the view) to deal with the different types of conflicts: **Update** and **Override and Update**. When you select the **Update** command in the Synchronize view, all selected incoming and auto−mergeable conflicting changes are processed while conflicts that are not auto−mergeable will not be updated (any files that have been successfully processed are removed from the view). The **Override and Update** command operates on conflicts and will replace the local resources with the remote contents. This "replace" behavior is rarely what is desired. An alternative is described later.

To update non−conflicting and auto−mergeable files:

1. The Structure Compare pane at the top of the Synchronize view contains the hierarchy of resources with incoming changes.

2. Select all conflicting files and choose **Update** from the pop−up menu. This will update the selected resources that are either incoming changes or auto−mergeable conflicts and remove them from the view. Conflicts whose contents are not auto−mergeable will be left in the view.

If your local Workbench contains any outgoing changes that are not auto−mergeable with incoming changes from the branch, then, instead of performing an **Override and Update**, you can merge the differences into your Workbench manually, as follows:

1. In the Structure Compare pane, if there is a conflict in the resource list (represented by red arrows), open it (either by double−clicking or selecting ***Open in Compare Editor*** from the context menu).
2. In the Text Compare area of the compare editor, local Workbench data is represented on the left, and repository branch data is represented on the right. Examine the differences between the two.
3. Use the text compare area to merge any changes. You can copy changes from the repository revision of the file to the Workbench copy of the file and save the merged Workbench file (using the pop−up menu in the left pane).
4. Once you are completed merging the remote changes into a local file, choose **Mark as Merged** from the pop−up menu in the Synchronize view. This will mark the local file as having been updated and allow your changes to be committed.

Note: The repository contents are not changed when you update. When you accept incoming changes, these changes are applied to your Workbench. The repository is only changed when you commit your outgoing changes.

Tip: In the Synchronize view, selecting an ancestor of a set of incoming changes will perform the operation on all the appropriate children. For instance, selecting the top−most folder and choosing **Update** will process all incoming and auto−mergeable conflicting changes and leave all other incoming changes unprocessed.

Warning: The behavior of the **Override and Update** command described above only applies to the incoming mode of the Synchronize view. In the **Incoming/Outgoing mode** of the view, the behavior for incoming changes and conflicts is the same but the command will revert outgoing changes to whatever the repository contents are. Exercise great caution if using this command in Incoming/Outgoing mode.

Related concepts

Team programming with CVS
Synchronizing with a CVS repository

Related tasks

Committing
Resolving conflicts
Comparing resources
Version control life cycle: adding and ignoring resources

Related reference

CVS
Synchronize view

Committing

You can commit Workbench resources that you have modified to the repository so that other team members can see your work. Only those changes committed on that branch will be visible to others working on that branch. The commit command can be issued from two places: the **Team > Commit** menu, or the **Synchronize** view.

To commit changes using **Team > Commit**:

1. In one of the navigation views, select the resources that you want to commit.
2. Right−click on the resources and select **Team > Commit** from the pop−up menu.
3. If there are new files whose file types cannot be determined automatically, the first page of the Commit dialog will present the unknown types and allow you to set them appropriately to either ASCII or binary. Click ***Next*** to continue.
4. On the Comment page, provide a comment for your changes (for example, `Fixed the spelling mistakes`).

The Comment page also allows the user to preview the files that are about to be committed. If any of the files are known to be conflicting changes, the commit will not be allowed. If there are no known conflicting changes the commit will be allowed but there could still be conflicting changes on the server (i.e. conflicting changes on the server become known to the client during a synchronize operation). If there are conflicting changes on any files that are committed, the operation will fail. If this occurs, you must either perform an update or use the Synchronize view to resolve the conflicts. It is considered a more ideal workflow to always update before committing in order to ensure that you have the latest state of the repository before committing more changes.

If one or more of the resources being committed are new and not yet added to CVS control, they will be added automatically unless they are explicitly removed by choosing ***Remove from View*** from the context menu.

To commit changes in the Synchronize view:

1. In one of the navigation views, select the resources that you want to commit.
2. Right−click to open the pop−up menu and select **Team > Synchronize with Repository**. The Synchronize view will open.
3. On the toolbar of the Synchronize view, select the **Outgoing mode** button to show any modified Workbench resources (outgoing changes) that you may have.
4. If there are conflicts (red arrows), resolve them. To do so. open them in a Compare editor and use the text compare area to merge resources with conflicts. You can copy changes from the repository revision of the file to the Workbench revision of the file and save the merged Workbench resource. Once all the conflicts in the Structure Compare area have been resolved, perform a ***Mark as Merged*** on the resource in the Synchronize view to make the change an outgoing change and you are ready to commit.
5. In the Structure Compare pane, right−click the top of the hierarchy that you want to commit, and select **Commit** from the pop−up menu.
6. In the Commit Comment dialog box, provide a comment for your changes (for example, `Fixed the spelling mistakes`). Again, if there are new files of an unknown type, you will be asked to specify what type they should be.

Tip: You can commit files that are in conflict by performing an **Override and Commit**. This will commit the Workbench copy of the resource into the repository and thus remove any of the incoming changes.

Warning: The behavior of the **Override and Commit** command described above only applies to the outgoing mode of the Synchronize view. In the **Incoming/Outgoing mode** of the view, the behavior for outgoing changes and conflicts is the same but the command will revert incoming changes to whatever the local Workbench contents are. Exercise great caution if using this command in incoming/outgoing mode.

Related concepts

Team programming with CVS
CVS Repositories
Branches
Synchronizing with a CVS repository

Related tasks

Updating
Resolving conflicts
Comparing resources
Version control life cycle: adding and ignoring resources

Related reference

CVS
Synchronize view

Version control life cycle: adding and ignoring resources

When committing resources, often there are resources that you do not want to store in the repository. For example, external editors might create temporary files in your project, compilation of .java files might create .class files, or some build operations might result in binary files. When put together, these generated files may be quite large. They may also be regenerated whenever a build is performed, resulting in many outgoing changes. Typically these are not files that one wants to persist in the repository or to share with other members of a team.

Team CVS has two related tasks that allow you to control which files are stored in the repository: adding a resource to version control, and ignoring a resource.

Adding a file to version control

Team CVS does not automatically add files to version control. Rather, it's your choice to explicitly add files to version control. This is accomplished by selecting the **Team > Add to Version Control** menu.

When performed on a file, it will add that file to version control. The result is that the CVS repository immediately creates an entry so that it can start maintaining history state for that file. This occurs even before you commit the file to the repository.

When adding a folder or project, the action will recursively descend into sub−folders, adding those files it finds to version control, provided the files have not been explicitly ignored.

Tip: When committing files, if the selection either directly or recursively contains files which have not been added to version control, you will be prompted whether or not you want them added. This is a convenience function to help ensure that you do not miss committing new resources which were unintentionally never added.

How may I ignore thee, let me count the ways

There are several facilities that allow you to specify which resources should be excluded from version control:

1. There is a global preference which you can use for ignoring files and directories that match a certain filename pattern. For example, if you create a global ignore for `/bin`, and any resource that matches "bin" in any directory in the workspace will be ignored for version control. This preference can be found in **Window > Preferences > Team > Ignored Resources**.
2. Any resource marked as *derived* will be automatically ignored for version management by Team CVS. Some builders, such as the Java builder, mark all of its build output (e.g. .class files) as derived.
3. CVS supports the creation of a special *.cvsignore* file whose contents describe which files or folders to ignore for version management. The .cvsignore file only applies the pattern to resources in the *same* directory as the .cvsignore file itself.

Tip: Once a resource is under version control, it cannot easily be subsequently ignored. This is why adding to version control is an explicit operation performed by you the user.

Related concepts

Team programming with CVS

Related tasks

Creating a global ignore pattern
Authoring the CVS .cvsignore file
Committing
Updating

Creating a global ignore pattern

When synchronizing resources, often there are resources that you do not want to commit to the repository. The Workbench provides a global pattern matching facility for ignoring resources that can appear anywhere in your project hierarchy.

1. From the main menu bar, select **Window > Preferences**.
2. In the left pane of the Preferences window, expand the **Team** category and select **Ignored Resources**. This displays a list of resource name patterns against which resources will be screened, before they are considered as candidates for version control.
3. Click the **Add** push button and type the pattern that you want to match, for example `*.obj`.
4. Click **OK**.

All files with an extension that matches the pattern that you added will be excluded from version control.

The patterns that you define may contain the wildcard characters * and ?. The asterisk represents any sequence of zero or more characters, the question mark represents a single character. For example, you can specify a pattern of `*~`, which would match any temporary files that end with a tilde (~). Any file or directory that matches any one of the patterns will be ignored when files are being considered for version control. You can temporarily disable ignoring the resource name pattern by deselecting it from the list. You do not have to remove the specified pattern from the list.

It is important to note that the path leading up to the resource name is not included in the matching. For example, for the file "/path/to/file.txt", only the string "file.txt" is matched against the patterns. This facility is not intended for specifying fully−qualified path names but for specifying globally applicable patterns.

Related concepts

Team programming with CVS

Related tasks

Version control life cycle: adding and ignoring resources
Overriding or removing resource ignore patterns
Synchronizing with the repository

Overriding or removing resource ignore patterns

After you have added global file−matching patterns to the list of files to be ignored for version management, you may wish to occasionally override the "ignore".

1. From the main menu bar, select **Window > Preferences > Team > Ignored Resources**.
2. To temporarily override one or more of the file−matching patterns, clear their checkboxes.
3. Click **OK**.

Files that match the patterns you have deselected will be considered for version control.

To permanently remove a pattern, select it and click **Remove**.

Related concepts

Team programming with CVS

Related tasks

Version control life cycle: adding and ignoring resources
Creating a global ignore pattern
Synchronizing with the repository

Authoring the CVS .cvsignore file

When committing resources, often there are resources that you do not want to add to version control. One way you can do this is by using the CVS "ignore" facility, which reads the contents of the file *.cvsignore* to determine what to ignore.

The *.cvsignore* file can be added to any directory of a project. Many existing CVS projects already contain several of these files. This text file consists of a list of files, directories, or patterns.

For example, to add a .cvsignore file to ignore the entire bin directory of an existing project:

1. In one of the navigation views, select a project that contains a /bin directory. The bin directory will commonly contain the projects build output. These are files that are generated from the project's source files and are usually not version−controlled.
2. From the pop−up menu for the project, select **New > File**.
3. Type `.cvsignore` as the file name, then press **Finish**. The file will be created in your project's root directory. You should see it in one of the navigation views.
4. Enter `bin` in the .cvsignore file and save it.
5. Select the project. From the context menu, select the **Team > Synchronize with Repository** menu item. Notice that the bin directory does not show as an outgoing change. It is ignored.

The cvsignore file consists of a list of files, directories, or patterns. In a similar way to the global ignore facility, the wildcards * and ? may be present in any entry in the .cvsignore file. Any file or sub−directory *in the current directory* that matches any one of the patterns will be ignored.

Tip: In the Team menu and in the Synchronize view resource context menu there is a menu item (**Add to .cvsignore**) for adding a file pattern to the appropriate .cvsignore file. This menu item is enabled for resources that are not yet under CVS version control.

Related concepts

Team programming with CVS

Related tasks

Synchronizing with the repository
Version control life cycle: adding and ignoring resources
Creating a global ignore pattern

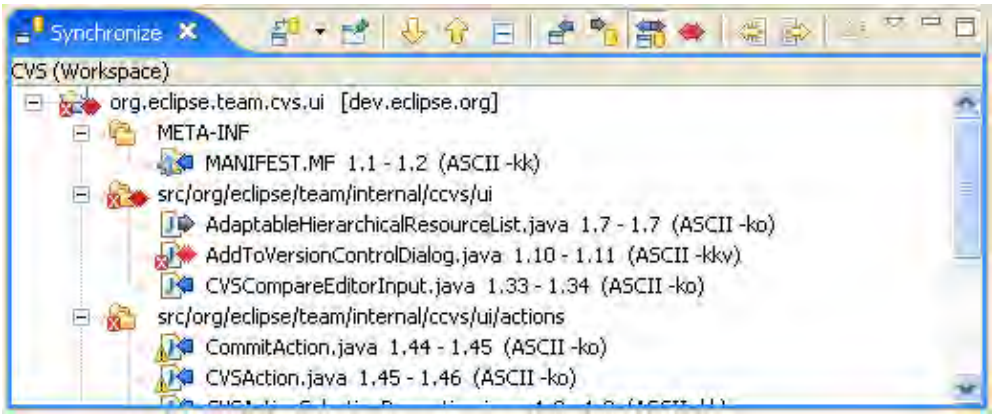
Related reference

www.cvshome.org: Ignoring files via cvsignore

CVS Workspace Synchronization

CVS workspace synchronizations launched using the **Team > Synchronize** menu command or the **Synchronize** toolbar command in the Team Perspective will appear in the Synchronize view. This view allows you to inspect the differences between the local Workbench resources and their remote counterparts as well as update resources in the Workbench and commit resources from the Workbench to a repository.

Here is what the CVS Workspace Synchronization in the Synchronize view looks like:



Features

The following is a brief summary of some of the features of the synchronize view.

Synchronization state

The synchronize view shows the synchronization state of resources in your workspace compared to those in the repository. This state is shown by using icons and can also be configured to show the state as text appended to the resource name. A description of the icons is shown in the table below:

	An incoming addition means that a resource has been added to the repository. ***Updating*** will transfer the resource to your workspace.
	An incoming change means that the file has changed in the repository. ***Updating*** will transfer the new file revision to your workspace.
	An incoming deletion means that a resource was deleted from the server. ***Updating*** will delete your local resource.
	An outgoing addition means that the file was added to your workspace and is not yet in the repository. ***Adding*** then ***Committing*** will transfer the new file to the repository.
	An outgoing change means that the file was change locally. ***Committing*** the file will transfer the changes to the repository and create a new revision of the file.
	An outgoing deletion is a resource that has been deleted locally. ***Committing*** these resources will cause the remote resource to be deleted. ***Note***: in CVS directories are never really deleted from the repository. Instead, files are deleted and empty directories are pruned from your workspace.
	A conflicting additions means that the resource has been added locally and remotely.
	A conflicting change means that the file has been changed locally and remotely. A manual or automatic merge will be required by the user. Also, any entries in the view that contain children

	that are conflicts will also be decorated with the conflict icon. This is done to make conflicts easy to find.
	A conflicting deletion means that the resource was deleted locally and remotely.

Mode

The Synchronize view can be filtered using modes using either the toolbar actions or the menu items in the view's drop down menu. Modes can be used to show only incoming, outgoing or conflicting changes. The advantages to modes are:

- They support and encourage the ideal work flow.
- They reduce the amount of information you are faced with at any one time.
- They provide a degree of safety restricting you to those operations appropriate to that mode. For example, you can't accidentally commit a conflict when in incoming mode.

Important: It is preferable to update resources in the Workbench first, resolve any conflicts that exist by merging, then commit Workbench resources to the repository.

Layout

There are three options for Synchronize view layout for CVS: Flat, Tree and Compressed Folders.

- Flat layout shows all the out−of−sync resources as top level items.
- Tree layout shows the resource hierarchy as it is shown in the Resource Navigator.
- Compress Folders shows changes grouped by project and then by folder. This results in a hierarchy that is at most three levels deep with folder paths being compressed into a single level (similar to a Java package).

Navigation

The Synchronize view provides toolbar actions for navigating through the changes in the view. This actions not only navigate between files but also go from change to change within a file.

Update and Commit Operations

There are several flavors of update and commit operations available in the Synchronize view. You can perform the standard update and commit operation on all visible applicable changes or a selected subset. You can also choose to override and update, thus ignoring any local changes, or override and commit, thus making the remote resource match the contents of the local resource. You can also choose to clean the timestamps for files that have been modified locally (perhaps by an external build tool) but whose contents match that of the server.

Conflict Handling

When dealing with conflicts, you can first perform an update and any conflicting changes The update operation will correctly update conflicts that are auto−mergeable (i.e. files content changes do not overlap) but will skip files that contain changes that overlap. Alternatively, conflicts can be handled using a Compare editor. A Compare editor can be opened by double−clicking (or single−clicking if you have change your open strategy in the preferences) on the conflict or by choosing **Open in Compare Editor** from the context menu. The Compare editor allows you to manually resolve the conflicts in the file. Once completed, perform a **Mark**

as Merged on the conflict to indicate that you are done. This will change the conflict into an outgoing change.

Problem Markers

The Synchronize view will show *error* or *warning* problem markers on any change that appears in the view or any folder or project that appears in the view that contains a resource that has such a problem marker. This is done to prevent resources with problems from getting committed to the repository.

Toolbar

Synchronize

This command allows you to repeat the current synchronization which refetches the remote state of the resources. The drop down of the toolbar item allows you to select other existing synchronizations or create new ones using the **Synchronize...** menu command.

Pin Current Synchronization

This command allows you to pin or unpin the current synchronization. Pinned synchronizations will not be replaced by the next synchronization of the same type while unpinned synchronizations will be replaced. This allows you to keep a synchronization around for easy access. Such a synchronization will automatically update when changes to Workbench resources take place and can be configured to perform scheduled refreshes in order to keep up−to−date with the remote state of the resources.

Go to Next Difference

This command will go to the next difference displayed by the view. All the differences within a single file will be visited before opening the next file in the view.

Go to Previous Difference

This command will go to the previous difference displayed by the view. All the differences within a single file will be visited before opening the previous file in the view.

Collapse All

Collapse all the expanded entries in the view.

Incoming Mode

In this mode, only resources which have been changed in the repository since they were last loaded or synchronized with those in the Workbench (incoming changes) are visible.

Outgoing Mode

In this mode, only resources which have been modified in the Workbench (outgoing changes) are visible.

Incoming/Outgoing Mode

In this mode both incoming and outgoing changes are shown, and you can both update and commit. The advantage to using this dual mode is you can do either task as you choose. The disadvantage is that performing a simultaneous bi−directional merge is often complicated.

Conflicts Mode

In this mode only conflicts (resources modified both in the Workbench and in the repository) are shown.

Update All Incoming Changes

This command updates all the incoming changes visible in the view. The command will also update conflicts that are auto−mergeable but will skip files whose contents contain conflicts.

Commit All Outgoing Changes

This command commits all the outgoing changes visible in the view. Conflicts are not included in the commit.

Change Sets

Change sets can be enabled in incoming mode and outgoing mode.

- In incoming mode, Change Sets shows incoming changes grouped by commit comment which is handy for seeing who released what and why.
- In outgoing mode, the user can create Change Sets for grouping related changes together. Grouping is performed using various commands in the context menu.

Drop Down Menu

The drop down menu allows you to remove the current or all synchronizations, change the mode or layout, configure the current synchronization to perform scheduled refreshes as well as set other view preferences.

Context menu

From the context menu of the Synchronize view you can perform a number of interesting operations.

Open in Compare Editor

This command open the selected change in a Compare editor which allows you to inspect the changes within the file.

Open

This command open the local resource for the selected change in the default Workbench editor for the file type.

Open With

This command allows you to open the local resource for the selected change in the a Workbench editor.

Synchronize

This command refreshed the remote state of the selected resources.

Remove From View

This command removes the selected resources from the view. The resources will only reappear if the state of the removed resources changes or the Workbench is restarted.

Update

This command updates the selected resources. Conflicts can be included in an update but only auto−mergeable conflicts will be updated. Non−mergeable conflicts will be skipped and should be merged manually using a compare editor.

Commit

This command commits the selected resources. Only outgoing changes can be committed.

Override and Update

This command operates on conflicts and outgoing changes and replaces the local contents of those resources with the contents from the server.

Override and Commit

This command operates on conflicts and incoming changes and overwrites the server contents with the contents from their local counterparts.

Mark as Merged

This command adjusts the CVS timestamps of conflicting changes so they become outgoing changes. This command should be performed after conflicts are merged manually using a compare editor.

Clean Timestamps

This command adjusts the timestamps of outgoing changes whose contents already match the contents of the corresponding resource on the server so that the file is no longer an outgoing change. This is useful in situations where build tools regenerate files whose contents have not changed.

Related concepts

Team programming with CVS
Synchronizing with a CVS repository

Related tasks

Synchronizing with the repository
Updating
Resolving conflicts
Merging from a branch
Committing
Comparing resources
Merging changes in the compare editor

Related reference

CVS
Compare editor

Merging from a branch

After creating and working in a CVS branch for some time, you may want to merge your changes from the branch into another branch, or into HEAD. To do this, you will need to know:

1. The name of the branch or version that contains your changes.
2. The version from which the branch was created. This is the version name that you supplied when branching.

To merge your changes:

1. Ensure that the destination is loaded into your workspace. For example, if you wish to merge your changes into HEAD, make sure the project is shared with HEAD in your workspace. To do this, select the project and choose **Replace With > Another Branch or Version** from the context menu. Then select the branch to replace with.
2. Select the project and choose **Team > Merge**.
3. Enter the branch or version which contains the changes you want to merge.
4. The merge dialog will try to guess an appropriate start point. If that fails, enter the start point of the merge. This is the version from which the branch was created. Click **Finish**.
5. A CVS Merge synchronization will be added to the Synchronize view, showing all differences between your workspace and the branch with the changes.
6. Load all of the desired changes into the workspace. This may be accomplished by either manually merging changes, or by choosing **Update**, **Override and Update**, or **Mark as Merged** from the tree's context menu.
7. After all desired changes are in the workspace, choose **Team > Synchronize with Repository**. You may then commit all the changes to the repository.

Tip: If you don't know the start point of the merge, you can choose to perform the merge directly into the workspace. This relies on the CVS server to pick an appropriate start point. Because the merge is performed directly into the workspace, CVS may introduce a text based conflict markup into the file. This markup is typically hard to deal with so merging directly into the workspace should only be done when necessary.

Merge actions

The actions in the merge editor complement the manual merge toolbar actions that are available in the bottom half of the merge editor.

Update	Running this action will bring the changes into the file in the workspace. Any conflicts that are not auto−mergeable will be skipped.
Override and Update	This action is enabled on files with conflicting changes. Running this action will discard any local changes you have and replace the file with the remote contents.
Mark as Merged	This action will remove the selected changes from the view. The changes will only reappear if the remote state of the resource changes and the CVS Merge Synchronization is refreshed.

Tip: You can perform ongoing merges by pinning a CVS Merge Synchronization in the Synchronize view. This will allow you to keep your workspace up−to−date with the changes released to a different branch.

Related concepts

Team programming with CVS

Branches
Synchronizing with a CVS repository

Related tasks

Branching
Synchronizing with the repository
Updating
Committing

Related reference

CVS
Merge wizard
CVS Merge Synchronization

Branching

Creating a branch and releasing resources to that branch is useful in cases where you are not yet ready to put your changes in the main development flow. It is also useful for creating incremental patches to existing versions.

To create a branch:

1. From the context menu of the project, select **Team > Branch**.
2. Enter the Branch Name for the new branch in the text area.
3. If the project in your workspace is not already a version, enter the Version Name to create in the text area.
4. If you would like to start working in the branch immediately, make sure the checkbox is selected.
5. Click **OK**.

Note: The Version Name is important; you will need this when you want to merge your changes later on. It identifies the point at which the branch was created.

Related concepts

Team programming with CVS
Branches
Synchronizing with a CVS repository

Related tasks

Synchronizing with the repository
Merging from a branch

Related reference

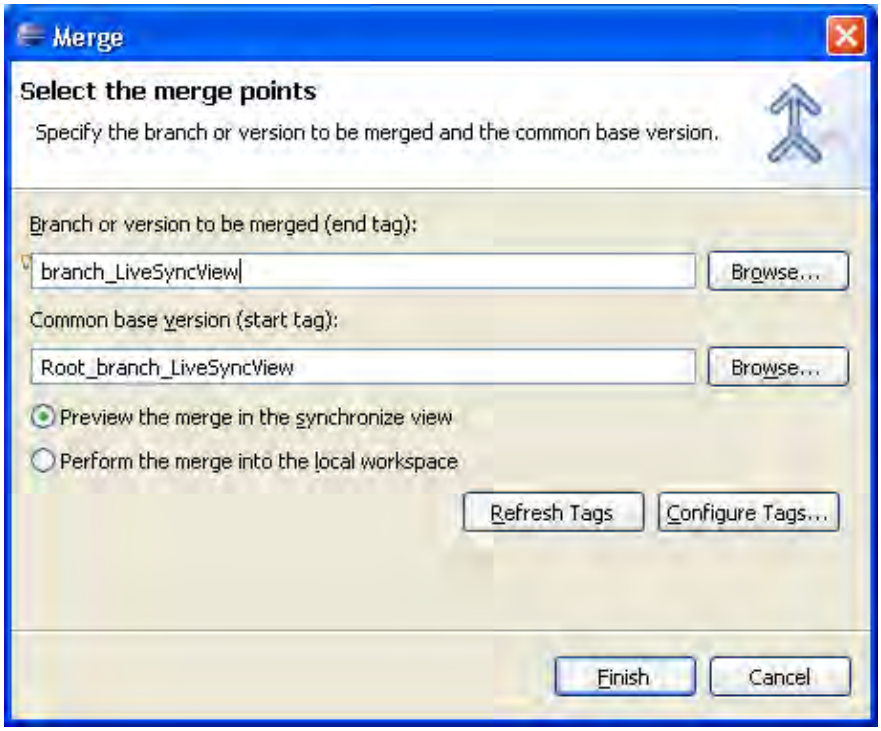
CVS

Merge wizard

This wizard helps you merge changes between two states of a project into your workspace. Often the Merge Wizard is used to move changes from one branch into another, for example after splitting a branch to work on a bug fix. The merge operation takes changes between two points in a branch, the **start point** and the **end point**, and merges them into your workspace. Typically the start point will be the root of a branch (version tag) and the end point can either be the tip (latest and best) of the branch or another version tag.

It is very important to understand that the destination of the merge is always the project in your workspace. After the merge has been completed you can test the changes locally and then commit them to the new branch (usually HEAD).

To start a merge, select a project (or one or more resources), and select **Team > Merge...** from the pop−up menu. The following dialog will appear.



First, you should choose the end tag of the merge. This is the branch or version being merged into the workspace. Choose a version when you want to merge the differences between two versions of a project into your workspace. Choose a branch if you want to merge the changes made in the branch into your local workspace.

The wizard will try to pick an appropriate tag for the start tag or base tag. If one cannot be determined, you should enter it manually. If there is no start tag, you can choose to merge without a preview, in which case a start tag is not required but the merge will happen directly into the workspace. The drawback of this is that CVS uses a text base markup to identify conflicts and this is cumbersome to work with.

If the merge preview option is selected, after the finish button is pressed, the changes between the start point and end point are calculated and displayed as a CVS Merge Synchronization in the Synchronize view. Depending on the size of the project you are merging, this may take some time but can be run in the

background. In the Synchronize view you can update or merge changes into your Workbench.

Related concepts

Team programming with CVS
Branches

Related tasks

Branching
Merging from a branch

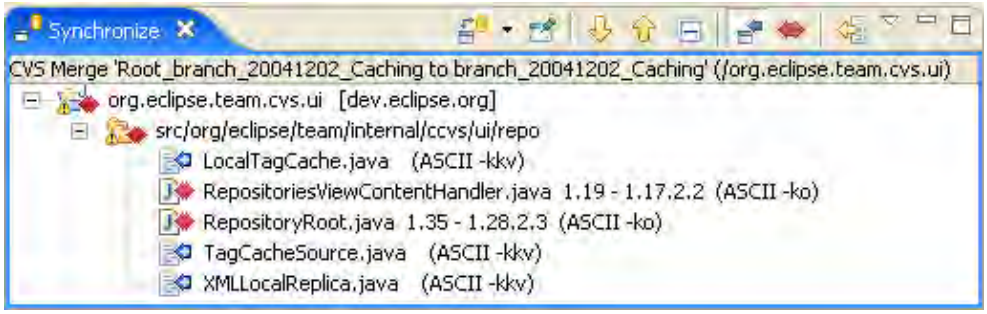
Related reference

CVS
CVS Merge Synchronization
Compare editor

CVS Merge Synchronization

CVS Merge synchronizations launched using the **Team > Merge** menu command will appear in the Synchronize view. This view allows you to inspect the differences between the local Workbench resources and their remote counterparts that are being merged and allows you to update the local resources. Committing is not supported when merging.

Here is what the CVS Merge Synchronization in the Synchronize view looks like:



Features

The features of the CVS Merge Synchronization are similar to those of the CVS Workspace Synchronization with the following differences.

- **Modes:** When merging, only incoming and conflicting changes are possible. As such, only the Incoming Mode and Conflicts mode are supported.
- **Operations:** Only updating is supported when merging.
- **Ongoing merges:** In order to compensate for the poor ongoing merge support of CVS, a CVS Merge Synchronization can be pinned and kept indefinitely. The timestamps of merged changes are recorded and, on refresh, only new changes since the last merge are shown in the view.

Toolbar

Synchronize

This command allows you to repeat the current synchronization which refetches the remote state of the resources. The drop down of the toolbar item allows you to select other existing synchronizations or create new ones using the **Synchronize...** menu command.

Pin Current Synchronization

This command allows you to pin or unpin the current synchronization. Pinned synchronizations will not be replaced by the next synchronization of the same type while unpinned synchronizations will be replaced. This allows you to keep a synchronization around for easy access. Such a synchronization will automatically update when changes to Workbench resources take place and can be configured to perform scheduled refreshes in order to keep up−to−date with the remote state of the resources. This is especially handy for ongoing merges from a more stable branch (e.g. HEAD) into a development branch.

Go to Next Difference

This command will go to the next difference displayed by the view. All the differences within a single file will be visited before opening the next file in the view.

Go to Previous Difference

This command will go to the previous difference displayed by the view. All the differences within a single file will be visited before opening the previous file in the view.

Collapse All

Collapse all the expanded entries in the view.

Incoming Mode

In this mode, all resources which have been changed in the branch being merged are visible.

Conflicts Mode

In this mode only conflicts (resources modified both in the Workbench and in the branch being merged) are shown.

Update All Incoming Changes

This command updates all the incoming changes visible in the view. The command will also update conflicts that are auto−mergeable but will skip files whose contents contain conflicts.

Drop Down Menu

The drop down menu allows you to remove the current or all synchronizations, change the mode or layout, configure the current synchronization to perform scheduled refreshes as well as set other view preferences.

Context menu

From the context menu of the Synchronize view you can perform a number of interesting operations.

Open in Compare Editor

This command open the selected change in a Compare editor which allows you to inspect the changes within the file.

Open

This command open the local resource for the selected change in the default Workbench editor for the file type.

Open With

This command allows you to open the local resource for the selected change in the a Workbench editor.

Synchronize

This command refreshed the remote state of the selected resources.

Remove From View

This command removes the selected resources from the view. The resources will only reappear if the state of the removed resources changes or the Workbench is restarted.

Update

This command updates the selected resources. Conflicts can be included in an update but only auto−mergeable conflicts will be updated. Non−mergeable conflicts will be skipped and should be merged manually using a compare editor.

Override and Update

This command operates on conflicts and outgoing changes and replaces the local contents of those resources with the contents from the server.

Mark as Merged

This command removes the change from the view. This command should be performed after conflicts are merged manually using a compare editor. The change will only reappear if the view is refreshed and new changes are found in the branch being merged.

Related concepts

Team programming with CVS

Related tasks

Resolving conflicts
Merging from a branch
Comparing resources
Merging changes in the compare editor

Related reference

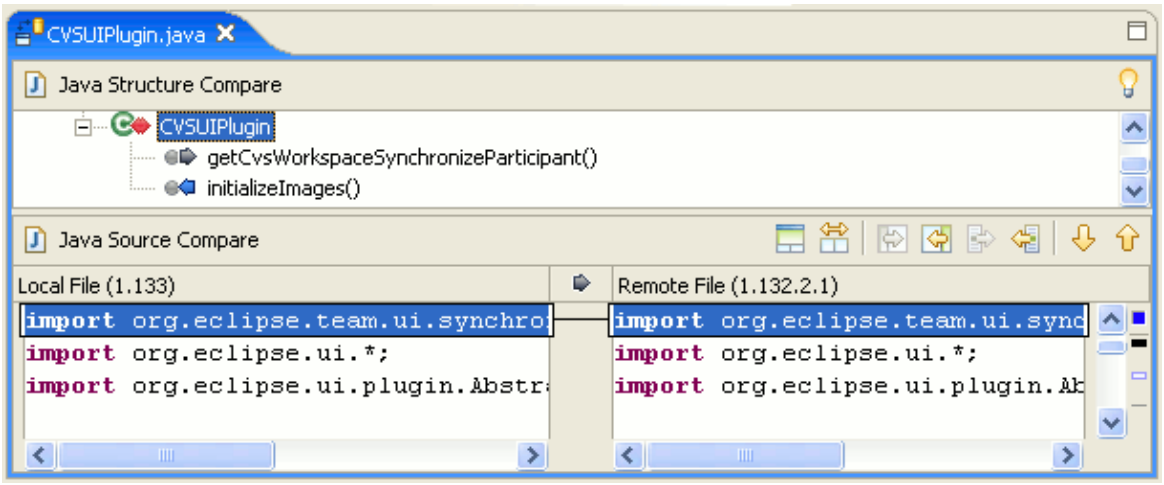
CVS
Merge wizard
CVS Workspace Synchronization
Compare editor

Compare editor

You can view the differences between two files by comparing them. You can compare different files, you can compare versions in the Workbench with versions in the repository, or with the local edit history. In some cases you can compare three files (when a common ancestor exists).

After a comparison is carried out, the compare editor opens in the editor area. In the compare editor, you can browse through all the differences and copy highlighted differences between the compared resources. You can save changes to resources that are made in the comparison editor.

Here is how the compare editor looks when you open it from a CVS Workspace synchronization..



Toolbar

The toolbar of the Compare editor includes the following buttons:

Control Visibility of Ancestor Pane
: There are two conditions under which a three way compare will occur, both when using the Team version management support: when comparing a file that is in conflict, and when comparing a file being merged from a branch. In both cases, the system will determine a common ancestor in the repository to compare the conflict or merge against. This button determines the visibility of the third editor. By default, the ancestor pane is not visible.

Perform Three way/Two way Compare
: The compare editor can be toggled between performing a three way compare or a two way compare which ignores the common ancestor.

Copy All from Left to Right
: Copies the entire contents of the file in the left pane into the file in the right pane, making the contents of the two files identical.

Copy All Non−Conflicting Changes from Right to Left
: Copies all the non−conflicting changes from the right pane into the left pane. Conflicting changes must be copied individually.

Copy Current Change from Left to Right
: Merges changes in two files by copying the highlighted change in the left pane into the highlighted fragment on the right. This will overwrite the highlighted fragment in the right pane.

Copy Current Change from Right to Left

Does the opposite of the one just described.

Select Next Difference

Highlights the next difference that is found between the compared resources.

Select Previous Difference

Highlights the previous difference that is found between the compared resources.

Related concepts

Synchronizing with a CVS repository
Three way comparisons

Related tasks

Comparing resources
Synchronizing with the repository
Merging changes in the compare editor
Resolving conflicts
Setting preferences for comparing files
Comparing resources with repository versions

Setting preferences for comparing files

When you select to compare or synchronize two or more resources in the Workbench, one or more comparison editors usually open. To customize how these editors behave:

1. On the main menu bar, select **Window > Preferences**. The Preferences window opens
2. Expand the **General** category on the left and select **Compare/Patch**.
3. Set your preferences and click **OK**.

You can configure the following options on the **General** tab.

Option	Description	Default
Open structure compare automatically	Makes an additional information area visible which shows differences in the underlying structure of the resources being compared. This information may not be available for all comparisons.	On
Show additional compare information in the status line	Causes the status line to display additional context information about the comparison.	Off
Ignore white space	Causes the comparison to ignore differences which are whitespace characters (spaces, tabs, etc.). Also causes differences in line terminators (LF versus CRLF) to be ignored.	Off
Automatically save dirty editors before patching	This option controls whether any unsaved changes are automatically saved before a patch is applied.	Off

You can configure the following options in the **Text Compare** tab.

Option	Description	Default
Synchronize scrolling between panes in compare viewers	The two compare viewers will "lock scroll" along with one another in order to keep identical and corresponding portions of the code in each pane side by side.	On
Initially show ancestor pane	Sometimes you may want to compare two versions of a resource with the previous version from which they were both derived. This is called their *common ancestor*, and it appears in its own comparison pane during a three way comparison.	Off
Show pseudo conflicts	Displays conflicts that occur when two developers make the same change, for example when both add or remove the same line of code.	Off
Connect ranges	Controls whether differing ranges are visually	On

with single line	connected by a single line or a range delimited by two lines.	

Related concepts

Three way comparisons

Related tasks

Comparing resources
Synchronizing with the repository
Merging changes in the Compare editor

Related reference

Compare editor

Comparing resources with repository versions

To compare Workbench resources with those in the repository:

1. Select a resource in one of the navigation views.
2. From the resource's pop−up menu, select one of the following menu items:
    - **Compare With > Latest From <Branch Name>** or **<*Version*>**− Compare the Workbench resource with the latest resources currently committed to the branch that the local project is shared with or if the local project is checked out as a version then compare against the version. A compare editor will be opened. This editor will display a tree of all resource differences between the workspace resource and the latest resources in the repository.
    - **Compare With > Another Branch or Version** − Compare the Workbench resource with a specific version or branch that you select in the repository. A compare editor will be opened. The editor will display a tree of all resource differences between the workspace resource and the resources in the specified version or branch.
    - **Compare With > Revision** − (available only on a file) Compare the Workbench file with another revision of the file. A compare editor will be opened. The editor will display a table of all available revisions of the selected file. Selecting one of the revisions will show the differences between the workspace revision and the selected revision.

Tip: When comparing with a branch or version, you can specify a particular date instead of a a version or branch tag. This can be done by right clicking on the ***Dates*** category in the tag selection list and choosing ***Add Date***. A Date Tag dialog is displayed that allows you to specify a date and optionally a time. After you click ***Ok***, the date tag will appear in the tag selection list.

Related concepts

Three way comparisons
Synchronizing with a CVS repository

Related tasks

Synchronizing with the repository
Setting preferences for comparing files

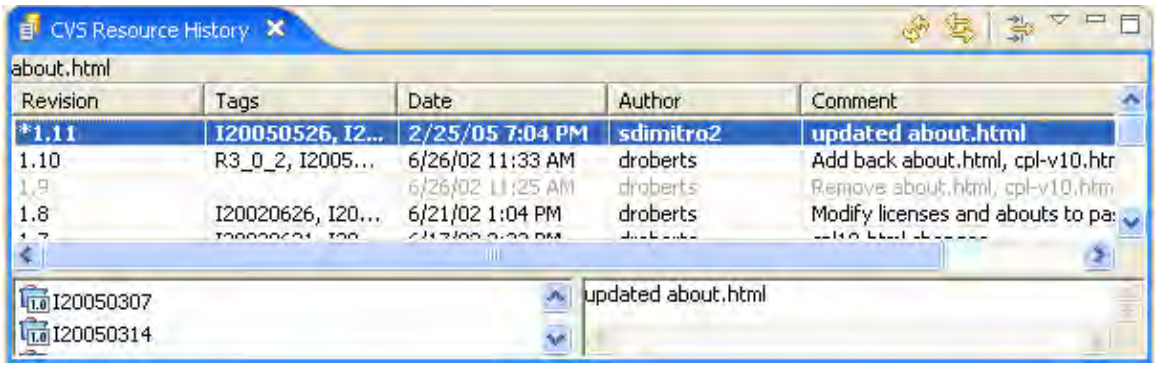
Related reference

CVS
Compare editor

CVS Resource History view

This view provides a list of all the revisions of a resource in the repository. From this view you can compare two revisions, replace (get sticky) or revert (get contents) the corresponding workspace file to the revision, or open an editor on a revision.

Here is what the CVS Resource History View looks like:



Columns

For each resource, the following is displayed in the table:

Revision

This column displays the revision number in the history.

Tags

The tags that are associated with the revision. Selecting a revision line will list the tags in the lower left pane of the view.

Date

This column displays the creation date and time of the revision in the history.

Author

This column displays the name of the person who created and authored the version.

Comment

This column displays the comment (if any) supplied for this revision at the time it was committed. Selecting a revision line will show the complete comment in the lower right pane of the view.

Toolbar

Refresh View

This command refreshes the contents of the view, fetching the latest history information for the resource from the server.

Link with Editor

When enabled, the view will display the history for the resource of the active editor.

Filter History

This command will open a dialog that allows the specification of view filters. The resource history can be filtered by author, comment or date.

Drop Down Menu

The drop down menu in the toolbar allows the comment and tag panes to be hidden or shown.

Context menu

From the context menu of the CVS Resource History view you can perform a number of interesting operations.

Get Contents

This command will load the contents of the selected revision into the local copy of the file whose history is displayed in the view. The local file can then be committed to make the contents of latest revision in HEAD (or a branch) match the contents of the selected revision.

Get Sticky Revision

This command will load the the selected revision over the local file whose history is being displayed. This operation is rarely needed and is mainly used to change the tag on one or more resources. The loaded revision cannot be subsequently committed but can be tagged.

Tag with Existing...

This command allows an existing tag to be moved to the selected file revision.

Show Annotation

This command shows the contents of file with annotations identifying the author of each line of code in the file.

Compare

This command opens an editor comparing the contents of the two selected revisions.

Open

This command opens an editor displaying the contents of the selected file revision.

Refresh View

This command refreshes the contents of the view.

Related reference

CVS Repositories view

Related concepts

Resources
CVS Repositories
Local history

Discovering branch and version tags

*Prerequisite:*Before you can discover the branch and version tags for a repository, you must have a valid repository location in the CVS Repositories view.

To discover existing branch tags for multiple folders in a single repository location:

1. Switch to the CVS Repository Exploring perspective or add the CVS Repositories view to the current perspective.
2. In the CVS Repositories view, select a repository location.
3. From the pop−up menu for the CVS Repositories view, select **Refresh Branches**. The Refresh Branches dialog will open.
4. In the folder list, select the folders whose tags you would like to refresh. Alternatively, at the bottom of the dialog, you can choose a working set and all remote folders whose names match the name of a project in the working set will become checked.
5. Click **Finish** to discovered the tags for the checked projects. These tags will be added to the repositories view.

Note: This operation may be long running especially if the number of selected folders is large.

Tip: If the above operation does not result in the discovery of any tags, it is probable that the folder in the repository does not have a .project file in it. The .project file is the default file used by the tag discovery process. The file can be changed for individual projects using ***Configure Branches and Versions*** (see below).

To discover existing branch and version tags for a single project folder:

1. Switch to the CVS Repository Exploring perspective or add the CVS Repositories view to the current perspective.
2. In the CVS Repositories view, expand the repository location.
3. Expand ***HEAD*** and select the root folder whose tags are to be discovered.
4. From the pop−up menu for the CVS Repositories view, select **Configure Branches and Versions**. The Tag Configuration dialog will open.
5. In the top pane on the left (titled *Browse files for tags*), expand one or more projects to find a file that is known to have tags that should be added to the CVS Repositories view.
6. Select a file in the left top pane. If the file selected has tags defined on it, they will be displayed in the right top pane (titled *New tags found in selected files*).
7. Check or uncheck the tags in the right pane as desired.
8. Click **Add Checked Tags** to add the checked tags to the list of remembered tags.
9. Click **OK** to add the remembered tags to the selected repository location or project.

Note: You can also set the auto−refresh files by selecting a file in the files pane and clicking ***Add Selected Files*** in the lower right hand portion of the dialog.

Related concepts

Team programming with CVS
CVS Repositories
Branches
Versions
Synchronizing with a CVS repository

Related tasks

Creating a CVS repository location
Checking out a project from a CVS repository
Sharing a new project using CVS

Versioning projects in the repository

You can version a project in the repository without adding the project to the Workbench.

1. Switch to the CVS Repository Exploring perspective or add the CVS Repositories view to the current perspective.
2. In the CVS Repositories view, expand a repository location and find the project you wish to version.
3. Right−click on the project and select **Tag as Version**.
4. Enter the name of the version you wish to create.
5. Click **OK** to close the dialog and create the version.

Tip: You can browse existing versions of your project by clicking on the **Details** button in the Tag Resources dialog. Clicking **Refresh from Repository** should populate the Existing Versions list. It is sometimes helpful to see existing versions before naming your new version.

Tip: If the version tag you wish to apply already exists on some of the resources to be versioned, you can perform a **Tag with Existing** instead. This operation will move an existing tag to the selected resources and can be used to move both version and branch tags. If the tag you wish to apply is not show in the list, you can click **Refresh from Repository**. If this doesn't find the tag, you can click **Configure Tags** which opens a dialog that allows you to search for tags on specific files in the repository.

Related concepts

Branches

Related tasks

Creating a version of a project
Sharing a new project using CVS
Viewing a file's revision history
Branching

Related reference

CVS

Enabling the CVS resource decorations

When enabled, CVS will decorate resources in Workbench views with icon and label decorations that indicate the CVS state of the resource.

To enable the CVS decorations:

1. From the main menu bar, select **Window > Preferences** to open the Preferences dialog.
2. In the dialog, select the **General > Appearance > Label Decorations** preference page. This page allows the enabling and disabling of all decorations defined by different plug−ins in the Workbench.
3. Click on the CVS box to enable the CVS decorations.
4. Click **OK**.

Note: In some cases, the decorations may not appear immediately due to the fact that they are expensive to compute. In those cases, they are calculated in the background.

To configure the CVS decorations:

1. From the main menu bar, select **Window > Preferences** to open the Preferences dialog.
2. In the dialog, select the **Team > CVS > Label Decorations** preference page. This page allows the configuration of how decorations appear on CVS projects, folders and files.
3. To configure the text label decorations click the ***Text*** tab and then:
    a. Pick the resource type (file, folder, project) for which you want to change the decoration.
    b. Click in the decoration string for that resource type at the location where the new decoration element should go.
    c. Click the **Add Variables** button for that resource type.
    d. Choose the CVS information variable to add. For instance, for files you could select the *added_flag* which will be inserted in the decoration where you placed the cursor. The string *{added_flag}* will be added to the decoration string which will decorate newly added resources using the added resource label (which can be set in the *Label Decoration for Added* field).
4. Repeat the above for any other decorations to be added.
5. Decorations can be removed by deleting the decoration variable name (i.e. *{added_flag}*) from the decoration string. In a similar manner, spacing can be added.
6. To configure icon decorations, click the ***Icon*** tab simply enable/disable the icon checkboxes.
7. Click **OK** or **Apply** for the new decorations to take effect.

Note: Under the ***General*** tab is an option to disable the deep dirty decoration of folders. This will make decorator determination more efficient but will make it harder to find dirty resources in one of the navigation views. Under the ***Synchronize View*** tab, the Synchronize view can be configured to show the synchronization state of a resource as part of the text label displayed in the Structure Compare pane.

You can also change the color and font of items based on the item's CVS state. To set the color and font:

1. From the main menu bar, select **Window > Preferences** to open the Preferences dialog.
2. In the dialog, select the **Team > CVS > Label Decorations** preference page and choose to *Enable font and color decorations*.
3. Next, select the **General > Appearance > Colors and Fonts** preference page.
4. In the ***CVS*** category are the various CVS states for which you can set the colors and fonts. Set them as desired.

5. You can return to the **Team > CVS > Label Decorations** preference page to preview the changes.
6. Click **OK** or **Apply** for the new decorations to take effect.

Related concepts

Team programming with CVS
Synchronizing with a CVS repository

Related tasks

Checking out a project from a CVS repository
Sharing a new project using CVS

Related reference

CVS
CVS Label Decorations

CVS Label Decorations

Label Decorations are used by CVS to show important synchronization information about workspace resources. Decorations can effect the text or the icon of the label associated with a resource. The text decorators are configurable on the CVS Label Decorations Preference page which can also be used to indicate which icons are to be enabled. Here is a description of the icons used by CVS.

Decoration	*Resource type*	*Description*
	Any	Indicates that the resource is under version control.
	Any	Indicates that the resource or one of its children contains outgoing changes.
	File	Indicates that the project containing the file has been configured to use watch/edit and the file is being edited locally.
	File	Indicates that the file contains a merge conflict which has not yet been resolved. The conflict is considered resolved once the file is modified and saved.
	Project or Folder	Indicates that the folder is under version control but does not correspond to an existing remote directory but is instead most likely a module defined in the CVSROOT/modules file.
	File or Folder	Indicates that the resource is not under version control.

Note: Resources that are ignored by CVS will not be decorated by any of the above CVS decorations.

Related concepts

Label Decorations

Related tasks

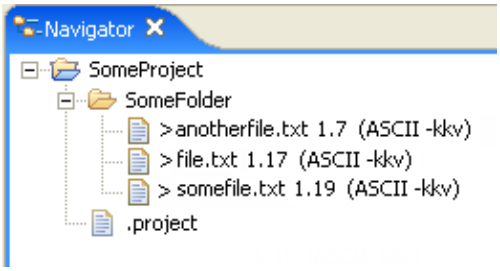
Enabling the CVS resource decorations

Label decorations

Label decorations are used to show extra information about an item by modifying its label or icon. They can be used to obtain information about the state of an item without having to look at its properties in the Properties view or open its Properties dialog.

A number of label decorators may be available. To control which decorators are visible, go to **Window > Preferences > General > Appearance > Label Decorations**. This preference page provides a selectable list and description of the available decorations.

The following is the Navigator view showing resources with CVS decorations:



Moving version tags

Warning: Although many people would prefer a CVS version to be frozen in time and not be modifiable, version and branch tags in CVS are mutable. As a result many are convinced that modifying a version is bad practice, but there are a couple of scenarios where it actually comes in handy. With that said, please be cautious when moving tags around.

Moving a tag on a single file

Let's say that you have just submitted your build by versioning your project as R1. But you soon after discover that there is a small change to a file that should be made and included in the build. Instead of having to re−version the project you can move the R1 version tag for the modified file.

1. Modify the file(s) Select the file that was modified after R1 was created and from the context menu select ***Team > Show in Resource History***.
2. From within the Resource History view select the revision that should be marked with the R1 version.
3. From the context menu select ***Tag with Existing...***.
4. Select the R1 version from the dialog box and press OK.
5. The resource history view will be updated to confirm that the version had been moved.

Moving a tag from within the repositories view

Many projects use a well defined version name for their current stable lineup in HEAD. For example, by versioning HEAD with the STABLE tag the build scripts could simply checkout the STABLE version for builds. As the code evolves the STABLE tag is moved regularly to mark the most current stable lineup. The repositories view provides an action for moving an existing tag.

1. Open the Repositories view and select a resource.
2. From the context menu select **Tag with Existing**
3. A tag selection dialog will appear from which you can select the tag to move. If the tag you wish to apply is not show in the list, you can click **Refresh from Repository**. If this doesn't find the tag, you can click **Configure Tags** which opens a dialog that allows you to search for tags on specific files in the repository.
4. Then press **OK** and your tag will be moved. The operation will move an existing tag to the selected resources and can be used to move both version and branch tags.

Related concepts

Team programming with CVS

Related tasks

Synchronizing with the repository
Committing

Refreshing the CVS Repositories view

There are two ways to refresh the CVS Repositories view in order to see the current connection status of all repositories:

- Click the **Refresh View** button on the view's toolbar
- Press F5 on the keyboard

Note: Refreshing the CVS Repositories view may change the resources displayed by the view but does not change the branch and version tags known to the view. Refreshing the tags is discussed in Discovering branch and version tags

Related concepts

Team programming with CVS
CVS Repositories

Related tasks

Creating a CVS repository location
Discarding a CVS repository location
Discovering branch and version tags

Related reference

CVS
CVS Repositories view

Changing the properties of a CVS repository location

You can change the connection properties of an existing repository location.

To change the properties of a CVS repository location:

1. Switch to the CVS Repository Exploring perspective or add the CVS Repositories view to the current perspective.
2. In the CVS Repositories view, select the repository location whose properties you want to change.
3. Select **Properties** from the pop−up menu. The properties dialog will open.
4. Select the ***CVS*** tab.
5. Change one or more of the location's properties.
6. Click **OK** to change the properties.

Note: The properties will be changed for all projects shared with the selected repository location.

Related concepts

Team programming with CVS
CVS Repositories

Related tasks

Creating a CVS repository location
Changing the sharing of a Project
Changing the encoding of a CVS repository location

Related reference

CVS
CVS Repositories view

Changing the sharing of a project

If a project is shared with a CVS repository location, you can change the sharing information so it is shared with a different repository location. The new location must map to the same repository but can have a different connection method and/or user name.

To change the sharing of a project:

1. In one of the navigation views, select the project whose sharing information is to be changed.
2. Select **Properties** from the pop−up menu. The properties dialog will open.
3. Select **CVS** to see the CVS properties page for the project.
4. Click **Change Sharing...**. A repository selection dialog opens containing the compatible repository locations. (***Note***: You can change the sharing to a non−compatible repository location by unchecking the ***Show only compatible repository locations*** button.)
5. Select the desired location and click **OK** to change the sharing for the project.

Related concepts

Team programming with CVS
CVS Repositories

Related tasks

Creating a CVS repository location
Changing the properties of a CVS Repository Location

Related reference

CVS
CVS Repositories view

Changing the encoding of a CVS repository location

You can change encoding of an existing repository location. This encoding will be used to translate file paths and commit messages from the client encoding to the server encoding and vice versa. The encoding does not affect the contents of file. You may use the editor encoding (available from the ***General>Editors*** preference page) or resource encodings (available from the ***Properties>Info*** page of resources) to configure the Workbench to use the proper file encodings.

To change the server encoding of a CVS repository location:

1. Switch to the CVS Repository Exploring perspective or add the CVS Repositories view to the current perspective.
2. In the CVS Repositories view, select the repository location whose properties you want to change.
3. Select **Properties** from the pop−up menu. The properties dialog will open.
4. Select the ***Server Encoding*** tab.
5. Choose to set the encoding and either choose or type in the desired encoding.
6. Click **OK** to change the properties.

Related concepts

Team programming with CVS
CVS Repositories

Related tasks

Creating a CVS repository location

Related reference

CVS
CVS Repositories view

Working with patches

Patches allow developers to share work without storing it in a repository. This is helpful when a developer wants to contribute to a project that is shared through a repository but does not have write access to the repository. In this situation, the developer can create a patch and either e−mail it to a developer who does have write access or attach it to a bug in the bug reporting system used by the project, depending on the process defined by the project. A developer that does have write access can then apply the patch to the project and commit the changes.

To create a patch from a CVS project:

1. Select the resource that contains the modifications to be included in the patch. Although this can be any folder, it is easiest to select the project itself because the patch must be applied to the same resource it is generated from. The patch should also be applied to the same file revisions that it is generated on so steps should be taken to ensure that the patch is applied to the same resource line−up (the easiest way to do this is to create the patch on top of a version).
2. From the popup menu, select **Team > Create Patch...**. The Create Patch wizard will open.
3. Choose where the patch should be saved:
    a. *Save to Clipboard* − this will place the patch on the clipboard so it can be pasted into a text editor such as an e−mail program.
    b. *Save to File System* − this will place the patch in the specified file in the local file system
    c. *Save in Workspace* − this will place the patch in the specified file inside one of the existing workbench projects.

   For small patches it may be reasonable to transfer the patch using the clipboard but in most cases the local file system in the best option to use. Click **Next** to configure how the patch is generated.
4. Choose how to configure the patch:
    a. *Recurse into sub−folders* − If disabled, only the direct children of the selection are included in the patch. Otherwise, all descendants are included.
    b. *Include new files in patch* − If disabled, only files that are under CVS version control are included. Otherwise, files that have been newly created but not added or ignored will also be included.
    c. *Diff output format* − Allows the choice of several common diff output formats. *Unified* is the format used by many patch application tools including Eclipse.
5. Click **Finish**.
6. Transfer the patch as appropriate for the project being patched.

To apply a patch:

1. Select the resource that the patch was generated on. This resource should contain the same file revisions as the line−up on which the patch was generated.
2. From the pop−up menu, select **Team>Apply Patch...**. The Resource Patcher wizard will open.
3. Indicate where the patch is to be found:
    a. *File* − the patch is in a file on the local file system. Either type in the full path to the file or use the **Browse...** button to find the file.
    b. *Clipboard* − the patch is on the clipboard. **Warning:** It is safer to use a file based patch. Line endings may not be handled properly if the clipboard is used and the patch was generated on a different platform (i.e. Linux vs. Windows).

   Click **Next** to see the effect of applying the patch.

4. The top pane of this page shows whether the patch could be successfully applied to files in your workspace. If you select a leaf item in the tree the bottom pane shows the part of the patch file (known as 'hunk' in patch terminology) in an easy to read side by side presentation. **Note:** The bottom pane **does not** show a preview of how resources in your workspace would look after applying the hunk. It just shows the contents of the patch file.
    a. A checked item indicates that a patch (or hunk) could be successfully applied to a workspace resource. You can exclude patches or individual hunks by unchecking them.
    b. A red exclamation mark indicates that there is a problem with a patch or hunk.
    This happens if the patch is not well formed or the revision of one or more files that the patch were generated on do not match the revisions that the patch is being applied to.
    You find the reason for the failure in parenthesis.

    In order to apply the full patch successfully you will have to eliminate the problems (red exclamation marks) and get checked items everywhere by tweaking the options on this wizard page (see 'Options' below).
5. If all is well, click **Finish** to apply the patch. The workspace will now contain outgoing changes for each file modified by the patch.

Options for applying a patch

For getting successful matches of your patch file you have the following options:

1. Go back to the first page of the Resource Patcher wizard and select the correct resource to which the patch should be applied.
2. If a common prefix of the path names stored in the patch file doesn't match the path names in you current workspace, you can 'Ignore leading path name segments'.
3. Use the 'Ignore whitespace' option to make the matching process independent from whitespace differences between the patch file and files in your workspace.
4. Adjust the 'Maximum fuzz factor' (patch terminology). This factor determines how far from its original line a hunk is allowed to match. The default is two. So if a hunk does not match at the line given in the patch file, the Resource Patcher tries to match the hunk 'fuzz' number of lines before or after the position.
5. Use the 'Reverse patch' option for patch files that already have been applied to your workspace. This option is also useful to undo or redo a patch.

Related concepts

Team programming with CVS

Related tasks

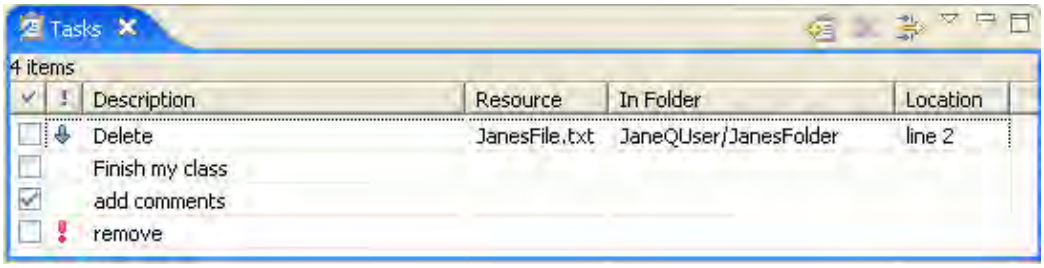
Comparing resources

Related reference

CVS

Tasks view

You can assign tasks within your project by right clicking marker bar's context menu and selecting ***Add Task*** or you can add items within the Tasks view by selecting ***Add Task***  . For example, if you would like to record reminders to follow up on something later, add it to the tasks view. When you add a task, you have the option of associating it with a resource so that you can use the Tasks view to quickly open that resource for editing.



The first column of the Tasks view displays an icon that denotes the type of line item. You can sort and filter line items in the task view, to view only high−priority tasks or tasks associated with a particular resource or group of resources.

By default, the Tasks view is included in the Resource perspective. To add it to the current perspective, click **Window > Show View > Other > Basic > Tasks**.

Related concepts

Markers
Bookmarks
Problems view

Related tasks

Opening views
Moving and docking views
Adding line items in the Tasks view
Associating a task with a resource
Filtering the Tasks view
Deleting tasks

Related reference

Tasks view

Markers

Markers are objects that may be associated with Workbench resources. There are many uses of markers in the Workbench. The three main uses of markers in the Workbench are:

- Tasks
- Problems
- Bookmarks

Markers are shown in a marker view (Tasks, Problems or Bookmark view) or on the marker bar in the editor area.

The following sections give more detail on each of these marker types.

Tasks

A task marker represents a work item. There are two kinds of tasks:

- Automatically−generated information associated with the resource
- User specified tasks that may or may not be associated with a resource

Both of these task types appear in the Tasks view.

Problems

Problem markers represent invalid states and are categorized as follows..

- ***Errors***: Error markers are often used to indicate the source location of syntax or compilation errors.
- ***Warnings***: Warning markers indicate the source location of, for example, compilation warnings.
- ***Information***: Information markers indicate the source location of information only tasks.

Problems are shown in the Problems view.

Bookmarks

Bookmarks place an anchor either at a specific line in a resource or on the resource itself. They are shown in the Bookmarks view.

Related concepts

Bookmarks
Tasks view
Problems view

Related tasks

Creating a bookmark within a file
Creating a bookmark for an entire file

Bookmarks

You can place an "anchor" either on a resource within the Workbench, or at a specific line within a file, by creating a bookmark. Then you can use the Bookmarks view to return to those files quickly.

The Bookmarks view (**Window > Show View > Bookmarks**) displays all bookmarks that you have created.

Related concepts

Tasks view

Related tasks

Creating a bookmark for an entire file
Creating a bookmark within a file
Deleting a bookmark
Adding line items in the Tasks view
Associating a task with an editable resource

Related concepts

Markers
Tasks view
Problems view

Related reference

Bookmarks view

Creating a bookmark for an entire file

You can bookmark individual files in the Workbench, in order to open them quickly from the Bookmarks view later.

1. In one of the navigation views, select the file that you want to add to your list of bookmarks.
2. From the main Workbench menu select ***Edit > Add Bookmark***.

At any time, you can open the bookmarked file for editing by double−clicking the bookmark in the Bookmarks view (***Window > Show View > Bookmarks***).

You can also create bookmarks for specific lines of text or source code or within a file. See the list of related topics below.

Related concepts

Bookmarks
Tasks view

Related tasks

Creating a bookmark within a file
Deleting a Bookmark
Adding line items in the Tasks view
Associating a task with a resource

Creating a bookmark within a file

The Workbench allows you to create bookmarks in files that you edit so that you can quickly reopen those files from the Bookmarks view.

1. With the file open in an editor, right−click in the gray border at the left of the editor area, next to the line of code or text that you want to bookmark.
2. Select **Add Bookmark** from the pop−up menu.
3. Notice that an icon for the bookmark now appears in the left border of the editor area. A line is also added to the Bookmarks view (**Window > Show View > Bookmarks**.)

You can reopen the file for editing at any time by double−clicking the bookmark in the Bookmarks view.

Related concepts

Bookmarks
Tasks view

Related tasks

Creating a bookmark for an entire file
Deleting a Bookmark
Adding line items in the Tasks view
Associating a task with a resource

Deleting a bookmark

To delete any bookmark:

1. Open the Bookmarks view.
2. Right−click the bookmark that you want to delete and select **Delete** from the pop−up menu.

If you have added bookmarks for specific lines within a source file, you can also delete them while editing the file by right−clicking the bookmark icon in the editor area.

Related concepts

Bookmarks

Related tasks

Creating a bookmark for an entire file
Creating a bookmark within a file

Adding line items in the Tasks view

The Tasks view contains line items for system−generated problems, warnings, and errors. You can add your own entries to the table to build a list of to−do items, or tasks.

1. On the toolbar in the Tasks view, click the **New Task** button  . The **New Task** dialog will open.
2. Type a brief description for the task in the **Description** field. The new task is assigned default priority and completed values. These values may also be modified within the **New Task** dialog.
3. Press **OK**.

Related concepts

Tasks view
Bookmarks

Related tasks

Deleting tasks
Associating a task with a resource
Filtering the Tasks view
Creating a bookmark for an entire file
Creating a bookmark within a file

Deleting tasks

You can delete associated tasks from the gray border at the left of the editor area.

1. In the marker bar in the editor area, locate the task marker that you want to delete.
2. From the marker's pop−up menu, select **Remove Task**.
3. The task marker disappears and the task is removed from the Tasks view.

You can also delete one or more line items from the Tasks view by pressing the Delete key or by clicking the **Delete** button on the Tasks view toolbar, or from the pop−up menu.

Tip: To delete all completed tasks, right−click in the Tasks view and select **Delete Completed Tasks** from the pop−up menu.

Related concepts

Tasks view

Associating a task with a resource

You can associate tasks with an editable resource, for instance to remind yourself to update a line of source code later.

1. In one of the navigation views, double−click the resource with which you wish to associate the new task. The resource opens in the editor area.
2. Right−click in the gray border at the left of the editor area, beside the line of text or source code against which you want to log the new task.
3. On the pop−up menu, select **Add Task**.
4. When prompted, enter a brief description of the task.

A new task icon appears in the border of the editor area, to the left of the line where you added the task. When you move the mouse pointer over the marker, the description of the task is displayed as a tooltip. The task is also added to the Tasks view. You can delete a task either by right−clicking its icon in the editor area and selecting **Remove Task**, or by pressing the Delete key in the Tasks view.

Related concepts

Tasks view
Bookmarks

Related tasks

Adding line items in the Tasks view
Deleting tasks
Filtering the Tasks view
Creating a bookmark for an entire file
Creating a bookmark within a file

Filtering the Tasks and Problems views

You can filter the tasks or problems that are displayed in the Tasks or Problems views. For example, you might wish to see only problems that have been logged by the Workbench, or tasks that you have logged as reminders to yourself. You can filter items according to which resource or group of resources they are associated with, by text string within the Description field, by problem severity, by task priority, or by task status.

1. On the toolbar of the Tasks or Problems view, click **Filter**.  .
2. Select the radio buttons and checkboxes that correspond to your filtering objectives.
3. Click **OK**.

Related concepts

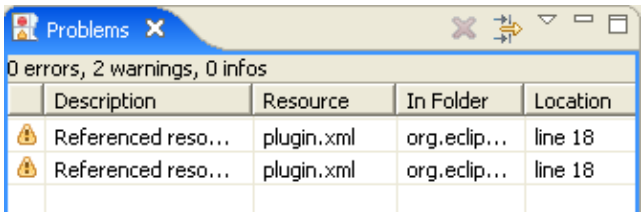
Tasks view

Related tasks

Adding line items in the Tasks view
Associating a task with a resource
Deleting tasks

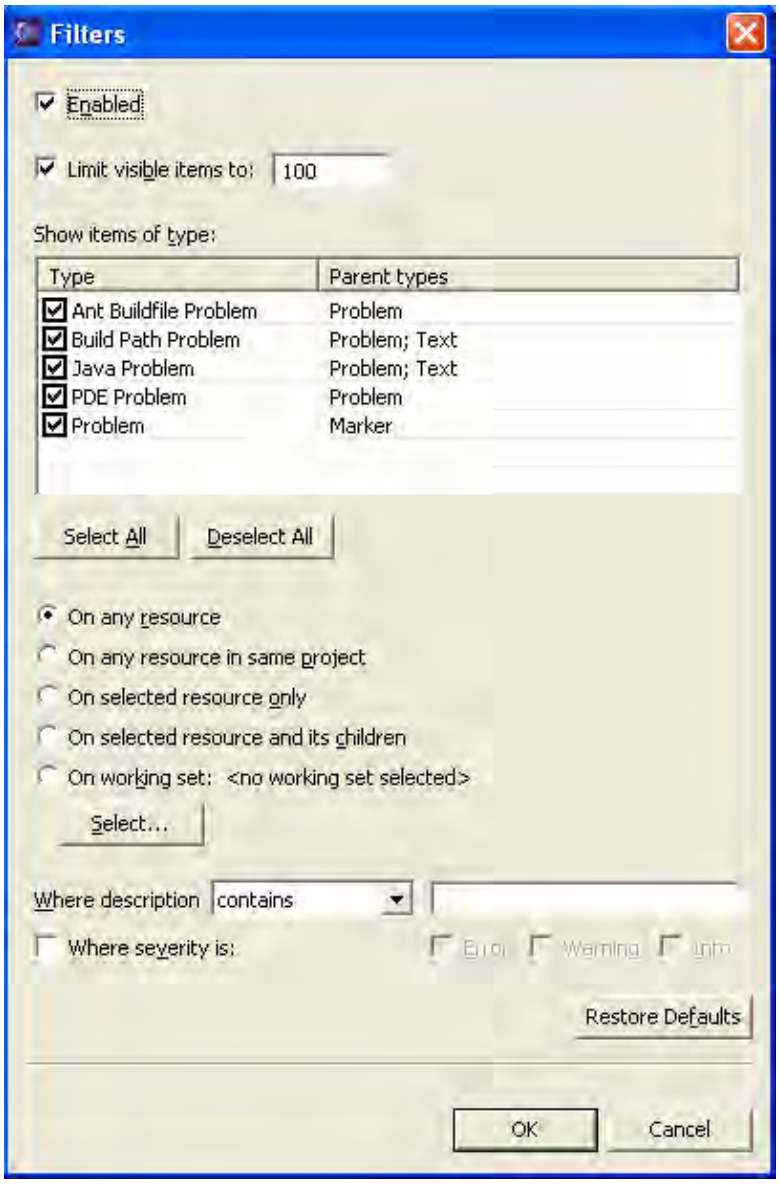
Problems view

As you work with resources in the workbench, various builders may automatically log problems, errors, or warnings in the Problems view. For example, when you save a Java source file that contains syntax errors, those will be logged in the Problems view. When you double−click the icon for a problem, error, or warning, the editor for the associated resource automatically opens to the relevant line of code.



The first column of the Problems view displays an icon that denotes the type of line item. Click on the item to open the file in an editor, the line containing the problem will be highlighted.

You can filter line items in the Problems view, to view only warnings and errors associated with a particular resource or group of resources.



To add the Problems view to the current perspective, click **Window > Show View > Other > Basic > Problems**.

Related concepts

Markers
Bookmarks
Tasks view

Related tasks

Opening views
Moving and docking views
Automatically fixing problems

Opening views

Perspectives offer pre−defined combinations of views and editors. To open a view that is not included in the current perspective, select **Window > Show View** from the main menu bar.

You can create *fast views* to provide quick access to views that you use often.

After adding a view to the current perspective, you may wish to save your new layout by clicking **Window > Save Perspective As**.

Related concepts

Views
Fast views
Perspectives

Related tasks

Moving and docking views
Creating fast views
Maximizing a view or editor
Resetting perspectives

Fast views

Fast views are hidden views that can be quickly opened and closed. They work like other views except they do not take up space in your Workbench window.

Fast views are represented by toolbar buttons on the *fast view bar*, which is the horizontal toolbar initially on the bottom left of the Workbench window. When you click the toolbar button for a fast view, that view opens temporarily in the current perspective. As soon as you click outside that view, it is hidden again. The fast view bar can also be docked on the right and left side of the Workbench window.

You can create a new fast view by dragging any open view to the shortcut bar or by selecting **Fast View** from the menu that opens when you click the icon at the left end of the view's title bar.

Related concepts

Workbench
Toolbars

Related tasks

Creating fast views
Moving and docking views
Maximizing a view or editor

Creating fast views

Fast views are hidden views that can be quickly opened and closed. They work like other views except they do not take up space in your Workbench window.

To create a fast view:

1. Click the title bar of the view that you want. Hold the mouse button down.
2. Drag the view to the shortcut bar and release the mouse button. By default the shortcut bar is located in the lower left corner of the window.

The icon for the view that you dragged now appears on the shortcut bar. You can look at the view by clicking its icon on the shortcut bar. As soon as you click somewhere else outside the view, it is hidden again.

To restore the view to its original location (to delete the fast view), toggle the fast view item in the view button's context menu.

You can also create and restore fast views by selecting **Fast View** from the context menu that opens when you click the icon at the left side of the view's title bar.

Related concepts

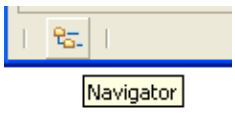
Views
Fast views
Perspectives

Related tasks

Working with fast views
Opening views
Moving and docking views
Maximizing a view or editor
Saving a user−defined perspective
Resetting perspectives

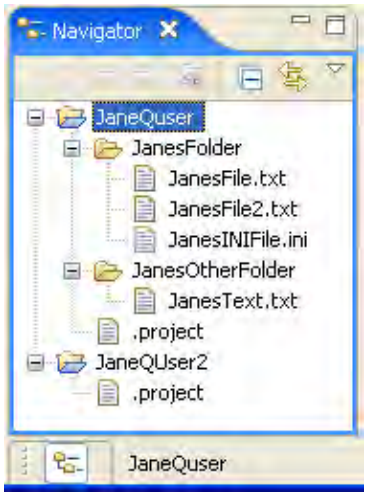
Working with fast views

If you have converted the Navigator to a fast view it will appear in the shortcut bar as shown below.



To work with a fast view proceed as follows.

1. In the shortcut bar click on the Navigator fast view button.
2. Observe the Navigator view slides out from the shortcut bar.



3. You can use the Navigator fast view as you would normally.
4. To hide the fast view simply click off of it or click on the Minimize button on the fast view's toolbar

Note: If you open a file from the Navigator fast view, the fast view will automatically hide itself to allow you to work with the file.

To convert a fast view that has been maximized back to a regular sized view, select ***Restore*** from the context menu of the icon in the top left corner of the view. To reposition a fast view, drag the fast view's title bar (or close the fast view and then drag its button from the shortcut bar) and drop it on the workbench like a normal view.

Related concepts

Views
Fast views
Perspectives

Related tasks

Creating fast views
Opening views
Docking views
Maximizing a view or editor
Saving a user defined perspective
Resetting perspectives

Moving and docking views

To change the location of a view in the current perspective:

1. Drag the view by its title bar. Do not release the left mouse button yet.
2. As you move the view around the Workbench, the mouse pointer changes to one of the *drop cursors* shown in the table below. The drop cursor indicates where the view will be docked if you release the left mouse button. To see the drop cursor change, drag the view over the left, right, top, or bottom border of another view or editor.
3. When the view is in the location that you want, relative to the view or editor area underneath the drop cursor, release the left mouse button.
4. (Optional) If you want to save your changes, select **Window > Save Perspective As** from the main menu bar.
5. Note that a group of stacked views can be dragged using the empty space to the right of the view tabs.

You can also move a view by using the pop−up menu for the view. (Left−click on the icon at the left end of the view's title bar, or right−click anywhere else in the view's title bar).

Drop cursor	**Where the view will be moved to**
	Dock above: The view is docked above the view underneath the cursor.
	Dock below: The view is docked below the view underneath the cursor.
	Dock to the right: The view is docked to the right of the view underneath the cursor.
	Dock to the left: The view is docked to the left of the view underneath the cursor.
	Stack: The view is docked as a Tab in the same pane as the view underneath the cursor.
	Restricted: You cannot dock the view in this area.

Related concepts

Views
Fast views
Perspectives

Related tasks

Opening views
Creating fast views
Maximizing a view or editor
Saving a user defined perspective
Resetting perspectives

Maximizing a view or editor

You can temporarily maximize a view or editor so that it fills the Workbench window.

1. Open the menu for the view or editor by clicking the icon at the far left of its title bar. (You can get the same effect by right−clicking anywhere in the title bar.)
2. Select **Maximize**.

To restore the view or editor to its previous position and size:

1. Open the menu again.
2. Select **Restore**.

Tip: You can also maximize or restore a view or editor by double−clicking its title bar or using the ***Maximize/Restore*** icon.

Related concepts

Views
Fast views
Perspectives

Related tasks

Opening views
Moving and docking views

Saving a user defined perspective

If you have modified a perspective by adding, deleting, or moving (docking) views, you can save your changes for future use.

1. Switch to the perspective that you want to save.
2. Click **Window > Save Perspective As**.
3. Type a new name for the perspective into the **Name** field.
4. Click **OK**.

The name of the new perspective is added to the **Window > Open Perspective** menu.

Related concepts

Perspectives
Views

Related tasks

Resetting perspectives
Deleting a user defined perspective
Opening views
Moving and docking views

Resetting perspectives

To restore a perspective to its original layout:

1. Click **Window > Preferences**.
2. Expand **General** and choose **Perspectives**.
3. From the **Available perspectives** list, select the perspective you want to restore.
4. Click **Reset**.
5. Click **OK**.

Related concepts

Perspectives

Related tasks

Saving a user defined perspective

Deleting a user defined perspective

You can delete perspectives that you defined yourself, but not those that are delivered with the Workbench.

1. On the main menu bar, click **Window > Preferences**. The Preferences window opens.
2. Expand the **General** category and click **Perspectives**.
3. From the **Available perspectives** list, select the one that you want to delete and click **Delete**.
4. Click **OK**.

Related concepts

Perspectives

Related tasks

Saving a user defined perspective
Resetting perspectives

Automatically fixing problems

Problems displayed in the Problems view may offer a **Quick Fix** menu option in the context menu. Selecting this menu option will present one or more possible fixes that can be automatically applied for you.

Certain Java problems may be resolved using the Quick Fix feature. For example, a missing import statement in a Java file will result in a Quick Fix suggestion to add the import statement or change the type name to a name that is already imported.

1. In the Problems view, right−click the task you want to Quick Fix.
2. Select **Quick Fix** from the pop−up menu.
3. Choose from one of the suggested fixes.

Related concepts

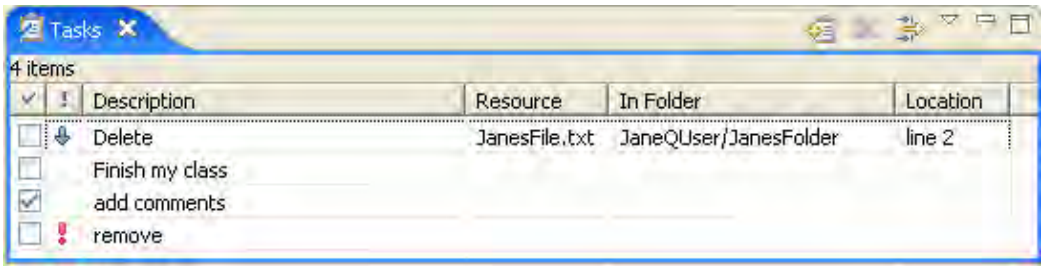
Markers
Tasks view

Related reference

Tasks view

Tasks view

The Tasks view displays tasks that you add manually. You can associate a task with a resource in the Workbench, but this is not required.



By default, the Tasks view is included in the Resources perspective. To add it to the current perspective, click **Window > Show View > Other > Basic > Tasks**.

The following icons are used by the Tasks view:

Icon	Description
	High priority task
	Low priority task
	Completed task
	Add task
	Delete
	Filter

The first column indicates whether the task is completed. Completed tasks are flagged with a check mark, which you add manually.

The second column indicates whether the task is high, normal, or low priority.

The Description column contains a description of the line item. You can edit the description of user−defined tasks by selecting **Properties** from the context menu.

The Resource and In Folder columns provide the name and location of the resource associated with each line item.

The Location column indicates the line number of the line item within its associated resource.

Toolbar

The toolbar of the Tasks view includes the following buttons.

Add task
: Manually add a "to do" item to the Tasks view.

Delete
: Delete the selected line item.

Filter
: Filter the view according to the type of item.

Menus

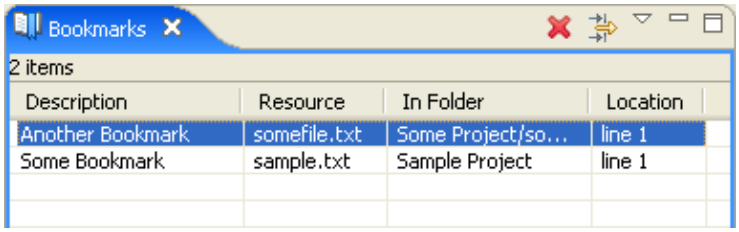
Click the icon at the left end of the view's title bar to open a menu of items generic to all views. Click the upside−down triangle icon to open a menu of items specific to the Tasks view. Right−click inside the view to open a context menu.

Related tasks

Adding line items in the Tasks view
Associating a task with a resource
Deleting tasks
Filtering the task view

Bookmarks view

The Bookmarks view displays user defined bookmarks.



To add the Bookmarks view to the current perspective, click **Window > Show View > Other > Basic > Bookmarks**.

The Description column contains a description of the bookmark. You can edit the description by selecting **Properties** from the context menu.

The Resource and In Folder columns provide the name and location of the resource associated with each bookmark.

The Location column indicates the line number of the bookmark within its associated resource.

Toolbar

The toolbar of the Bookmarks view includes the following buttons.

Delete
: Delete the selected bookmark.

Go to
: Open the bookmark's resource and navigate to the bookmarked region.

Menus

Click the icon at the left end of the view's title bar to open a menu of items generic to all views. Click the black upside−down triangle icon to open a menu of items specific to the Bookmarks view. Right−click inside the view to open a context menu.

Related tasks

Creating a bookmark for an entire file
Creating a bookmark within a file
Deleting a Bookmark

Opening perspectives

Perspectives provide combinations of views and editors that are suited to performing a particular set of tasks. For example, you would normally open the Debug perspective to debug a Java program.

To open a new perspective:

1. Click the **Open Perspective** button  on the shortcut bar on the left side of the Workbench window. (This provides the same function as the **Window > Open Perspective** menu on the menu bar.)
2. To see a complete list of perspectives, select **Other** from the drop−down menu.
3. Select the perspective that you want to open.

When the perspective opens, the title bar of the window it is in changes to display the name of the perspective. In addition, an icon is added to the shortcut bar, allowing you to quickly switch back to that perspective from other perspectives in the same window.

By default, a perspective will open in the same window. If you would rather it opened in a new window, change the setting in **Window > Preferences > General > Perspectives**.

Related concepts

Perspectives

Related tasks

Opening views
Changing where perspectives open
Specifying the default perspective
Switching between perspectives

Changing where perspectives open

You can change the default behavior for how perspectives are opened in the Workbench.

1. From the main menu bar, select **Window > Preferences**.
2. Expand the **General** category on the left and select **Perspectives**. The Perspectives preferences page opens.
3. Select either **In the same window** or **In a new window** from the **Open a new perspective** group.
4. Click **OK**.

Related concepts

Perspectives

Related tasks

Opening perspectives
Specifying the default perspective
Switching between perspectives
Configuring perspectives

Specifying the default perspective

The default perspective is indicated in the **Window > Open Perspective > Other** menu. The pre−defined default perspective is indicated by the word default in brackets following the perspective name, for example, ***Resource (default)***.

To change the default perspective:

1. From the main menu bar, select **Window > Preferences**.
2. Expand the **General** category on the left and select **Perspectives**. The Perspectives preferences page opens.
3. Select the perspective that you want to define as the default, and click **Make Default**. The default indicator moves to the perspective that you selected.
4. Click **OK**.

Related concepts

The Workbench
Perspectives

Related tasks

Opening perspectives
Changing where perspectives open
Configuring perspectives

Configuring perspectives

In addition to configuring the layout of your perspective you can also control several other key aspects of a perspective. These include:

- The options available on the **File > New** submenu.
- The options available on the **Window > Open Perspective** submenu.
- The options available on the **Window > Show View** submenu.
- Action sets (buttons and menu options) that show up on the toolbar and menu bar.

To configure a perspective:

1. Switch to the perspective that you want to configure.
2. Select **Window > Customize Perspective**.
3. Expand the item that you want to customize.
4. Use the checkboxes to select which elements you want to see on drop−down menus in the selected perspective. Items you do not select will still be accessible by clicking the **Other** menu option.
5. Click **OK**.

Related concepts

Perspectives

Related tasks

Changing where perspectives open
Specifying the default perspective
Saving a user defined perspective

Switching between perspectives

Open perspectives are represented by icons on the perspective bar. When you have more than one perspective open, you can switch between them by clicking the icons on the shortcut bar.

Related concepts

Perspectives

Related tasks

Opening perspectives
Changing where perspectives open

Narrowing the scope of the Navigator view

By default, one of the navigation views shows all resources in your Workbench. You can focus on a subset of resources by temporarily "going into" a project or folder and hiding all other resources.

1. In one of the navigation views, right−click the project of folder that you want to focus on.
2. From the pop−up menu, select **Go Into**.

The Navigator now shows only the contents of the selected project or folder. The title of the Navigator shows the name of the resource you are currently looking at. You can use the **Back**, **Forward**, and **Up** buttons on the toolbar of one of the navigation views to change the scope.

Related concepts

Navigator view

Related tasks

Showing or hiding files in the Navigator view

Showing or hiding files in the Navigator view

You can choose to hide system files or generated class files in one of the navigation views. (System files are those that have only a file extension but no file name, for example .classpath.)

1. On the toolbar for one of the navigation views, click the **Menu** button to open the drop−down menu of display options.
2. Select **Filter**.
3. In the dialog box that opens, select the checkboxes for the types of files that you want to hide.

In addition, you can restrict the displayed files to a working set.

1. On the toolbar for one of the navigation views, click the **Menu** button to open the drop−down menu of display options.
2. Choose **Select Working Set...**
3. Select an existing working set from the list or create a new one by selecting **New...**

Related concepts

Resources
Resource hierarchies
Navigator view
Working sets

Related tasks

Viewing resource properties
Finding a resource quickly
Narrowing the scope of the Navigator view
Sorting resources in the Navigator view

Resource hierarchies

Resources are stored and displayed in the Workbench in hierarchies. Described below are the terms used when referring to resources that are stored and displayed in a hierarchical structure.

Root
: The top level of the Workbench contents (in the file system).

Parent resource
: Any resource that contains another resource. Only projects and folders can be parent resources.

Child resource
: Any resource that is contained within another resource. Only files and folders can be child resources.

Resource hierarchies are displayed in the Navigator view, which is one of the default views in the Resource perspective.

Related concepts

Resources
Navigator view

Related tasks

Finding a resource quickly
Narrowing the scope of the Navigator view
Sorting resources in the Navigator view
Showing or hiding files in the Navigator view

Finding a resource quickly

To navigate to a particular resource in a view such as the Navigator view:

1. From **Navigate** menu, select **Go To > Resource**.
2. In the window that opens, start typing the name of the resource in the **Pattern** field. As you type the file name, the system offers a list of possible matches, based on what you have entered so far.
3. Select the resource that you want from the **Matching Resources** list, and click **OK**.

The view displays the selected resource.

If you want to open a particular resource in an editor rather than select it in a view, you can use the **Navigate > Open Resource** action. This action presents the same dialog as the **Go To > Resource** action, but immediately opens the matching resource for editing.

You can also do a contextual search for character strings contained within files in the Workbench. See the links to related tasks below.

Related concepts

Resources
Resource hierarchies
Navigator view

Related tasks

Searching for text within a file
Narrowing the scope of the Navigator view
Sorting resources in the Navigator view
Showing or hiding files in the Navigator view

Searching for text within a file

The **Go To** selection in the pop−up menu for one of the navigation views allows you to quickly find a resource in the Workbench by searching resource *names*.

You can also do contextual searches for information that is contained *inside* files in the Workbench. To find all files that contain a particular string of characters:

1. On the main toolbar, click the **Search** button  .
2. Type your search string in the **Containing Text** field, or use the pull−down list to select a previously entered search expression.
   Use `\` as an escape character for search strings that contain the special characters `*`, `?`, or `\`, (for example: `d:\\directory\\filename.ext`).
3. Finish entering your search options, (for example, to scope the search to specified file types), and click **Search**.
4. The Search view displays the results of your search. Right−click on any item in the Search view to open a pop−up menu that allows you to remove items from the list, copy search results to the clipboard, or rerun the search. To open one of the listed files, double−click it or select **Go to File** from its pop−up menu.

If you close the Search view, you can return to it later by selecting **Window > Show View > Other > Basic > Search**.

Related concepts

Search view

Related tasks

Searching for files

Related reference

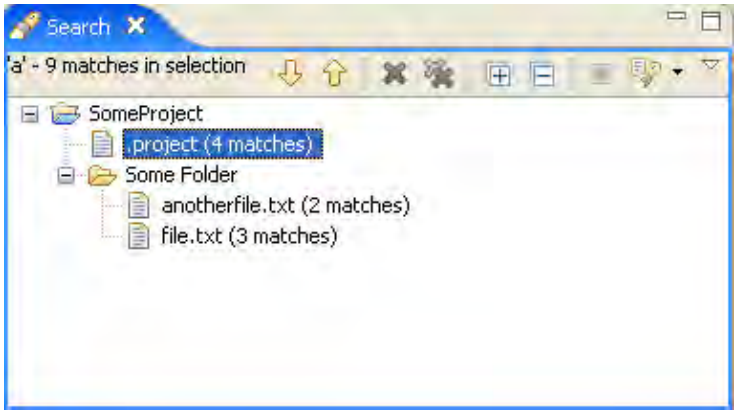
Search view
File search

Search view

This view displays the results of a search.

Text searches will only search for expressions in files with extensions (file types) specified in the search dialog.

Here is what the Search view looks like:



Toolbar

The toolbar in the Search view contains the following buttons:

Show Next Match
: This command highlights the next match of the search expression in the editor area, opening the file if required.

Show Previous Match
: This command highlights the previous match of the search expression in the editor area, opening the file if required.

Remove Selected Matches
: Removes all highlighted matches from the search results.

Search
: This command opens the Search dialog.

Previous Search Results
: This command allows you to browse previously conducted searches and repeat a previous search. You can select a previous search from the drop−down menu or clear the search history.

Related concepts

Resources
Views

Related tasks

Searching for text within a file
Searching for files

Related reference

Search view
File search

Searching for files

The **Go To > Resource** action in the **Navigate** menu allows you to quickly find a resource in the Workbench by searching resource *names*.

You can also do more complex searches for files in the Workbench. For example, to find all files that end with .xml:

1. On the main toolbar, click the **Search** button  .
2. Type `.xml` into the **File name patterns** field and leave the **Containing text** field empty. (You can use the pull−down list to select `.xml` if it had been previously entered.)
3. Finish entering your search options, for example to scope the search to specified working sets, and click **Search**.
4. The Search view displays the results of your search. Right−click on any item in the Search view to open a pop−up menu that allows you to remove items from the list, copy search results to the clipboard, or rerun the search. To open one of the listed files, double−click it or select **Go to File** from its pop−up menu.

If you close the Search view, you can return to it later by selecting **Window > Show View > Other > Basic > Search**.

Related concepts

Search view
Working sets

Related tasks

Searching for text within a file

Related reference

Search view
File search

Working sets

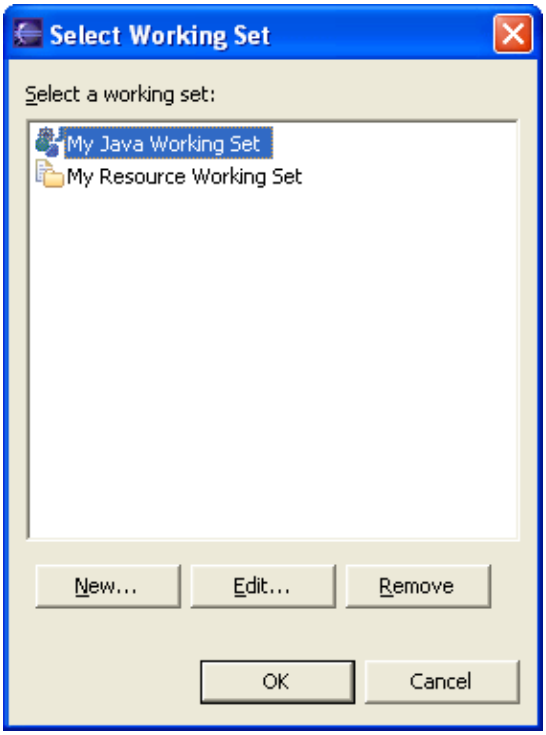
Working sets group elements for display in views or for operations on a set of elements.

The navigation views use working sets to restrict the set of resources that are displayed. If a working set is selected in the navigator, only resources, children of resources, and parents of resources contained in the working set are shown.

Working sets may be used in the task view to limit the display of tasks in a similar fashion as the navigation views.

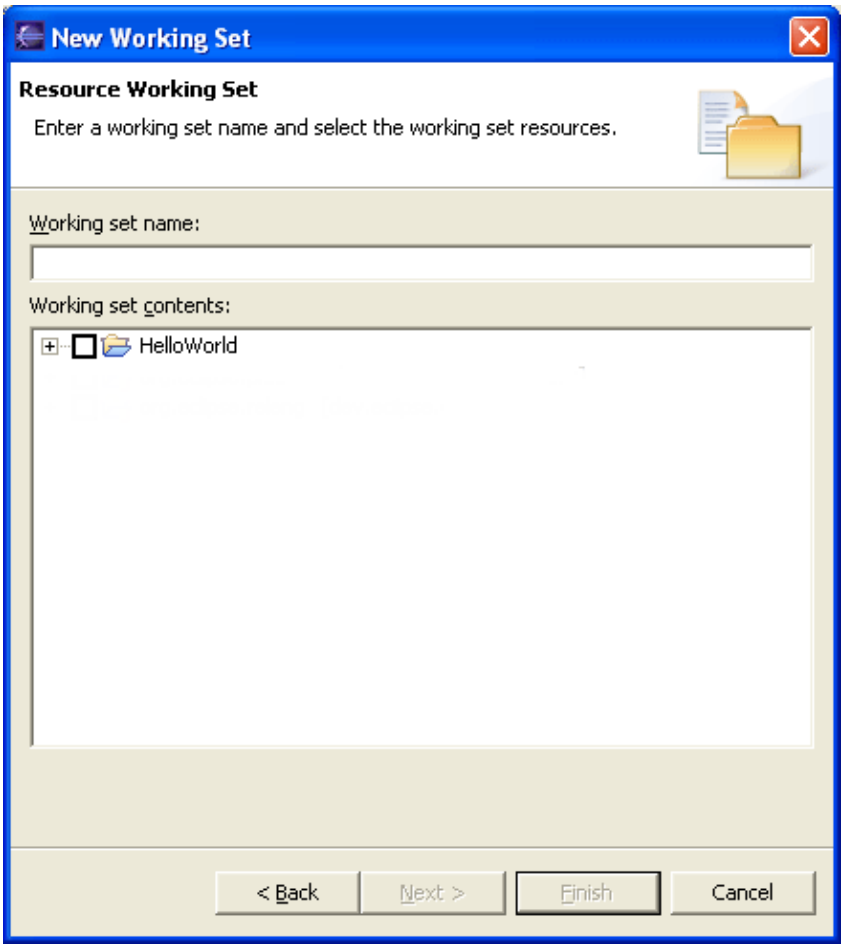
When using the Workbench search facility, working sets may be used to restrict the set of elements that are searched.

Different views provide different means to specify a working set. However, they typically use the following working set selection dialog to manage existing working sets and to create new working sets.



When you create a new working set you may be able to choose from different types of working sets. In the example below you can create a resource working set, a Java working set or a help working set.

If you create a new resource working set you will be able to select the working set resources as shown below. The same wizard is used to edit an existing working set. Different types of working sets provide different kinds of working set editing wizards.



Note: Newly created resources are not automatically included in the active working set. They are implicitly included in a working set if they are children of an existing working set element. If you want to include other resources after you have created them you have to explicitly add them to the working set.

Related concepts

Navigator view
Tasks view

Related tasks

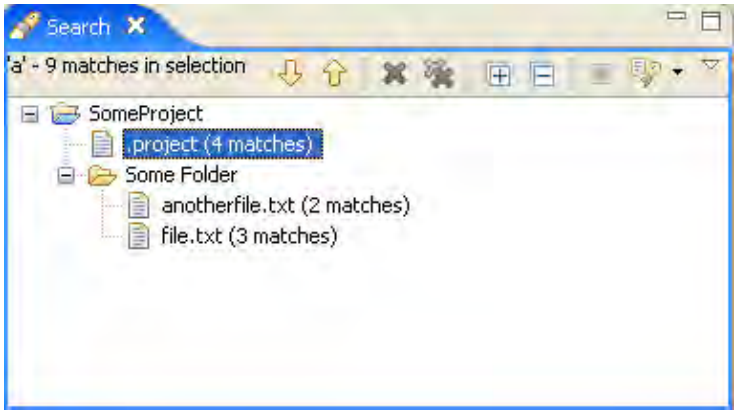
Searching for text within a file
Showing or hiding resources in the Navigator view

Search view

This view displays the results of a search.

Text searches will only search for expressions in files with extensions (file types) specified in the search dialog.

Here is what the Search view looks like:



Show Next Match
: This command highlights the next match of the search expression in the editor area, opening the file if required.

Show Previous Match
: This command highlights the previous match of the search expression in the editor area, opening the file if required.

Remove Selected Matches
: Removes all highlighted matches from the search results.

Remove All Matches
: Remove all matches from the search results.

Expand All
: Expand all matches in the hierarchical view.

Collapse All
: Collapse all matches in the hierarchical view.

Cancel Current Search
: Cancel a search that is running.

Previous Search Results
: This command allows you to browse previously conducted searches and repeat a previous search. You can select a previous search from the drop−down menu or clear the search history.

Related concepts

Search view

Related tasks

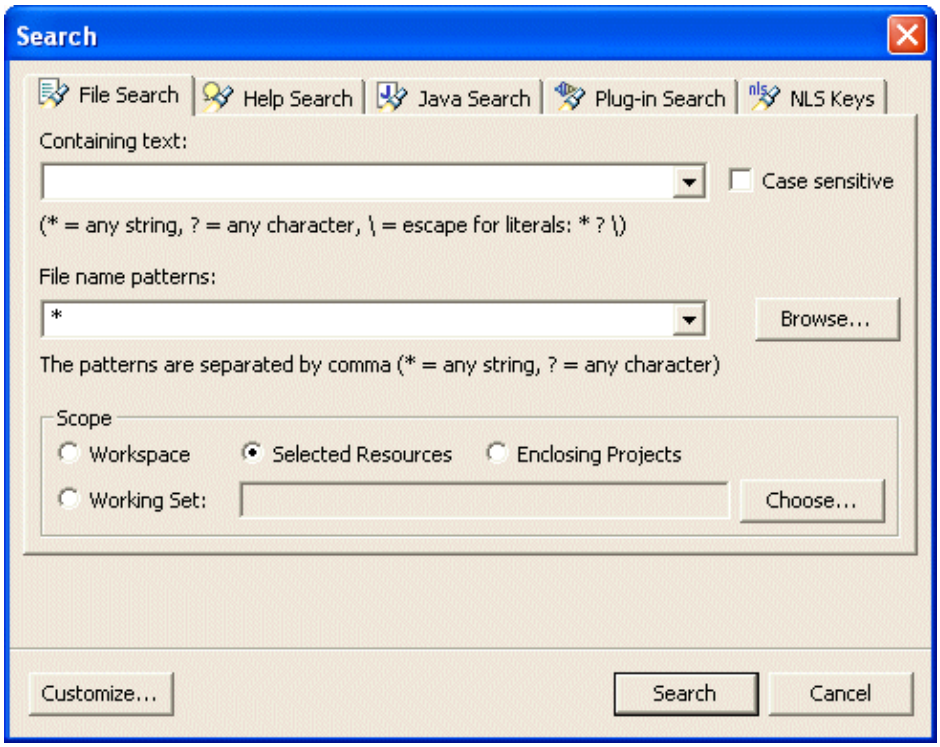
Searching for files
Searching for text within a file

Related reference

File search

File search

In the Search Dialog, the File Search tab allows you to search for files or text in the Workbench. You can bring up the Search Dialog by clicking on the Search toolbar button.



Containing text

Type the expression for which you wish to do the text search. Leave this field empty to search for files.

From the drop−down menu, you can choose to repeat or modify a recent search.

Wildcards

The available wildcards for search expressions are displayed in the search dialog:

- "*" matches any set of characters, including the empty string
- "?" matches for any character
- "\" is the escape for a literal; if you want to search for an asterisk, question mark, or backslash character, type a backslash before it to indicate that you are not using these characters as wildcards (e.g., "*", "\?", or "\\")

File name patterns

In this field, enter all the file name patterns for the files to find or search through for the specified expression.

Wildcards

The available wildcards for file name patterns are displayed in the search dialog:

- "*" matches any set of characters, including the empty string
- "?" matches for any character

Case sensitive

Turn this option on if you want the text search to be case sensitive.

Scope

Choose the scope of your search. You can either search the whole workspace, pre−defined working sets, previously selected resources or projects enclosing the selected resources.

Related concepts

Search view

Related tasks

Searching for files
Searching for text within a file

Related reference

Search view

Sorting resources in the Navigator view

To sort Workbench resources in one of the navigation views by name or by file type:

1. On the toolbar for one of the navigation views, click the **Menu** button to open the drop−down menu of display options.
2. Select **Sort**.
3. Select the desired sort option.

Related concepts

Resources
Resource hierarchies
Navigator view

Related tasks

Finding a resource quickly
Narrowing the scope of the Navigator view
Showing or hiding files in the Navigator view

Viewing resource properties

To display the various properties of a Workbench resource:

1. Right−click the resource in one of the navigation views.
2. Select **Properties** from the pop−up menu.

The kinds of properties that are displayed depend on the specific resource you have selected and the features and plug−in that are installed in the Workbench.

Tip: When the resource is selected in one of the navigation views you can also see basic properties for a resource by clicking **Window > Show View > Other > Basic > Properties**.

Related concepts

Resources

Related reference

Navigator view

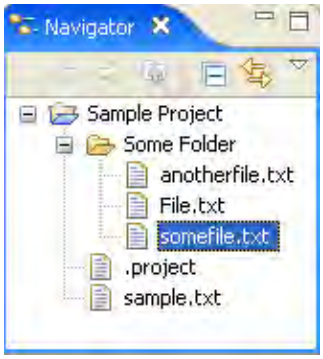
Related tasks

Showing or hiding files in the Navigator view

Navigator view

This view provides a hierarchical view of the resources in the Workbench.

Here is what the Navigator view looks like:



Toolbar

Back

This command displays the hierarchy that was displayed immediately prior to the current display. For example, if you Go Into a resource, then the Back command in the resulting display returns the view to the same hierarchy from which you activated the *Go Into* command. The hover help for this button tells you where it will take you. This command is similar to the Back button in a web browser.

Forward

This command displays the hierarchy that was displayed immediately after the current display. For example, if you've just selected the Back command, then selecting the Forward command in the resulting display returns the view to the same hierarchy from which you activated the Back command. The hover help for this button tells you where it will take you. This command is similar to the Forward button in a web browser.

Up

This command displays the hierarchy of the parent of the current highest level resource. The hover help for this button tells you where it will take you.

Collapse All

This command collapses the tree expansion state of all resources in the view.

Link with Editor

This command toggles whether the Navigator view selection is linked to the active editor. When this option is selected, changing the active editor will automatically update the Navigator selection to the resource being

edited.

Menus

Click the black upside−down triangle icon to open a menu of items specific to the Navigator view. Right−click inside the view to open a context menu.

Select Working Set

Opens the **Select Working Set** dialog to allow selecting a working set for the Navigator view.

Deselect Working Set

Deselects the current working set.

Edit Active Working Set

Opens the **Edit Working Set** dialog to allow changing the current working set.

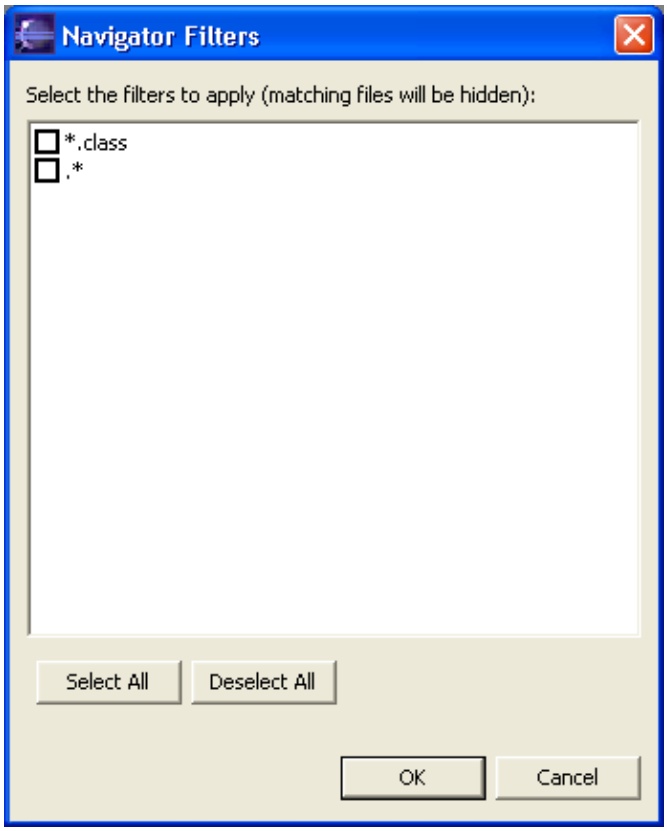
Sort

This command sorts the resources in the Navigator view according to the selected schema:

- **By Name**: Resources are sorted alphabetically, according to the full name of the resource (e.g., A.TXT, then B.DOC, then C.HTML, etc.)
- **By Type**: Resources are sorted alphabetically by file type/extension (e.g., all DOC files, then all HTML files, then all TXT files, etc.).

Filters

This command allows you to select filters to apply to the view so that you can show or hide various resources as needed. File types selected in the list will not be shown in the Navigator. Here is what the file filters dialog looks like:



Link with Editor

See the toolbar item description above.

In addition to these menu items, the Navigator view menu shows a list of recently used working sets that have been selected in the view.

Context menu

New

This command allows you to create a new resource in the Workbench. Select the type of resource to create from the submenu.

Go Into

This command displays a new hierarchy in the Navigator view, with the children of the selected resource as its contents. For example if you Go Into a project, the Navigator will be refocused on the immediate files and folders of the project.

Open

This command opens the selected resource. If the resource is a file that is associated with an editor, then the Workbench launches the associated internal, external, or ActiveX editor and opens the file in that editor.

Open With

This command allows you to open an editor other than the default editor for the selected resource. Specify the editor with which to open the resource by selecting an editor from the submenu.

Copy

This command copies the selected resource to the clipboard.

Paste

This command pastes resources on the clipboard into the selected project or folder. If a resource is selected the resources on the clipboard are pasted as siblings of the selected resource.

Delete

This command deletes the selected resource from the workspace.

Move

This command moves the selected resource to another location. A dialog will appear, prompting for the destination location to which the resource will be moved.

Rename

This command allows you to specify a new name for the selected resource.

Import

This command opens the import wizard and allows you to select resources to import into the Workbench.

Export

This command opens the export wizard and allows you to export resources to an external location.

Refresh

This command refreshes the Workbench's view of the selected resource and its children. For example, this is used when you create a new file for an existing project outside the Workbench and want the file to appear in the Navigator view.

Close Project

The close project command is visible when an open project is selected. This command closes the selected project.

Open Project

The open project command is visible when a closed project is selected. This command opens the selected project.

Team

Menu items in the Team submenu are related to version control management and are determined by the version control management system that is associated with the project. Eclipse provides the special menu item **Share Project...** for projects that are not under version control management. This command will present a wizard that allows the user to choose to share the project with any version control management systems that has been added to Eclipse. Eclipse ships with support for CVS.

Compare With

Commands on the Compare With submenu allow you to do one of the following types of compares:

- Compare two or three selected resources with each other
- Compare the selected resource with remote versions (if the project is associated with a version control management system).
- Compare the selected resource with a local history state

After you select the type of compare you want to do, you will either see a compare editor or a compare dialog. In the compare editor, you can browse and copy various changes between the compared resources. In the compare dialog, you can only browse through the changes.

Replace With

Commands on the Replace With submenu allow you to replace the selected resource with another state from the local history. If the project is under version control management, there may be additional items supplied by the version control management system as well.

Properties

This command displays the properties of the selected resource. The kinds of properties that are displayed depend on what type of resource is selected. Resource properties may include (but are not limited to):

- Path relative to the project in which it is held
- Type of resource
- Absolute file system path, or name of path variable when using linked resources
- Resolved path variable when using a path variable for a linked resource
- Size of resource
- Last modified date
- Read−only status

- Derived resource status
- Execution arguments, if it is an executable resource
- Program launchers, if it is launchable
- Project dependencies, if any

Related concepts

Team programming with CVS
Three−way compare
Linked resources

Related reference

Compare editor

Creating linked resources

Folders and files that are direct children of projects can be linked to locations in the file system outside of the project's location. These special folders and files are called linked resources.

To create a linked folder:

1. In one of the navigation views, right−click the project where you want to create the new folder.
2. From the pop−up menu, select **New > Folder**.
3. Specify the name of the folder as it will appear in the workbench. This name can be different from the name of the folder in the file system.
4. Click **Advanced**.
5. Check **Link to folder in the file system**.
6. Enter a file system path, or click **Browse** to select a folder in the file system.
7. Click **Finish**.

To create a linked file, follow the same steps as above, except choose **New > File** instead of **New > Folder** in the context menu.

Linked resource locations can also be specified relative to a variable. This makes it easier to share projects containing linked resources with other team members, since it avoids hard−coded absolute file system paths that may vary from one machine to the next.

To define a linked resource relative to a path variable, do the following after step 5 above:

6. Click the **Variables** button.
7. In the resulting dialog, select an existing path variable or create a new one.
8. If the chosen variable defines the exact path of the linked resource, click **OK**. Otherwise, click **Extend** to specify a file or folder below the location described by the path variable, then click **OK**.
9. Click **Finish**.

Tip: The **Window > Preferences > General > Workspace > Linked Resources** preference page also allows you to define path variables.

Note that, once you create a linked resource you will not be able to change the link target path that you entered in step 6. or 8. above.

Related concepts

Resources
Resource hierarchies
Linked resources

Related tasks

Creating a project
Creating a file
Creating a folder

Related reference

Linked resources
New Folder wizard
New File wizard

Creating a project

To create a project:

1. On the main menu bar, click **File > New Project**. The New Project wizard opens.
2. Select a category from the left column and then select the type of project to create from the right column. Click **Next**.
3. In the **Project name** field, type a name for your new project.
4. (Optional) The project that you create will map to a directory structure in the file system. The default file system location is displayed in the Location field. If you want to create the project and its contained resources in a different location, clear the **Use default location** checkbox and specify the new location.
5. If you want the new project to be dependent on one or more other projects, click **Next** and select the projects to be referenced.
6. Click **Finish**. The new project is listed in one of the navigation views.

Tip: The **General > Perspectives >** preference page allows you to specify the perspective behavior when a new project is created.

Related concepts

Resources
Resource hierarchies

Related tasks

Creating a folder
Creating a file
Copying resources
Moving resources

Creating a folder

To create a new folder:

1. In one of the navigation views, right−click the project or folder where you want to create the new folder.
2. From the pop−up menu, select **New > Folder**.
3. Enter the name of the new folder and click **Finish**.

Related concepts

Resources
Resource hierarchies

Related tasks

Creating a project
Creating a file
Creating linked resources
Copying resources
Moving resources

Creating a file

To create a file:

1. In one of the navigation views, right−click the project or folder where you want to create the new file.
2. From the pop−up menu, select **New > File**.
3. Specify the name of the file, including the file extension (for example, newfile.txt).
4. Click **Finish**.

The file opens in the editor associated with its type.

Related concepts

Resources
Resource hierarchies

Related tasks

Creating a project
Creating a folder
Creating linked resources
Copying resources
Moving resources

Copying resources

You can copy resources from one Workbench location to another (for example, from one project to another project).

1. In one of the navigation views, select the resources that you want to copy.
2. From the pop−up menu, select **Copy**.
3. In one of the navigation views, select the project or folder where you want to copy the resources to.
4. From the pop−up menu, select **Paste**.

Tip: You can also copy resources by holding down the Ctrl key and dragging them from the original location to the new location in one of the navigation views.

Copying a linked resource always copies the link and not the resource that it is linked to. E.g., when copying a linked folder the folder contents is not copied on the file system. Instead, a new linked folder is created linking to the same location on the file system. Keep in mind that linked resources can only be copied to a project and not to a folder.

Related concepts

Resources
Navigator view
Linked resources

Related tasks

Creating a project
Creating a folder
Creating a file
Moving resources
Renaming resources
Deleting resources

Moving resources

You can move resources from one Workbench location to another (for example, from one project to another project).

1. In one of the navigation views, select the resources that you want to move.
2. From the pop−up menu, select **Move**.
3. In the Folder Selection window, select the project or folder where you want to move the resources to and click **OK**.

Tip: You can also move resources by dragging them from the original location to the new location in one of the navigation views.

Moving a linked resource always moves the link and not the resource that it is linked to. E.g., when moving a linked folder the folder contents is not moved on the file system. Keep in mind that linked resources can only be moved to a project and not to a folder.

Related concepts

Resources
Navigator view
Linked resources

Related tasks

Finding a resource quickly
Copying resources
Renaming resources
Deleting resources

Renaming resources

You can rename Workbench resources using the Rename command on the context menu in one of the navigation views.

1. In one of the navigation views, right−click the resource that you want to rename.
2. From the pop−up menu, select **Rename**.
3. Type the new name for the resource.
4. Hit the return key.

Related concepts

Resources
Navigator view

Related tasks

Viewing resource properties
Moving resources

Deleting resources

To delete a resource from the Workbench:

1. In one of the navigation views, select the resources that you want to delete. (Hold down the Ctrl key to select more than one resource.)
2. Press the Delete key.
3. Click **Yes** in the dialog which appears.

Tip: You can achieve the same result by selecting **Delete** from the pop−up menu.

Deleting resources from the workspace will also delete the corresponding files and/or folders from the local file system. The only exception is linked resources, which are not removed from the file system when you delete the link in the workspace. However, deleting child resources of linked folders *will* delete those resources from the local file system.

In most cases it is possible to restore deleted files from the local history. See the related tasks for more details.

Related concepts

Resources
Navigator view
Linked resources

Related tasks

Deleting projects
Creating a project
Creating a folder
Creating a file
Creating linked resources
Restoring deleted resources from local history

Deleting projects

To delete a project and remove its contents from the file system:

1. Select the project in one of the navigation views.
2. Click **Delete** on the pop−up menu.
3. In the dialog which opens select **Also delete contents under ...**.
4. Click **Yes**.

To delete a project from the workspace without removing its contents from the file system:

1. Select the project in one of the navigation views.
2. Click **Delete** on the pop−up menu.
3. In the dialog which opens select **Do not delete contents**.
4. Click **Yes**.

Related concepts

Resources

Related tasks

Creating a project
Closing projects

Closing projects

When a project is closed, it can no longer be changed in the Workbench and its resources no longer appear in the Workbench, but they do still reside on the local file system. Closed projects require less memory. Also, since they are not examined during builds, closing a project can improve build time.

To close a project:

1. Select the project in one of the navigation views.
2. Click **Close Project** on the pop−up menu.

To re−open the project:

1. Select the project in one of the navigation views.
2. Click **Open Project** on the pop−up menu.

Related concepts

Resources
Builds

Related tasks

Creating a project
Deleting projects

Builds

A *build* is a process that derives new resources from existing ones, updates existing resources, or both.

In the Workbench, different *builders* are invoked for different types of projects. For example, when a build is triggered for a Java project, a Java builder converts each Java source file (.java files) into one or more executable class files (.class files). Builders usually enforce the constraints of some domain. For example, a Web link builder could update links to files whose name/location changes.

There are two kinds of builds:

- An *incremental build* leverages a previously built state and applies the transforms of the configured builders to the resources that have changed since the previous state was computed (that is, since the last build).
- A clean build discards any problems and previously built state. The next build after a clean will transform all resources according to the domain rules of the configured builders.

Incremental and clean builds can be done over a specific set of projects or the workspace as a whole. Specific files and folders cannot be built. There are two ways that builds can be performed:

- Automatic builds are performed as resources are saved. Automatic builds are always incremental and always operate over the entire workspace. You can configure your preferences (**Window > Preferences > General > Workspace**) to perform builds automatically on resource modification.
- Manual builds are initiated when you explicitly select a menu item or press the equivalent shortcut key. Manual builds can be either clean or incremental and can operate over collections of projects or the entire workspace.

Related concepts

External tools

Related tasks

Building resources
Performing builds manually
Performing builds automatically
Saving resources automatically before a manual build
Changing build order

External tools

External tools allow you to configure and run programs, batch files, Ant buildfiles, and others using the Workbench. You can save these external tool configurations and run them at a later time.

Output from external tools is displayed in the console view.

You can add external tools as part of the build process for a project. These external tools will run in the specified order every time a project is built.

The following variables are available when you configure an external tool. These variables are automatically expanded each time the external tool is run.

${workspace_loc} − The absolute path on the system's hard drive to Eclipse's workspace directory.

${workspace_loc:<*resource path*>} − The absolute path on the system's hard drive to the specified resource. The <*resource path*> is the full path of the resource relative to the workspace root. For example ${workspace_loc:/MyProject/MyFile.txt}. Note that the expanded result of this variable is not the same as ${workspace_loc}/MyProject/MyFile.txt if the project's contents directory for MyProject is outside the workspace directory.

${project_loc} − The absolute path on the system's hard drive to the currently selected resource's project or to the project being built if the external tool is run as part of a build.

${project_loc:<*resource path*>} − The absolute path on the system's hard drive to the specified resource's project. The <*resource path*> is the full path of the resource relative to the workspace root. For example ${workspace_loc:/MyProject/MyFile.txt}. Note that the expanded result of this variable is not the same as ${workspace_loc}/MyProject if the project's contents directory for MyProject is outside the workspace directory.

${container_loc} − The absolute path on the system's hard drive to the currently selected resource's parent (either a folder or project).

**${container_loc:<*resource path*>}** − The absolute path on the system's hard drive to the specified resource's parent (either a folder or project). The <*resource path*> is the full path of the resource relative to the workspace root. For example:${workspace_loc:/MyProject/MyFolder/MyFile.txt}. Note that the expanded result of this variable is not the same as ${workspace_loc}/MyProject/MyFolder if the project's contents directory for MyProject is outside the workspace directory.

${resource_loc} − The absolute path on the system's hard drive to the currently selected resource.

${resource_loc:<*resource path*>} − The absolute path on the system's hard drive to the specified resource. The <*resource path*> is the full path of the resource relative to the workspace root. For example ${workspace_loc:/MyProject/MyFile.txt}. Note that the expanded result of this variable is not the same as ${workspace_loc}/MyProject/MyFile.txt if the project's contents directory for MyProject is outside the workspace directory.

${project_path} − The full path, relative to the workspace root, of the currently selected resource's project or of the project being built if the external tool is run as part of a build.

${container_path} − The full path, relative to the workspace root, of the currently selected resource's parent (either a folder or project).

${resource_path} − The full path, relative to the workspace root, of the currently selected resource.

${project_name} − The name of the currently selected resource's project or of the project being built if the external tool is run as part of a build.

${container_name} − The name of the currently selected resource's parent (either a folder or project).

${resource_name} − The name of the currently selected resource.

${build_type} − The kind of build when the external tool is run as part of a build. The value can be one of "full", "incremental", or "auto". If the external tool is run outside of a build, the value is then "none".

Lets assume your Eclipse workspace directory is c:\eclipse\workspace and you have two projects, MyProject1 and MyProject2. The first project, MyProject1, is located inside the workspace directory, the second project, MyProject2, is located outside the workspace directory at c:\projects\MyProject2. Lets look at how the variable examples below will be expanded when an external tool is run, if the resource /MyProject2/MyFolder/MyFile.txt is selected.

Variable Examples	Expanded Results
${workspace_loc}	c:\eclipse\workspace
${workspace_loc:/MyProject1/MyFile.txt}	c:\eclipse\workspace\MyProject\MyFile.txt
${workspace_loc:/MyProject2/MyFile.txt}	c:\projects\MyProject2\MyFile.txt
${project_loc}	c:\projects\MyProject2
${project_loc:/MyProject1/MyFile.txt}	c:\eclipse\workspace\MyProject
${container_loc}	c:\projects\MyProject2\MyFolder
${resource_loc}	c:\projects\MyProject2\MyFile.txt
${project_path}	/MyProject2
${container_path}	/MyProject2/MyFolder
${resource_path}	/MyProject2/MyFolder/MyFile.txt
${project_name}	MyProject2
${container_name}	MyFolder
${resource_name}	MyFile.txt
${build_type}	none

Related concepts

Ant support
Builds

Related reference

External Tools preferences
External Tools and Ant icons

Related tasks

Running external tools
Running Ant buildfiles

Ant support

Apache Ant is an open source, Java−based build tool. See http://ant.apache.org for more details.

The Ant support allows you to create and run Ant buildfiles from the Workbench. These Ant buildfiles can operate on resources in the file system as well as resources in the workspace.

Output from an Ant buildfile is displayed in the console view in the same hierarchical format seen when running Ant from the command line. Ant tasks (for example "[mkdir]") are hyperlinked to the associated Ant buildfile, and javac error reports are hyperlinked to the associated Java source file and line number.

You can add classes to the Ant classpath and add Ant tasks and types from the ant runtime preference page, under **Window > Preferences > Ant > Runtime**.

Related concepts

Builds
External tools

Related tasks

Running Ant buildfiles
Modifying the Ant classpath
Adding new Ant tasks and types
Using a different version of Ant
antRunner application entry point

Related reference

Ant preferences
Ant editor preferences
Ant runtime preferences
Ant view
Ant editor
External Tools and Ant icons

Running Ant buildfiles

To run an Ant buildfile in the Workbench:

1. In one of the navigation views, select an XML file.
2. From the file's pop−up menu, select **Run Ant...**. The launch configuration dialog opens.
3. Select one or more targets from the **Targets** tab. The order in which you select the items is the order in which they will run. The order is displayed in the **Target execution order** box at the bottom of the tab. You can change the order of the targets by clicking the **Order...** button.
4. (Optional) Configure options on the other tabs. For example, on the **Main** tab, type any required arguments in the **Arguments** field.
5. Click **Run**.

The Ant buildfile will run on the selected targets. Unless you disabled the **Capture output** option on the **Main** tab, the console displays any applicable execution results as the buildfile runs.

These steps create a persisted launch configuration. The newly created configuration will appear in the launch history under **Run > External Tools** and will be available in the launch configuration dialog which is opened by clicking **Run > External Tools > External Tools...**.

Related concepts

Ant support
Builds
External tools

Related tasks

Running external tools
Modifying the Ant classpath
Using a different version of Ant
Adding new Ant tasks and types

Running external tools

The Workbench provides a mechanism for running tools that are not part of it. To configure an external tool:

1. Click **Run > External Tools > External Tools...**.
2. Select the Program configuration
3. Click the **New** button.
4. Enter a name for your external tool (for example, My External Tool).
5. Click the **Browse File System** button.
6. Find the tool you want to run (for example, on Windows it is usually a file with the extension .exe or .bat).
7. (Optional) In the **Arguments** field enter the necessary arguments for the tool.
8. (Optional)In the **Working directory** field enter the working directory for the tool.
9. Click **Run**.

These steps create a persisted launch configuration. The newly created configuration will appear in the launch history under **Run > External Tools** and will be available in the launch configuration dialog which is opened by clicking **Run > External Tools > External Tools...**.

It is also possible to set up and run an external tool to build a project.

1. Select the desired project.
2. From its pop−up menu choose **Properties**.
3. Click **Builders** from the list, and configure the tool as described above.

Related concepts

External tools

Related tasks

Running Ant buildfiles

Running Ant buildfiles

To run an Ant buildfile in the Workbench:

1. In one of the navigation views, select an XML file.
2. From the file's pop−up menu, select **Run Ant...**. The launch configuration dialog opens.
3. Select one or more targets from the **Targets** tab. The order in which you select the items is the order in which they will run. The order is displayed in the **Target execution order** box at the bottom of the tab. You can change the order of the targets by clicking the **Order...** button.
4. (Optional) Configure options on the other tabs. For example, on the **Main** tab, type any required arguments in the **Arguments** field.
5. Click **Run**.

The Ant buildfile will run on the selected targets. Unless you disabled the **Capture output** option on the **Main** tab, the console displays any applicable execution results as the buildfile runs.

These steps create a persisted launch configuration. The newly created configuration will appear in the launch history under **Run > External Tools** and will be available in the launch configuration dialog which is opened by clicking **Run > External Tools > External Tools...**.

Ant support
Builds
External tools

Running external tools
Modifying the Ant classpath
Using a different version of Ant
Adding new Ant tasks and types

Modifying the Ant classpath

When using an optional or custom task it is usually necessary to add extra libraries to the classpath. The Ant classpath can be modified globally or per launch configuration.

To modify the Ant classpath globally:

1. Click **Window > Preferences**.
2. Expand **Ant** and select **Runtime**. If you are not on it already, click the **Classpath** tab.
3. To add a JAR file to the classpath, click **Add Jar** and select the JAR file.
4. To add a folder to the classpath, click **Add Folder** and select the folder.
5. To remove an item from the classpath, select it and click **Remove**.
6. To restore the classpath to the default, click **Restore Defaults**.

To modify the Ant classpath for a launch configuration:

1. Click **Run > External Tools > External Tools...**
2. Select the Ant configuration whose classpath you wish to modify
3. Select the **Classpath** tab
4. To add a JAR file to the classpath, click **Add Jar** and select the JAR file.
5. To add a folder to the classpath, click **Add Folder** and select the folder.
6. To remove an item from the classpath, select it and click **Remove**.

Related concepts

Ant support
External tools

Related tasks

Running Ant buildfiles
Running external tools

Using a different version of Ant

The Eclipse platform provides Ant 1.6.1 as a plug−in library. When running an Ant buildfile in the Workbench, version 1.6.1 is used by default. It is possible to use different versions, although they are not supported. There are at least two ways of using a different version of Ant:

Changing the Ant runtime classpath:

When Ant runs a buildfile, it looks for the necessary classes on the Ant classpath. The Ant classpath consists of the plug−ins contributing new tasks, types or libraries, plus the classpath defined in the Ant runtime classpath preferences. To access the preferences, click **Window > Preferences > Ant > Runtime**. The JARs related to Ant 1.6.1 are grouped under the **Ant Home Entries** item. To change the Ant Home entries, click on the **Ant Home...** button and choose the Ant installation you wish to use.

After you change the Ant classpath, all future Ant builds will use the updated version instead of the default. To restore the Ant classpath to its original state, **Restore Defaults** button on the preference page.

Using Ant as an external tool:

When changing the Ant classpath is not an option, or if you just want to test a newer or beta version of Ant, using it as an external tool is an option. Usually when it is running in the Workbench, the Ant buildfile itself is considered to be a external tool, but this is not the only way. To install a binary distribution of Ant as an external tool (**Note:** These steps are for Windows, but similar methods can be used for other operating systems):

a. Download and install the binary version of Ant from http://ant.apache.org .
b. Click **Run > External Tools > External Tools...**.
c. Click **Program**
d. Click **New**.
e. Enter a name for your external tool (for example, External Ant).
f. For the **Location** field, click **Browse File System**.
g. Find and select a file called `ant.bat` (it should be in the `bin` folder of your Ant installation).
h. In the **Arguments** field enter the arguments for your buildfile that would normally enter for running the buildfile outside of the Workbench.
i. In the **Working Directory** field enter the directory of your buildfile.
j. Click **Run** to execute the buildfile.

When you run Ant as an external tool, none of the tasks or types contributed by Eclipse will work. Also, the Ant classpath preference has no effect in the buildfile execution.

Related concepts

Ant Support
External tools

Related tasks

Running Ant buildfiles
Running external tools
Modifying the Ant classpath
Adding new Ant tasks and types

Adding new Ant tasks and types

Add new Ant tasks and types through the Ant preferences page. These tasks and types will be available for buildfiles running in the Workbench without, having to use taskdef or typedef in the script declaration (For more on taskdef or typedef see the Ant documentation in http://ant.apache.org).

To add a new task or type:

1. Click **Window > Preferences**
2. Expand **Ant** and select **Runtime**.
3. Click the **Tasks** tab or the **Types** tab.
4. Click **Add Task** or **Add Type**.
5. Provide a name and class for the task or type.
6. Select the library where the task or type is declared. If the library is not present on the list, you must add it to the Ant classpath (see the Related task link below).

Related concepts

Ant Support
Builds
External tools

Related tasks

Running external tools
Modifying the Ant classpath
Using a different version of Ant

antRunner application entry point

The Ant core plug−in declares a main entry point that allows the running of Eclipse's integrated Ant support in a headless environment. An Ant build can then be run directly from the main platform launcher.

This is an example .bat file that shows how to make use of the antRunner application entry point on Windows.

```
echo off
setlocal

REM *****************************************************

REM The JRE java.exe to be used
set JAVAEXE="C:\jdk1.4.2\jre\bin\java.exe"

REM The Eclipse startup.jar
set STARTUPJAR="C:\eclipse\startup.jar"

REM The location of your workspace
set WORKSPACE=C:\runtime−workspace

REM The buildfile to use for the build
set BUILDFILE=build.xml

REM *****************************************************

if not exist %JAVAEXE% echo ERROR: incorrect java.exe=%JAVAEXE%, edit this file and correct the J
if not exist %JAVAEXE% goto done

if not exist %STARTUPJAR% echo ERROR: incorrect startup.jar=%STARTUPJAR%, edit this file and corr
if not exist %STARTUPJAR% goto done

if not exist %WORKSPACE% echo ERROR: incorrect workspace=%WORKSPACE%, edit this file and correct
if not exist %WORKSPACE% goto done

if not exist %BUILDFILE% echo ERROR: incorrect buildfile=%BUILDFILE%, edit this file and correct
if not exist %BUILDFILE% goto done

:run
@echo on
%JAVAEXE% −cp %STARTUPJAR% org.eclipse.core.launcher.Main −noupdate −application
org.eclipse.ant.core.antRunner −data %WORKSPACE% −buildfile %BUILDFILE%

:done
pause
```

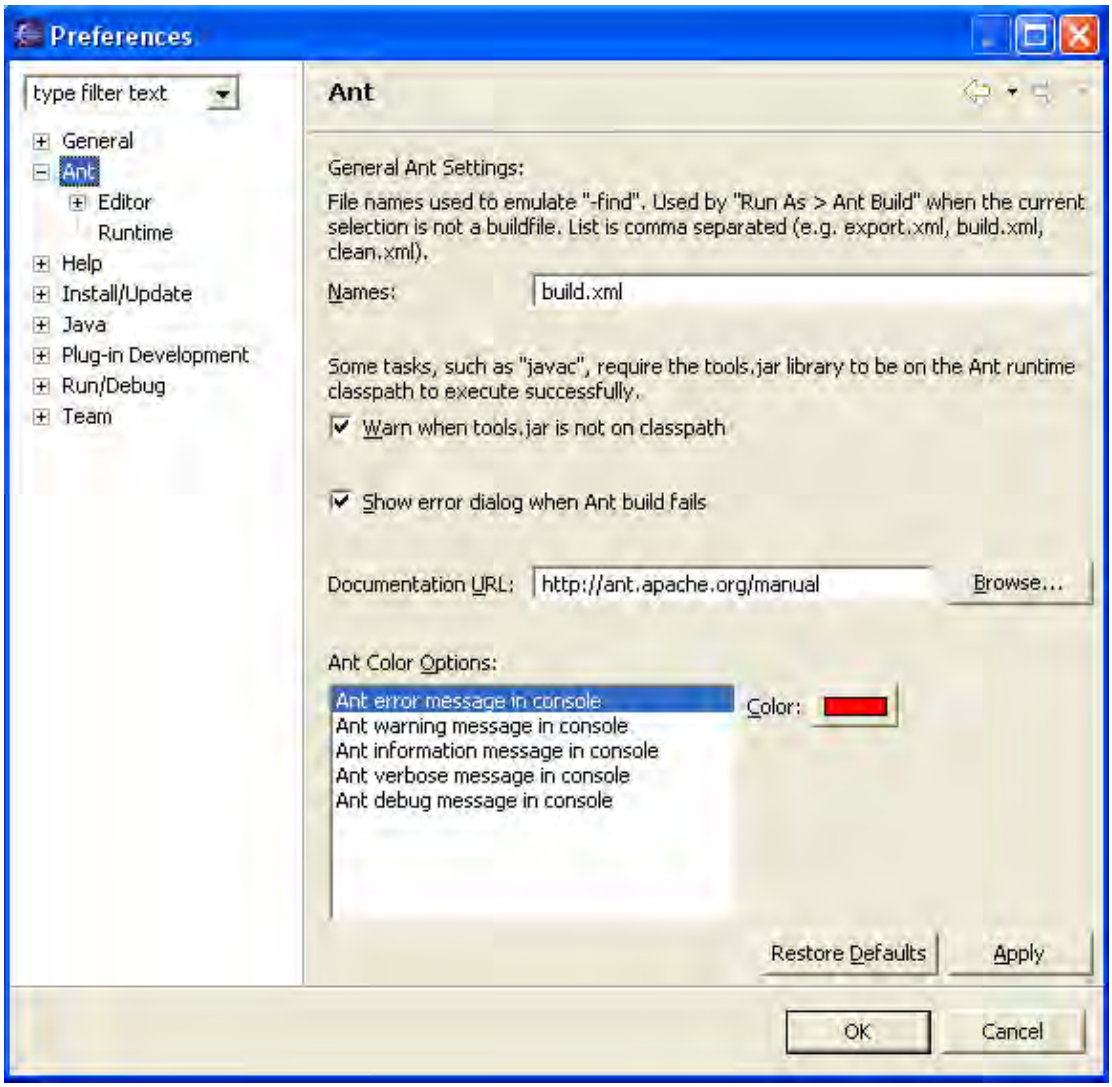
Related concepts

Ant support

Ant

The following preferences can be changed on the Ant page.

Option	Description	Default
Names	This option allows you to configure the buildfiles that Ant's "−find" emulation will search for.	build.xml
Warn when tools.jar is not on the classpath	Displays warning when tools.jar is not on the classpath.	On
Show error dialog when Ant build fails	Displays error message when the Ant build fails.	On
Documentation URL	This option allows you to browse or the applicable documentation URL	http://ant.apache.org/manual
Ant Color Options	You can also configure the color of the Ant build output.	Red



Related concepts

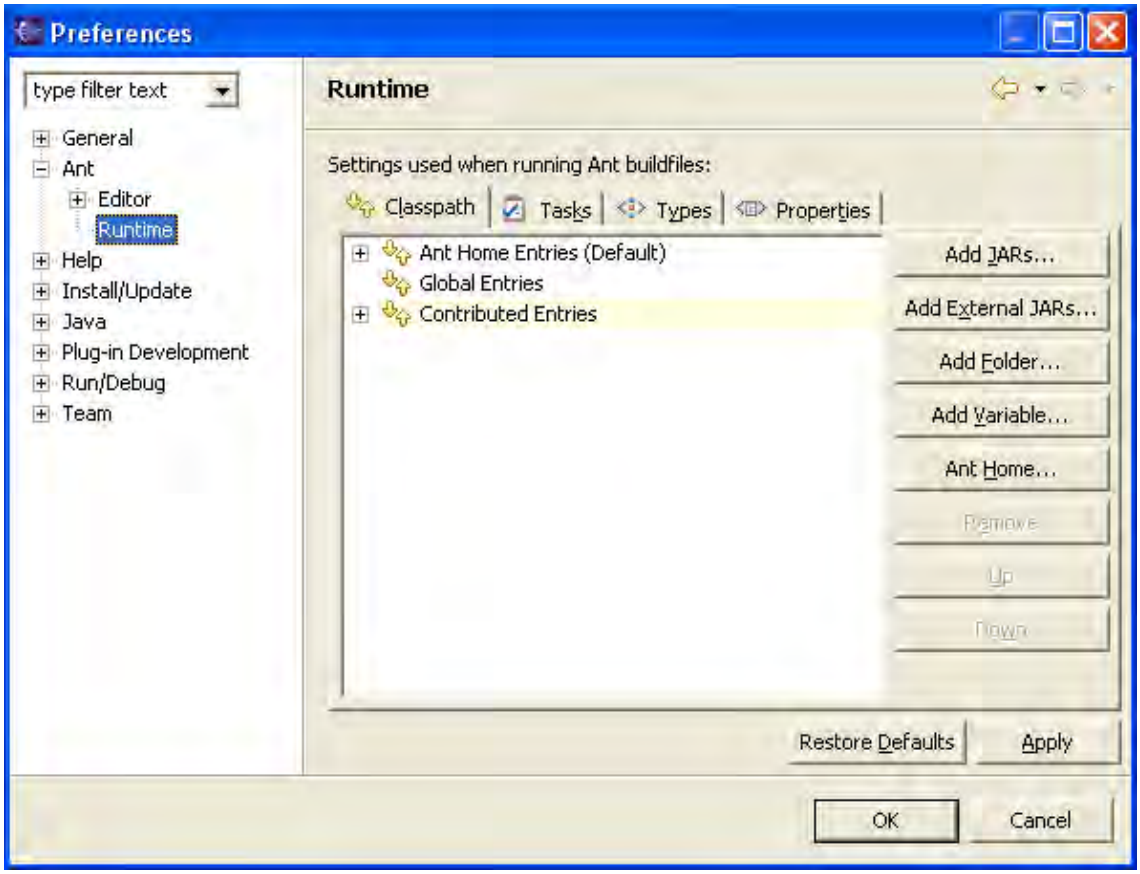
Ant support

Related reference

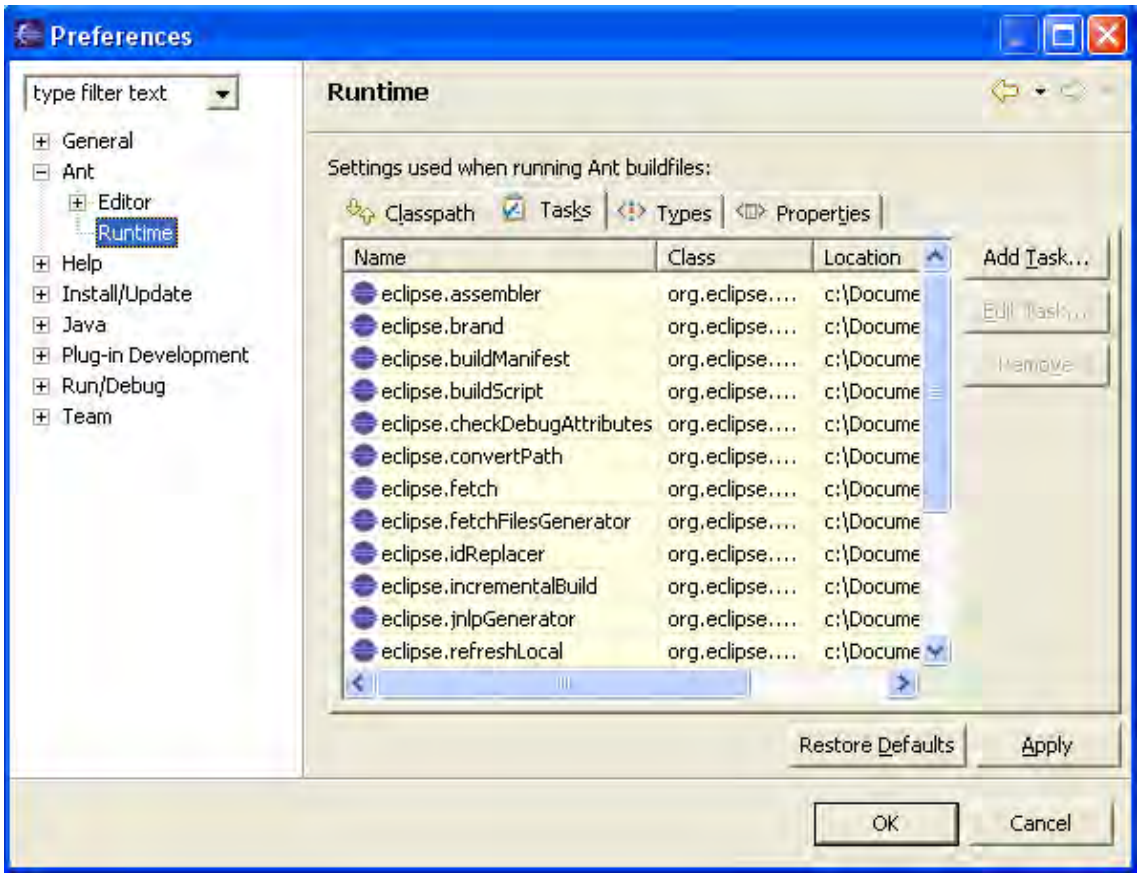
Ant runtime preferences
Ant editor preferences

Ant Runtime

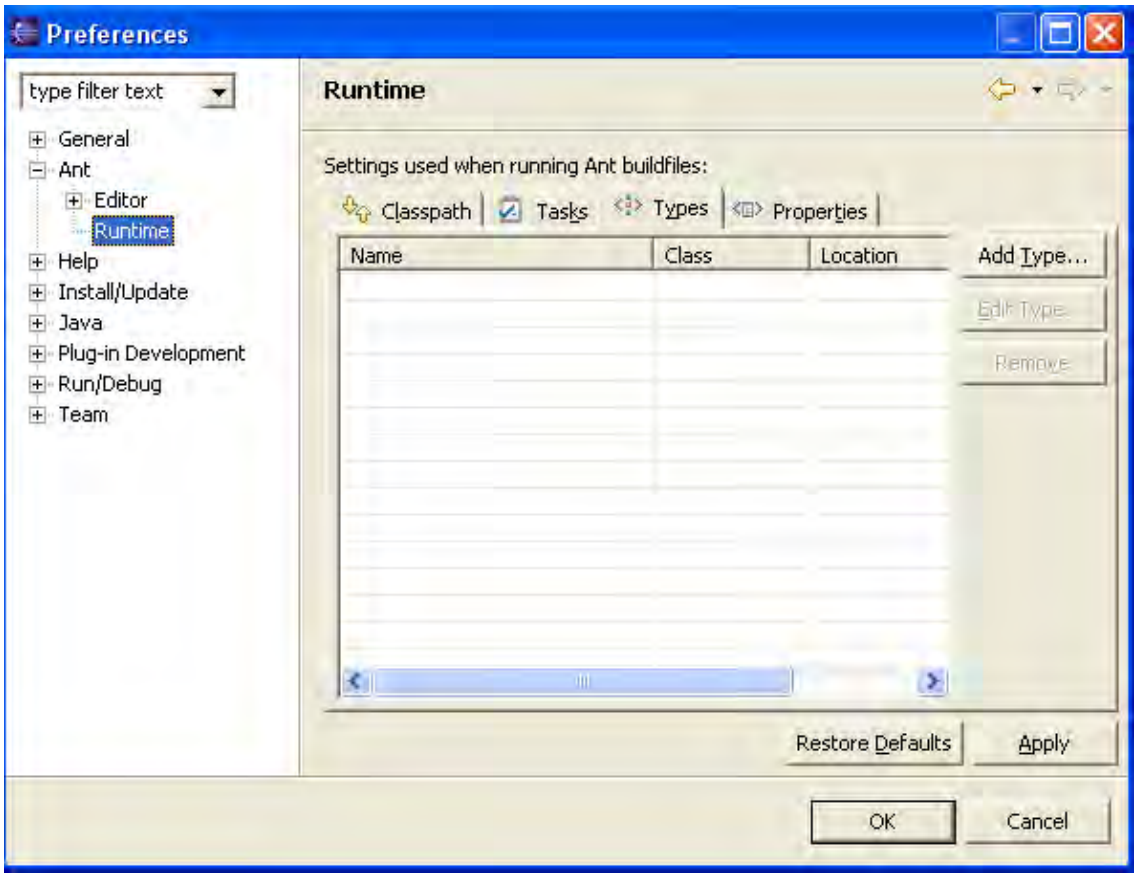
On the classpath page you can add additional classes defining tasks and types to the Ant classpath.



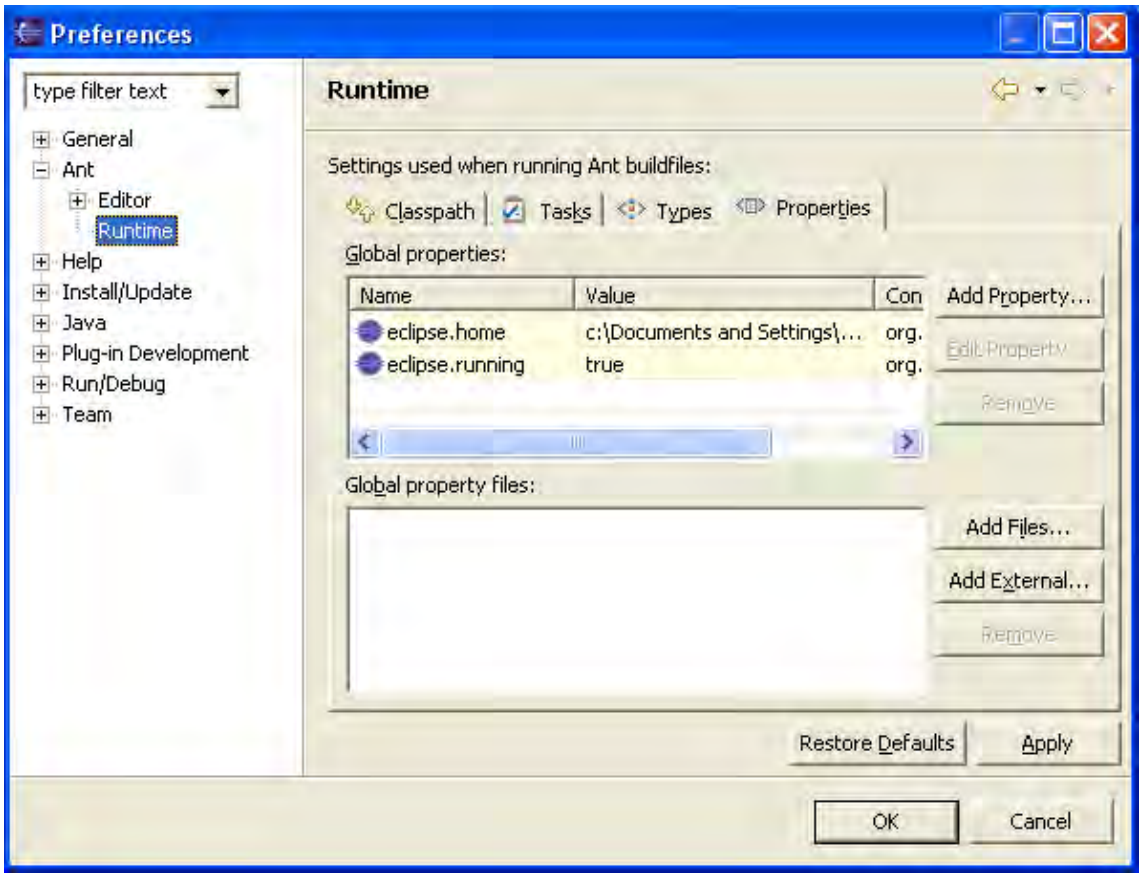
On the tasks page you can add tasks defined in one of the classes on the classpath.



On the types page you can add types defined in one of the classes on the classpath.



On the properties page you can add properties and property files that will be passed into Ant.



Related concepts

Ant support

Related reference

Ant preferences
Ant editor preferences

Ant Editor

The following preferences can be changed on the Ant Editor page.

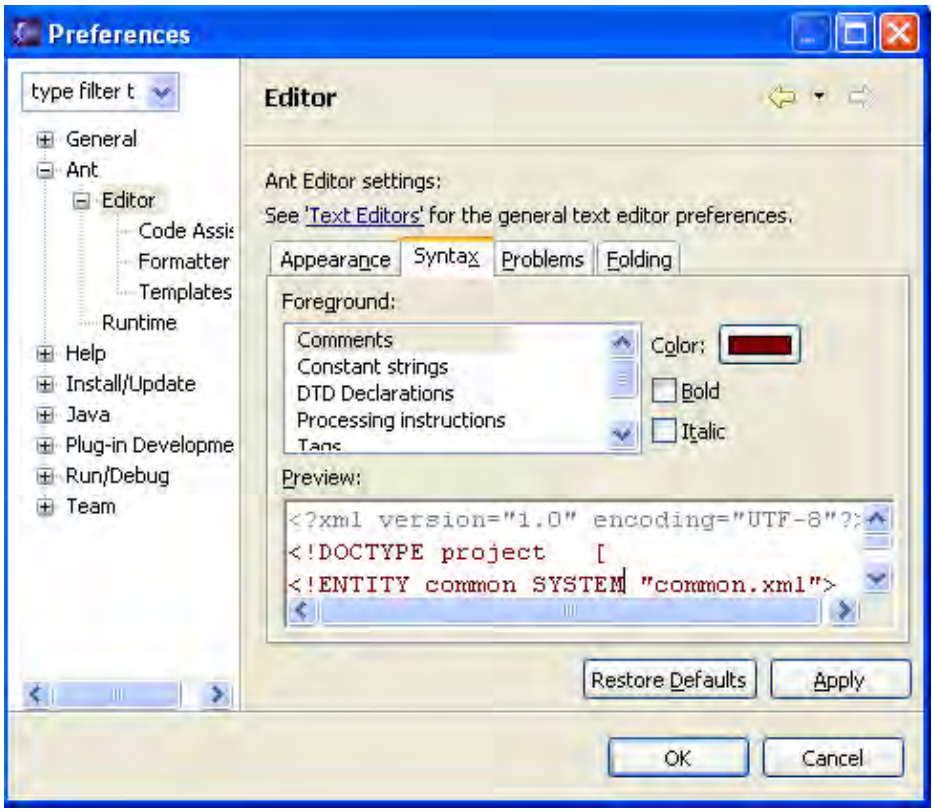
Appearance options

Option	Description	Default
Mark occurrences of the selected element in the current buildfile	This option allows you to select elements and mark occurrences in the current buildfile.	Off
Keep marks when selection changes	This option allows you to keep marks when the selection changes.	On
Insert spaces for tabs when typing	This option controls whether a tab is replaced with spaces when typed in the Ant editor.	Off
Show affordance in hover on how to make it sticky	This option controls whether or not the hover will display affordance on to make it sticky.	On

Here is what the Appearance tab of the Ant editor preference page looks like.

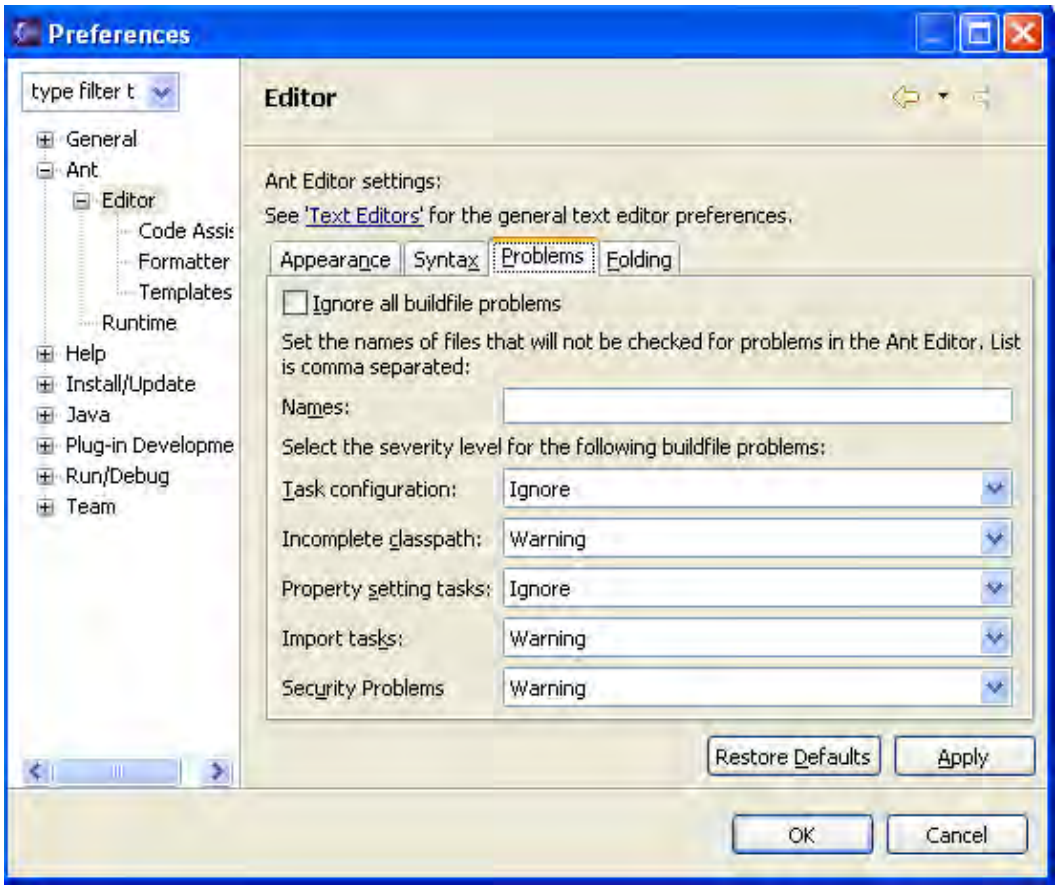
Syntax options

The Foreground options located on the Syntax tab, allow you to format the font of comments, constant strings, DTD declarations, processing instructions, tags and text. The Preview pane allows you to see your changes before you apply them. Here is what the Syntax tab of the Ant editor preference page looks like.



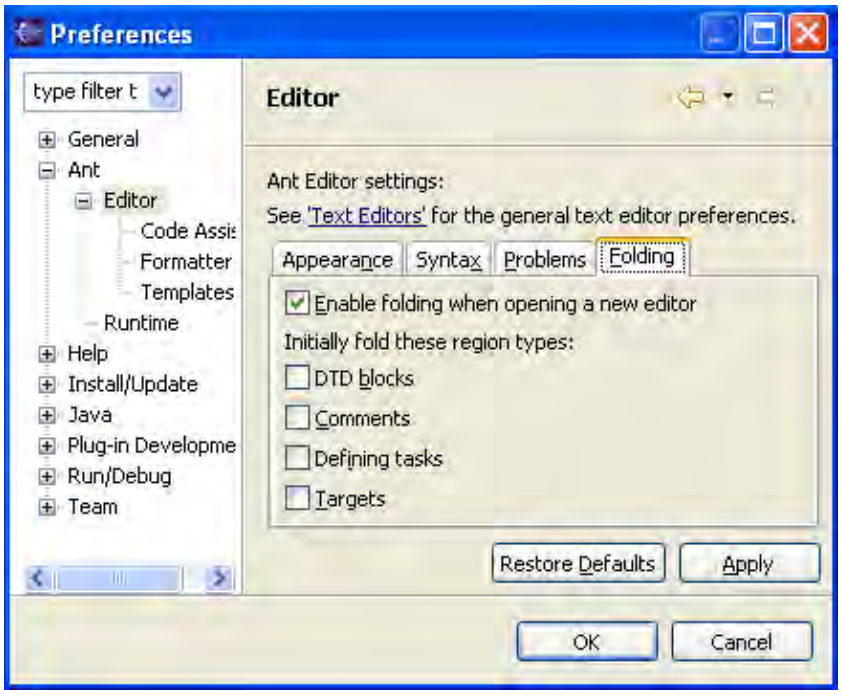
Problems options

The Problems options allow you to ignore all buildfile problems or to specify the specific files that you do not want the Ant Editor to check for problems. You can also set the severity level for buildfile problems such as task configuration, an incomplete classpath, property setting tasks, import tasks, and security problems. Here is what the Problems tab of the Ant editor preference page looks like.



Folding options

The Folding options allow you to enable folding when opening a new editor and to specify which region types should be folded. Here is what the Folding tab of the Ant editor preference page looks like.



Related concepts

Ant support

Related reference

Ant Editor
Ant preferences
Ant runtime preferences

Ant editor

The Ant editor provides specialized features for editor Ant buildfiles. Associated with the editor is a Ant buildfile specific Outline view which shows the structure of the Ant build file. It is updated as the user edits the buildfile

The editor includes the following features:

- Syntax highlighting
- Content/code assist (including Ant specific templates)
- Annotations

The most common way to invoke the Ant editor is to open an Ant buildfile from one of the navigation views or Package explorer using pop−up menus or by clicking the file (single or double−click depending on the user preferences).

Related concepts

Ant support
External tools

Related tasks

Running Ant buildfiles
Modifying the Ant classpath
Adding new Ant tasks and types
Using a different version of Ant

Related reference

Ant editor preferences
External Tools and Ant icons

External Tools and Ant icons

Objects

	Ant buildfiles
	Ant target containing an error
	Invalid project builder
	Default target
	Public Ant target (target with description)
	Ant internal target (target with no description)
	Jar file
	Ant property
	Ant task
	Ant type
	Ant import task
	Ant macrodef task

Launch configurations

	Launch external tool
	Ant launch configuration
	Program launch configuration
	Main tab
	Refresh tab
	Build tab
	Targets tab
	Properties tab
	Classpath tab

Ant View

	Ant view
	Add buildfile
	Add buildfiles with search
	Run selected buildfile or selected target
	Remove selected buildfiles
	Remove all buildfiles

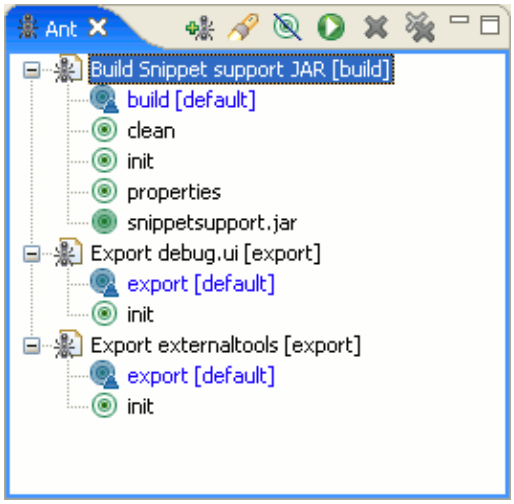
	Properties

Ant view

The Ant view provides a place to view Ant buildfiles and makes it easy to execute a buildfile or a given target within a buildfile. You can add files to the view and expand buildfiles to reveal the targets defined within them.

The pop−up menu for this view also allows you to open an editor on a buildfile, execute the selected target or buildfile, and configure the launch configuration used when executing.

Here is what the Ant view looks like:



Like other views in the Workbench, the Ant view has its own toolbar. Toolbar buttons are provided for adding buildfiles to the view and for executing them.

Toolbar

Add Buildfiles

This command prompts you to select buildfiles in the workspace to be added to the view.

Add Buildfiles with Search

This command searches for buildfiles and adds them to the Ant view. You can specify a file name pattern to search for and limit the scope of the search.

Hide Internal Targets

This command filters the Ant view so that internal targets are not displayed. Internal targets are targets with no description.

Run the Default Target of the Selected Buildfile

This command runs the default target of the selected buildfile or the selected target. The buildfile is executed using the launch configuration associated with the buildfile. Note that the target execution is based on the selection in the view; the targets selected in the launch configuration are ignored.

Remove Selected Buildfile

This command removes the selected buildfiles from the view.

Remove All Buildfiles

This command removes all buildfiles from the view.

Context menu

The context menu contains a few actions in addition to the actions available in the toolbar.

Run Ant

This command launches an Ant build for the selected buildfile or for a specific selected target of the buildfile

Run Ant...

This command brings up the launch configuration dialog to edit the configuration associated with the selected buildfile.

Refresh Buildfiles

This command refreshes the contents of the view to reflect the current state of all the represented buildfiles.

Open With >

This menu allows you to open an editor on the selected buildfile.

Related concepts

Ant support
External tools

Related tasks

Running Ant buildfiles
Modifying the Ant classpath
Adding new Ant tasks and types
Using a different version of Ant

Related reference

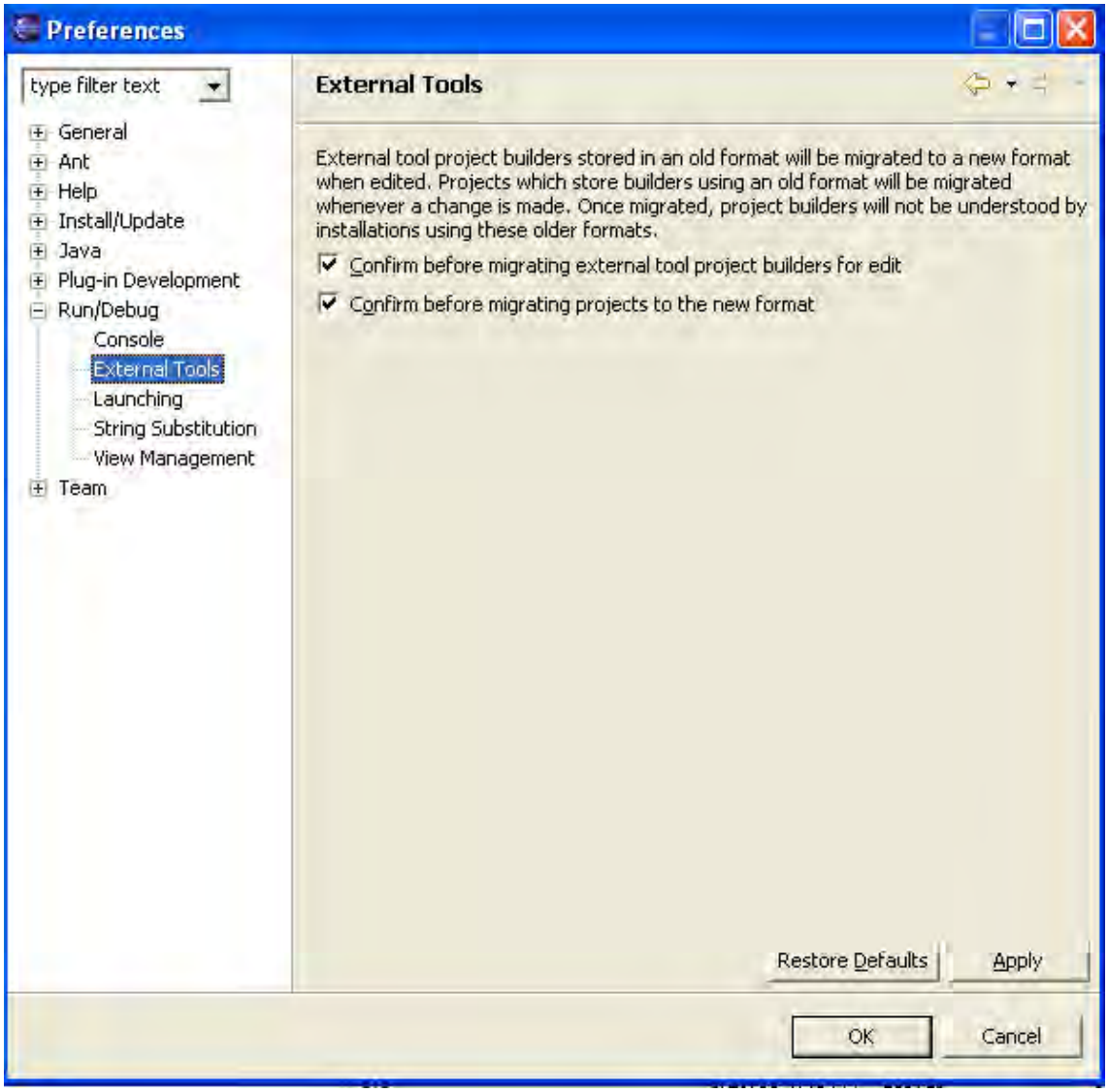
Ant runtime preferences

Ant Editor
External Tools and Ant icons

External Tools

The following preferences can be changed on the External Tools page.

You can configure whether or not you are prompted before external tool project builders are migrated.



Related concepts

Ant support

Related reference

Ant preferences
Ant runtime preferences
Ant editor preferences

Building resources

There are a number of ways that resources in the Workbench can be built. The scope of a build can be one or more selected projects, a working set, or the entire workspace.

A build will typically only operate on resources that have changed since the last build. A clean build will discard all existing built state, causing the next build to operate on all resources within the scope of the build.

Builds can be done automatically (each time resources are modified), or manually, using a menu item or keyboard shortcut. See the Related tasks links for more details.

Related concepts

Builds

Related tasks

Performing builds manually
Performing builds automatically
Saving resources automatically before a manual build

Performing builds manually

By default, builds are performed automatically when you save resources. If you need more control over when builds occur, you can disable automatic building and manually invoke builds. This is sometimes desirable in cases where you know building should wait until you finish a large set of changes. The disadvantage of manual building is that tasks generated to indicate build errors quickly become out of date until you build. In addition, it is very important that you remember to manually build before relying on build output (for example, before running your Java program).

Note: Some of the menu items described below are only available when the automatic build preference is disabled (ensure **Project > Build Automatically** is not checked).

To build projects in the workspace select the projects and click **Project > Build Project**. Alternatively, click **Project > Build All** to build all projects in the workspace. Both of these commands will search through the projects and only build the resources that have changed since the last build. To build all resources, even those that have not changed since the last build, run **Project > Clean...** before doing the build.

Related concepts

Builds

Related tasks

Saving resources automatically before a manual build
Changing build order
Performing builds automatically

Saving resources automatically before a manual build

To automatically save all modified resources in the Workbench before a manual build is done:

1. Click **Window** > **Preferences**.
2. Select the **General** category in the left pane to open the General Preferences page.
3. Select the **Save automatically before build** checkbox.
4. Click **OK** to close the Preferences page.

Related concepts

Builds

Related tasks

Building resources
Performing builds manually
Changing build order

Changing build order

By default, the Workbench computes the build order by interpreting project references as prerequisite relationships. Alternatively, you can explicitly define the order in which projects are built.

The Build Order preference page allows you to disable the **Use default build order** option so that you can access the projects list and manipulate the order of it.

To define the order in which the Workbench performs builds projects:

1. Click **Window** > **Preferences** to open the Preferences dialog.
2. Select the **Build Order** category in the left pane. The Build Order preference page opens, listing the projects for the build.
3. Ensure that the **Use default build order** checkbox is cleared. If it is selected, the buttons on this page are disabled, and builds are performed in the order of the project list.
4. (Optional) Click **Add Project** and **Remove Project** buttons to add and remove projects from the list.
5. Select one or more projects in the list and click **Up** or **Down** to to set the preferred project build order.
6. Click **OK** to close the Preferences dialog.

Related concepts

Builds

Related tasks

Performing builds manually
Performing builds automatically

Performing builds automatically

To indicate that you want the Workbench to perform incremental builds whenever resources are saved:

1. Click **Window** > **Preferences**.
2. Select the **General** category in the left pane. The General Preferences page opens.
3. Select the **Build automatically** checkbox.
4. Click **OK** to close the Preferences page. The Workbench will automatically perform incremental builds of resources modified since the last build. Whenever a resource is modified, another incremental build occurs.

Related concepts

Builds

Related tasks

Performing builds manually
Changing build order

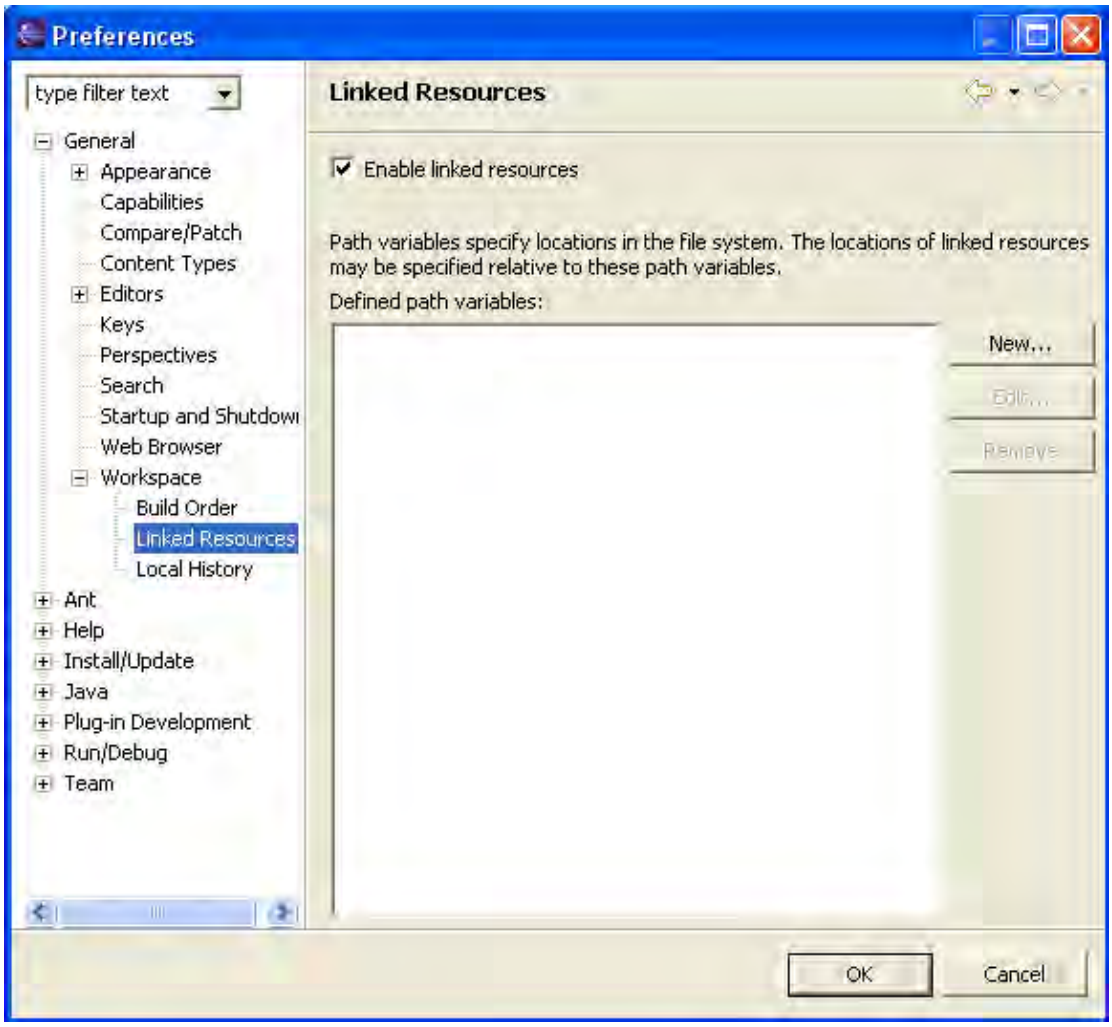
Linked Resources

This preference page is used when working with linked resources. The preference **Enable linked resources** is used to globally enable or disable the linked resource feature for the entire workspace. By default, linked resources are enabled. If you disable linked resources, then you will not be able to create any new linked resources, or import existing projects that contain linked resources.

Not all versions of the workbench support linked resources and recognize them as such. You may not want to use linked resources if you plan to share your workspace data with other users. Disable this preference if they will not be able to work with linked resources.

The remainder of this page is for defining path variables that are used when creating linked resources. Use the **New** button to define new variables, the **Edit** button to change the value of an existing variable, and the **Remove** button to get rid of an existing variable. Note if you change a path variable that is currently in use, you will need to perform a local refresh on those projects to "discover" what is different in the file system. You can refresh a resource by opening the one of the navigation views' context menu for that resource and selecting **Refresh**. It is not recommended that you remove a path variable that is currently in use.

Here is what the Linked Resources preference page looks like:



Related concepts

Linked resources
Path variables

Path variables

Path variables specify locations on the file system. The location of linked resources may be specified relative to these path variables. They allow you to avoid references to a fixed location on your file system.

By using a path variable, you can share projects containing linked resources with team members without requiring the exact same directory structure as on your file system.

You can load a project that uses path variables even if you do not currently have all the path variables defined in your workspace. A linked resource that uses a missing path variable is flagged using a special decorator icon. In addition, the **File > Properties > Info** property page and the **Window > Show View > Other > Basic > Properties** view for a linked resource indicate the variable and whether it is defined or not. A path variable can also specify a location that does not currently exist on the file system. Linked resources that use such a path variable are indicated using the same decorator icon mentioned above.

You can create new path variables and edit and remove existing path variables on the **General > Workspace > Linked Resources** preference page.

Related concepts

Linked resources

Related tasks

Creating linked resources
Viewing resource properties

Related reference

Linked resources

New Folder wizard

This wizard helps you create a new folder in the Workbench.

Here is what the New Folder wizard looks like:

New Folder Fields

Field	Description	Default
Enter or select the parent folder	The resource in which the new folder will be created. Type or navigate the list to select the resource.	The resource that was selected when you chose to create the new folder
Folder name	The name for the new folder.	<blank>

Advanced

The **Advanced** button reveals or hides a section of the wizard used to create a linked folder. Check the **Link to folder in the file system** checkbox if you want the new folder to reference a folder in the file system. Use the field below the checkbox to enter a folder path or the name of a path variable. Use the **Browse...** button to browse for a folder in the file system. Use the **Variables...** button if you want to use a path variable to reference a file system folder.

Related concepts

Linked resources
Path variables

Related tasks

Creating a folder
Creating linked resources

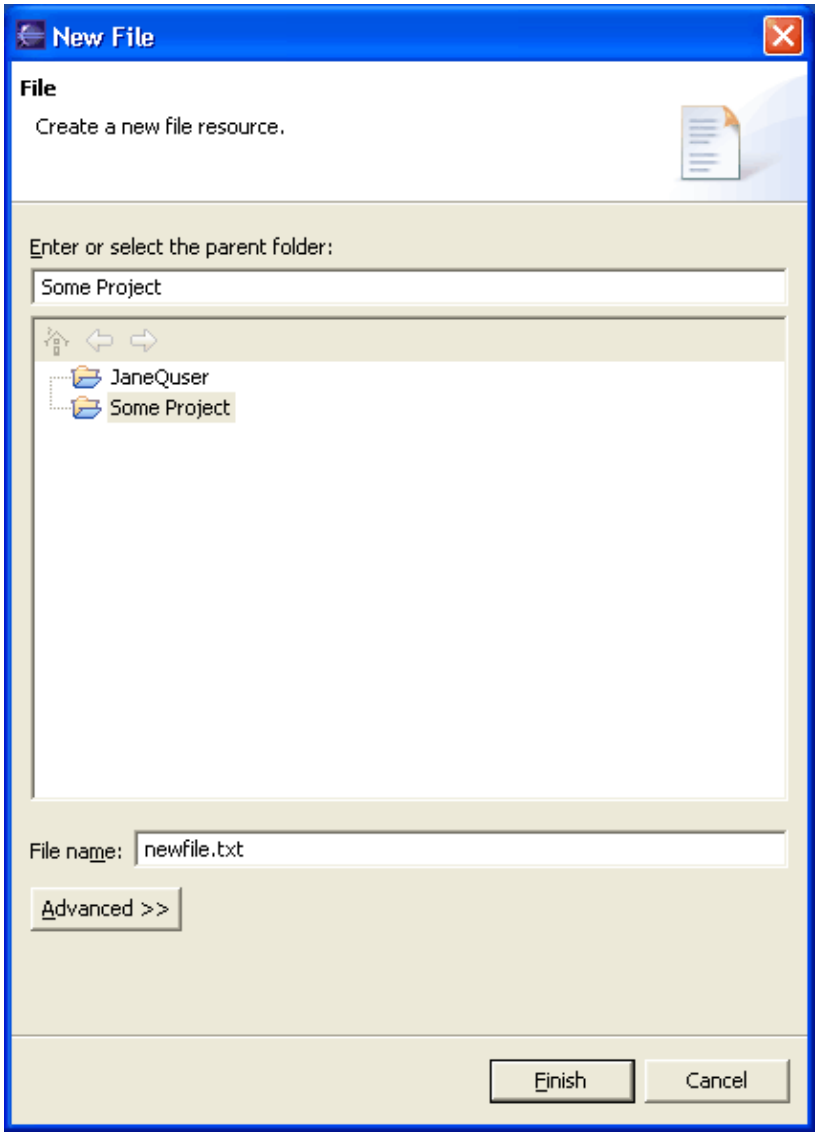
Related reference

Navigator View

New File wizard

This wizard helps you create a new file in the Workbench.

Here is what the New File wizard looks like:



New File Fields

Field	Description	Default
Enter or select the parent folder	The resource in which the new file will be created. Type or browse the list to select the resource.	The resource that was selected when you invoked the New File wizard.
File name	The name for the new file, including the file extension.	<blank>

Advanced

The **Advanced** button reveals or hides a section of the wizard used to create a linked file. Check the **Link to file in the file system** checkbox if you want the new file to reference a file in the file system. Use the field below the checkbox to enter a file path or the name of a path variable. Use the **Browse...** button to browse for a file in the file system. Use the **Variables...** button if you want to use a path variable to reference a file system file.

Related concepts

Linked resources
Path variables

Related tasks

Creating a file
Creating linked resources

Related reference

Navigator View

Importing

You can import files into the Workbench three ways:

- By using the import wizard
- WIN By dragging files or folders from the file system to one of the navigation views
- WIN By copying files or folders from the file system and pasting them into one of the navigation views

See the related tasks section for more details.

Related concepts

Resources

Related tasks

Importing resources from the file system
Importing resources from an Archive File
Exporting

Importing resources from the file system

You can use the Import Wizard to copy files from a file system directory into the Workbench.

1. From the main menu bar, select **File > Import**. The Import wizard opens.
2. Select **File System** and click **Next**.
3. Click the **Browse** button on the next page of the wizard to select the directories from which you would like to add the resources.
4. In the import selection panes, use the following methods to select exactly the resources you want to add:
    - Expand the hierarchies in the left pane and select or clear the checkboxes that represent the folders in the selected directory. Then in the right pane, select or clear checkboxes for individual files.
    - Click **Filter Types** to filter the current selection for files of a specific type.
    - Click **Select All** to select all resources in the directory, then go through and deselect the ones that you do not want to add.
    - Click **Deselect All** to deselect all resources in the directory, then go through and choose individual resources to add.
5. Specify the Workbench project or folder that will be the import destination.
6. When you have finished specifying your import options, click **Finish.**

Tip: You can also import folders and files by dragging them from the file system and dropping them into one of the navigation views, or by copying and pasting.

Related concepts

Resources

Related reference

Import wizard

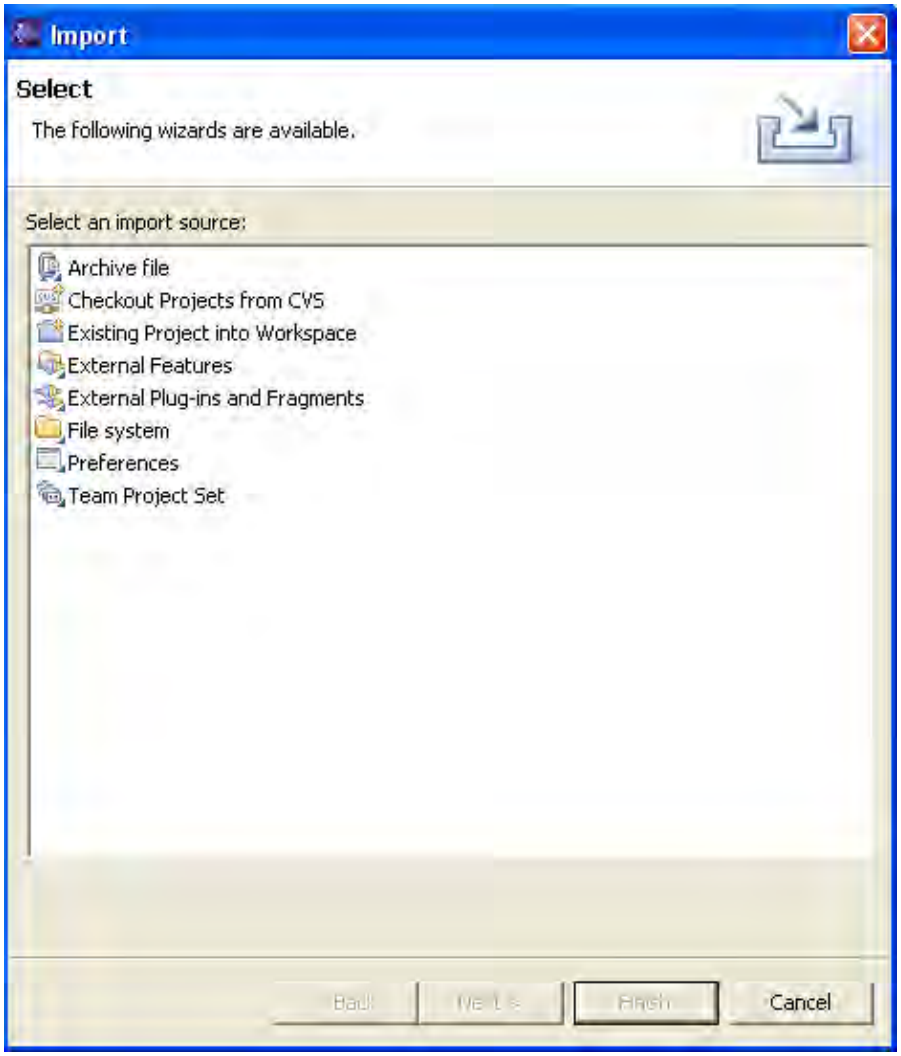
Related tasks

Importing existing projects
Importing resources from a ZIP File
Exporting

Import wizard

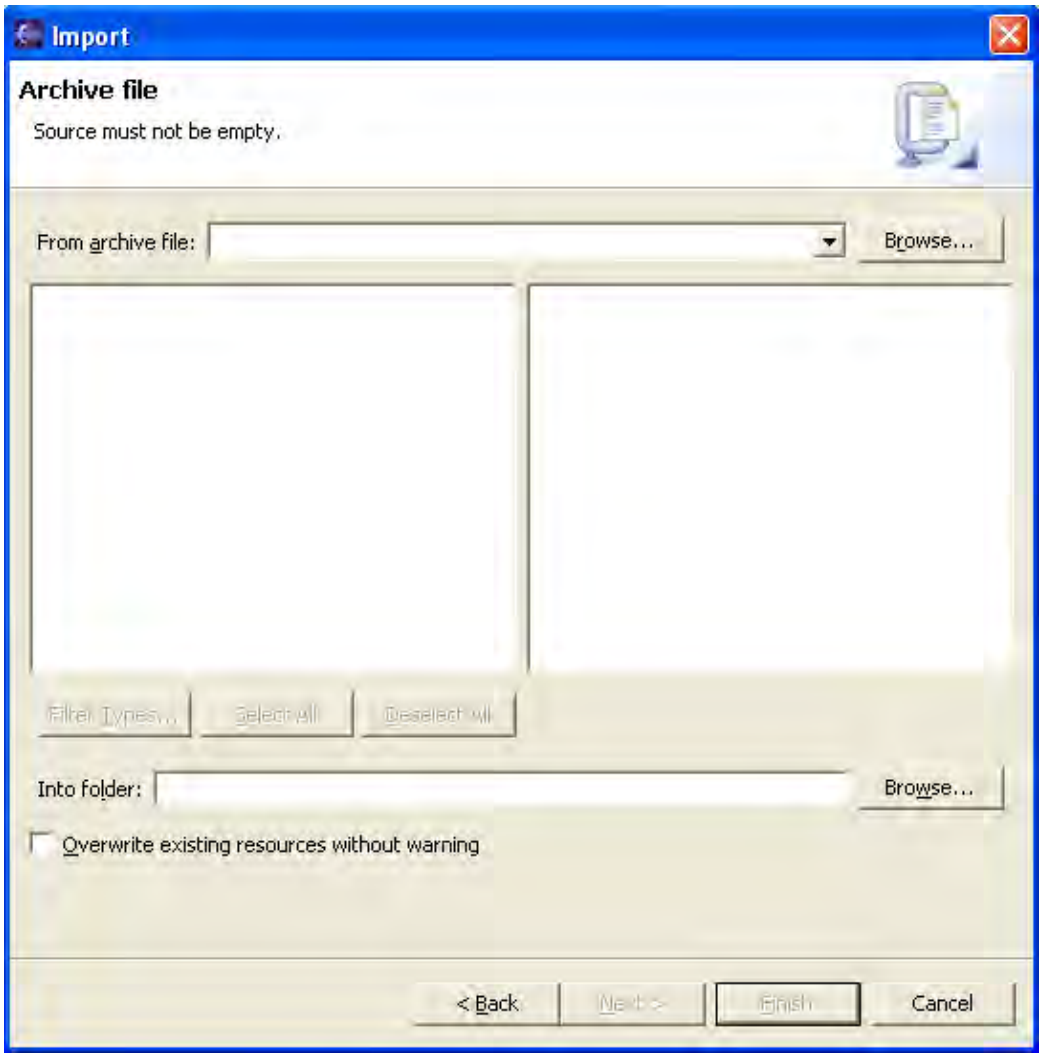
This wizard helps you import resources into the Workbench.

When the Import wizard first comes up, you must choose what type of import to do:



Archive File

If you choose this option, you will import files from an archive file.



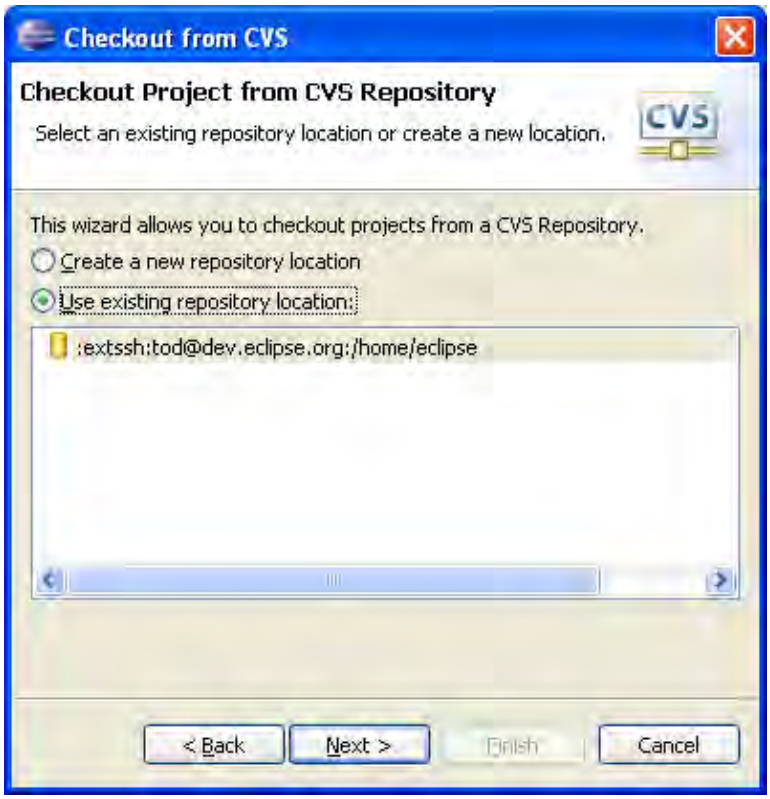
Import − Archive File Options

Option	Description	Default
Archive File	The file from which to import. Type in the full path or Browse to select the path on the file system.	<blank>
Filter Types...	Dialog to select which file types to import. Use this to restrict the import to only certain file types.	*N/A*
Select All	Check off all resources for import	*N/A*
Deselect All	Uncheck all resources.	*N/A*
Folder	The folder into which the resources will be imported. Type the path or Browse to select a path in the Workbench.	The folder holding the selected resource
Overwrite existing resources without warning	Determines whether importing a resource should silently overwrite a resource which already exists in the Workbench. If this option is off, you will be prompted before a given resource is overwritten, in which case	Off

	you can either overwrite the resource, skip it, or cancel the import.	

Checkout projects from CVS

Import a project from a CVS Repository.

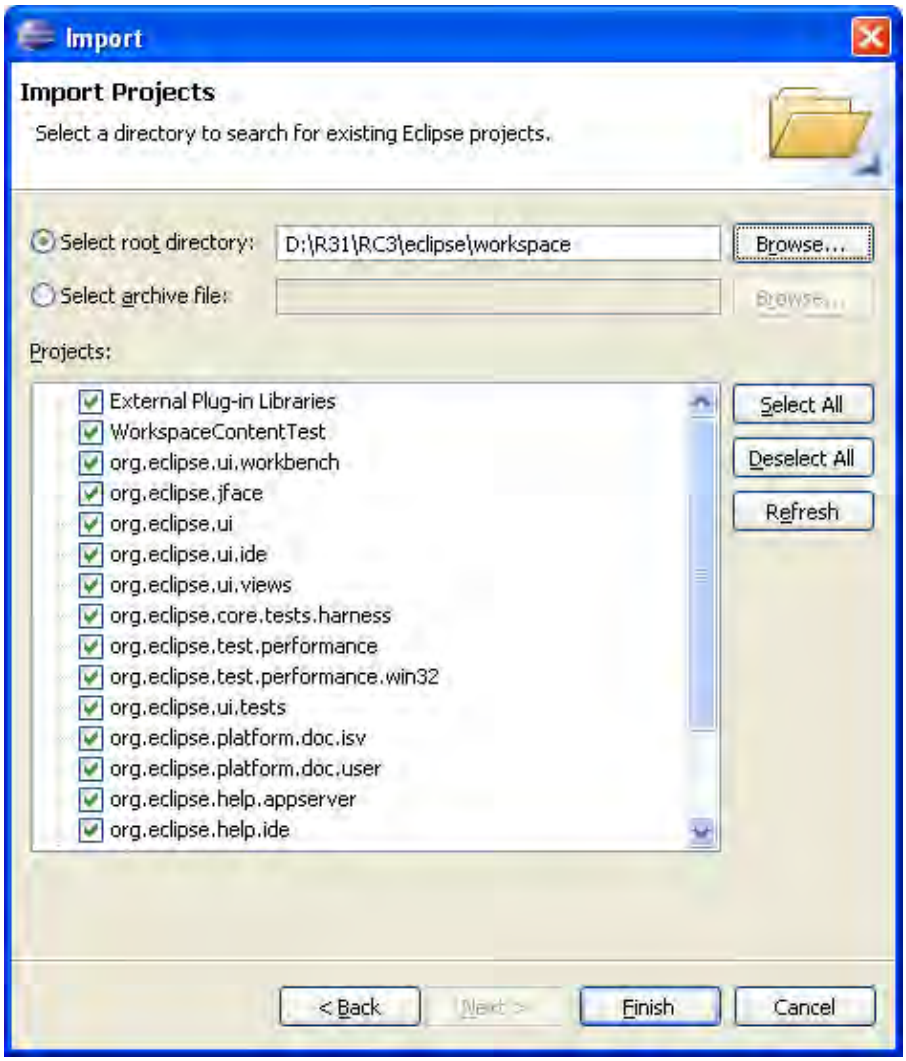


Import − Checkout projects from CVS Options

Option	Description	Default
Create a new repository location	Go to the New CVS Repository Location wizard.	<disabled>
Use existing repository location	Use a previously datafilled CVS location.	*enabled*

Existing Project into Workspace

Imports a project into this workspace that was previously located in this workspace, or that currently exists in another workspace.

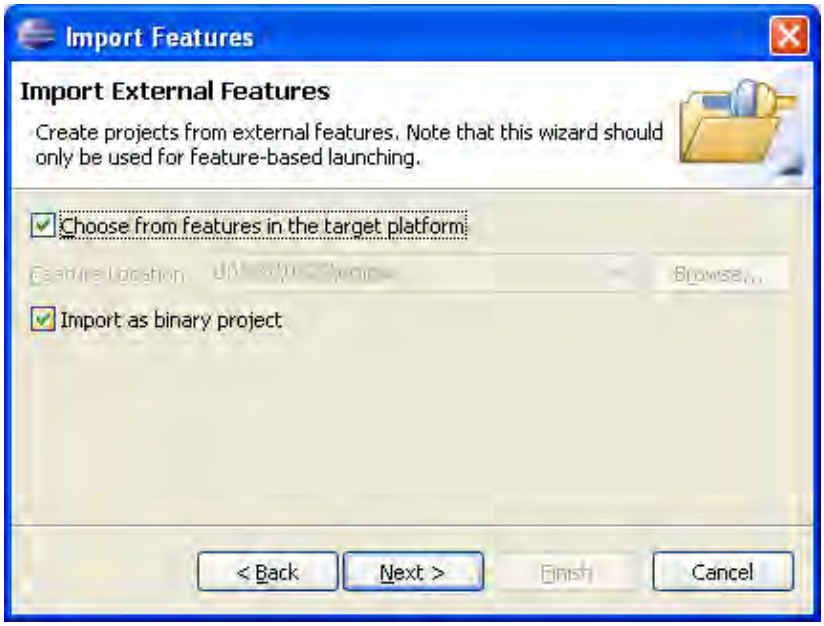


Import − Existing Project Options

Option	Description	Default
Select root directory	Root directory in the File System to start scanning for projects to import. Type in the full path or Browse to select the path on the file system.	<blank>
Select archive file	Archive file to scan for projects to import. Type in the full path or Browse to select the archive on the file system.	<disabled>
Select All	Check all of the projects that were found for import.	
Deselect All	Uncheck all projects.	
Refresh	Rescan the selected source for projects to import.	

External Features

Create projects from external features. Note that this wizard should only be used for feature−based launching.

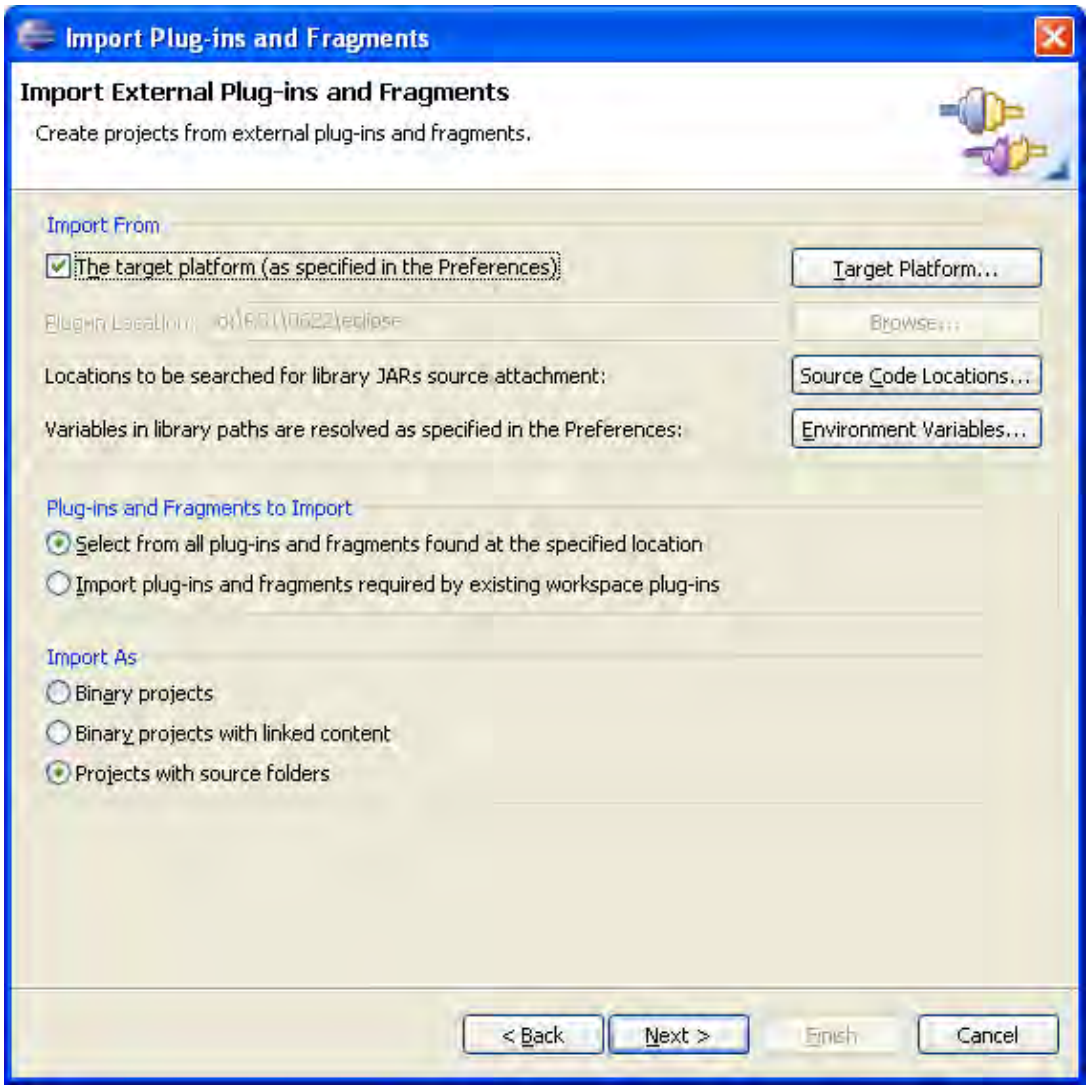


Import − External Feature Options

Option	Description	Default
Choose from features in the target platform	Scan the target platform for external features to import.	<checked>
Feature Location	The install location to search for features. It defaults to the target platform.	<disabled>
Import as binary project	Import the feature as a binary project.	<checked>

External Plug−ins and Fragments

Create projects from external plug−ins and fragments.

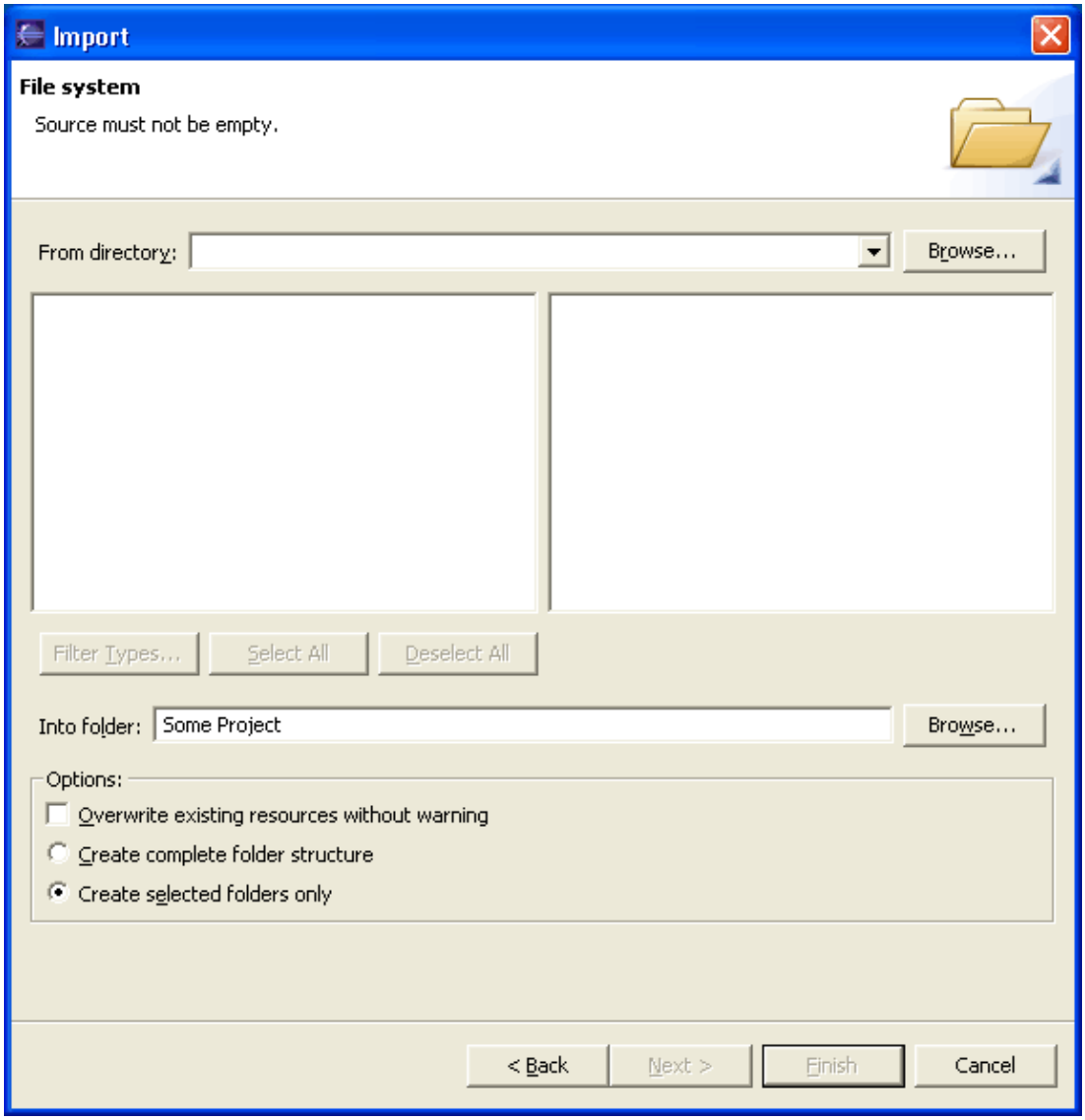


Import − External Plug−ins and Fragments Options

Option	Description	Default
Import From (category)		
Target Platform	The target platform (as specified in the Preferences).	<checked>
Plug−in Location	The install location to search for plug−ins and fragments. It defaults to the target platform.	<disabled>
Source Code Locations	Locations to be searched for library JARs source attachments.	
Environment Variables	Variables in library paths are resolved as specified in the Preferences.	
Import As (category)		
Binary projects	Import into the workspace but do not expand contents	
Binary projects with linked content	Only link to the external files	<checked>
Projects with source folders	Extract the contents to source files	

File System

If you choose this option, you will import files from the file system.



Import − File System Options

Option	Description	Default
Directory	The directory from which to import files. Select a previous path from the drop down combo or Browse to select the path in the file system.	<blank>
Filter Types	Dialog to select which file types to import. Use this to restrict the import to only certain file types.	*N/A*
Select All	Check off all files and folders for import.	*N/A*
Deselect All	Uncheck all resources.	*N/A*
Folder		

	The folder into which the resources will be imported. Type the path or Browse to select a path in the Workbench.	The folder holding the selected resource
Overwrite existing resources without warning	Determines whether importing a resource should silently overwrite a resource which already exists in the Workbench. If this option is off, you will be prompted before a given resource is overwritten, in which case you can either overwrite the resource, skip it, or cancel the import.	Off
Create complete folder structure	Create hierarchy (folder) structure in the Workbench to accommodate the resources being imported, and all parent folders of those resources in the file system.	Off
Create selected folders only	Create hierarchy (folder) structure in the Workbench to accommodate the resources being imported.	On

Preferences

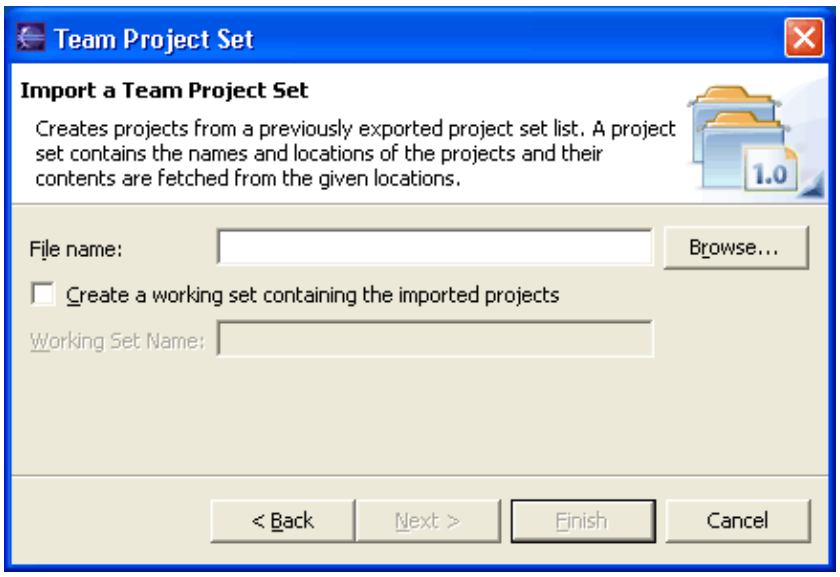
Import preferences from the local file system.

Import − Preferences Options

Option	Description	Default
From preference file	The file from which to import preferences. Select a previous file from the drop down combo or Browse to select the file in the file system.	<blank>
Import All	Import all of the preferences.	<checked>
Choose specific preferences to import	Choose from the preferences contained in the file, like CVS connection preferences or JRE preferences.	
Select All	Check off all files and folders for import.	*N/A*
Deselect All	Uncheck all resources.	*N/A*

Team Project Set

Imports a description of the repository and version control information for a set of projects. This allows you to synchronize correctly with the appropriate project state in your repository.



Import − Team Project Options

Option	Description	Default
File name	The name of the team project set export file	<blank>

Importing existing projects

You can use the Import Wizard to copy a project that exists in a different workspace, or one that previously existed in a workspace, into the Workbench.

1. From the main menu bar, select **File > Import**. The Import wizard opens.
2. Select **Existing Project into Workspace** and click **Next**.
3. Choose either ***Select root directory*** or ***Select archive file*** and click the associated **Browse** to locate the directory or file containing the projects.
4. Under ***Projects*** select the project or projects which you would like to import.
5. Click **Finish** to start the import.

Related reference

Import wizard

Related tasks

Importing resources from the file system
Importing resources from a ZIP File
Exporting

Importing resources from an Archive file

You can use the Import wizard to extract files from an archive file into the Workbench.

1. From the main menu bar, select **File > Import**. The Import wizard opens.
2. Select **Archive file** and click **Next**.
3. Click the **Browse** button on the next page of the wizard, to select the archive files that contain the files you want to extract and import into the Workbench.
4. In the import selection panes, use the following methods to select exactly the resources you want to add:
    - Expand the hierarchies in the left pane and select or clear the checkboxes that represent the folders in the selected directory. Then in the right pane, select or clear checkboxes for individual files.
    - Click **Filter Types** to filter the current selection for files of a specific type.
    - Click **Select All** to select all resources in the directory, then go through and deselect the ones that you do not want to add.
    - Click **Deselect All** to deselect all resources in the directory, then go through and choose individual resources to add.
5. Specify the Workbench project or folder that will be the import destination.
6. When you have finished specifying your import options. click **Finish.**

Related concepts

Resources

Related reference

Import wizard

Related tasks

Importing existing projects
Importing resources from the file system
Exporting

Exporting

You can export files from the Workbench in three ways:

- By using the Export wizard.
- WIN By dragging files or folders from one of the navigation views to the file system.
- WIN By copying files or folders from one of the navigation views and pasting them into the file system.

See the related tasks section for more details.

Related concepts

Resources

Related tasks

Exporting resources to the file system
Exporting resources to an Archive File
Importing

Exporting resources to the file system

You can use the Export wizard to export resources from the Workbench to the file system.

1. In one of the navigation views, select the resources that you want to export.
2. From the main menu bar, select **File > Export**. The Export wizard opens.
3. Select **File System** and click **Next**.
4. By default, the resources that you selected will be exported, along with all their children. Optionally, use the checkboxes in the left and right panes to select the set of resources to export, and use push buttons such as **Select Types** to filter the types of files that you want to export.
5. Click the **Browse** button on the next page of the wizard, to select the directory you would like to export the resources to.
6. Specify the directory in the file system that will be the export destination.
7. Click **Finish**.

Tip: You can also export folders and files by dragging them from one of the navigation views to the file system and dropping them in the file system, or by copy and paste.

Related concepts

Resources

Related tasks

Importing
Exporting resources to a ZIP File

Exporting resources to an Archive file

You can use the Export wizard to export resources from the Workbench to an archive file in the file system.

1. In one of the navigation views, select the resources that you want to export.
2. From the main menu bar, select **File > Export**. The Export wizard opens.
3. Select **Archive file** and click **Next**.
4. By default, the resources that you selected will be exported along with their children. Optionally, use the checkboxes in the left and right panes to select the set of resources to export, and use push buttons such as **Select Types** to filter the types of files that you want to export.
5. Specify the path and name of the archive file into which you want to export the selected resources.
6. Click **Finish.**

Related concepts

Resources

Related tasks

Exporting resources to the file system
Importing

Toolbar buttons

The following buttons may appear in the Workbench toolbar, toolbars for views, and the shortcut bar:

Button	Description	Button	Description
	Open a new perspective		Save the active editor contents
	Save the contents of all editors		Save editor contents under a new name or location
	Opens the search dialog		Print editor contents
	Open a resource creation wizard		Open a file creation wizard
	Open a folder creation wizard		Open a project creation wizard
	Open the import wizard		Open the export wizard
	Run incremental build		Run a program
	Debug a program		Run an external tool or Ant
	Cut selection to clipboard		Copy selection to clipboard
	Paste selection from clipboard		Undo most recent edit
	Redo most recent undone edit		Navigate to next item in a list
	Navigate to previous item in a list		Navigate forwards
	Navigate backwards		Navigate up one level
	Add bookmark or task		Open a view's drop down menu
	Close view or editor		Pin editor to prevent automatic reuse
	Filter tasks or properties		Go to a task, problem, or bookmark in the editor
	Restore default properties		Show items as a tree
	Refresh view contents		Sort list in alphabetical order
	Cancel a long running operation		Delete selected item or content
	Last edit location		Toggle Mark Occurrences
	Show source of selected element only		

Related concepts

Toolbars

Toolbars

There are four kinds of toolbars in the Workbench.

The *main toolbar*, sometimes called the Workbench toolbar, is displayed at the top of the Workbench window directly beneath the menu bar. The contents of this toolbar change based on the active perspective. Items in the toolbar might be enabled or disabled based on the state of either the active view or editor. Sections of the main toolbar can be rearranged using the mouse.

There are also individual *view toolbars*, which appear in the title bar of a view. Actions in a view's toolbar apply only to the view in which they appear. Some view toolbars include a **Menu** button, shown as an inverted triangle, that contain actions for that view.

A third type of toolbar is the perspective switcher. The perspective switcher allows quick access to perspectives that are currently open. It also has a button that can open new perspectives. The perspective switcher is normally located in the top−right, next to the main toolbar. However, it is also possible to position it below the main toolbar ("top−left"), or to position it vertically on the left−hand side of the workbench ("left"). The name of the perspectives is shown by default, but it is possible to hide the text and show only the icons. To reposition the perspective or hide the text, right−click on it and choose the appropriate item from the context menu.

Finally, the fast view bar is a toolbar that contains icons representing the current set of fast views. A fast view is a shortcut to a view that is frequently used; see the section on fast views for more information. The fast view bar appears in the bottom left corner of the workbench by default. However, it is possible to position it on the left or right as well.

In all cases, you can find out what toolbar buttons do by moving your mouse pointer over the button and reading the tooltip that opens. See the list of related reference topics below for a table of all toolbar buttons.

Related concepts

Workbench
Views
Perspectives
Fast views

Related tasks

Creating fast views
Rearranging the main toolbar

Related reference

Toolbar buttons

Rearranging the main toolbar

You can rearrange sections of the main toolbar. Toolbar sections are divided by a thin vertical line.

1. Make sure the toolbar is unlocked. The toolbar is unlocked if it has thick vertical bars next to the thin vertical toolbar dividers.

   If it is locked, unlock the toolbar by right clicking the toolbar and selecting the **Lock the Toolbars** menu item.
2. Grab the section of the toolbar you want to rearrange by moving the mouse over the thick vertical line on the left side of the desired segment. The mouse cursor changes its shape to indicate that you can click to move the toolbar section.
3. Click and hold the left mouse button to grab the toolbar section.
4. Move the section left and right or up and down. Release the mouse button to place it in the new location.
5. To prevent accidental changes to the toolbar lock it again by right clicking the toolbar and selecting the **Lock the Toolbars** menu item.

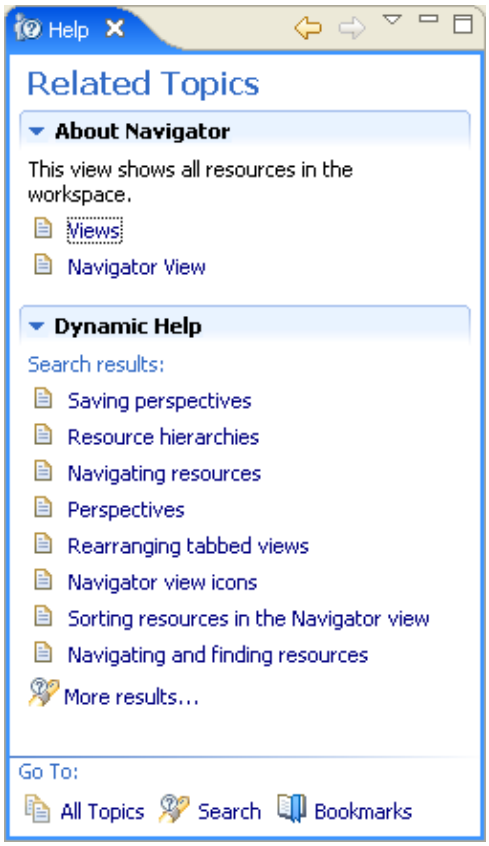
Related concepts

Toolbars

Help view

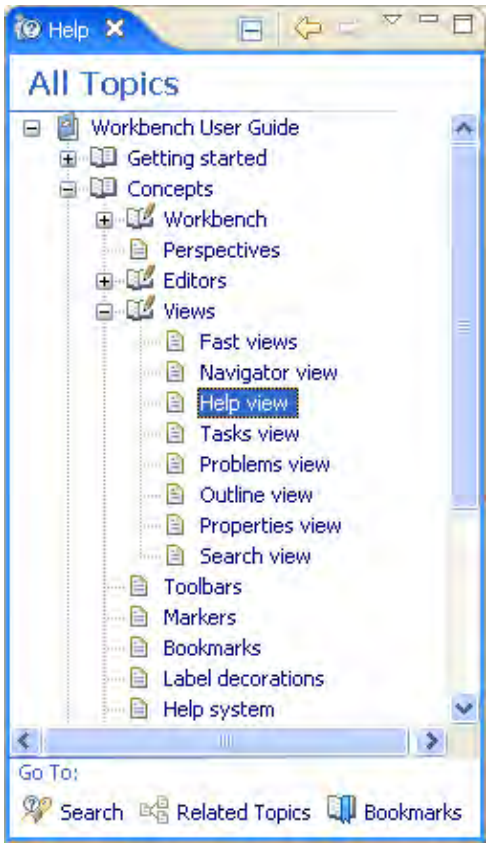
The Help view provides user assistance inside the Workbench. The view consists of four pages providing user assistance. Each page presents help topics in a slightly different fashion. Hyper links at the bottom of the help view allow switching among pages, and clicking any topic will display its contents.

Related Topics



The Related Topics page shows description and help topics related to the current workbench context. The topics include context help, as well as the result of a local help search on the current context, presented in "Dynamic Help" section of the page. For example, if the currently active workbench part is 'Package Explorer' inside the Java perspective, local help will find and present topics describing "Package Explorer view" or "Java perspective". The Related Topics page tracks changes in the workbench and continuously updates displayed information.

All Topics

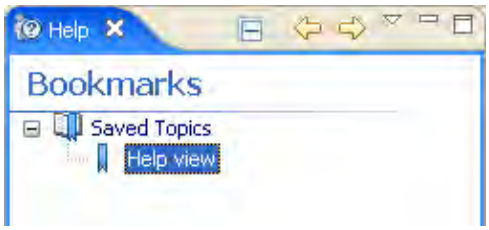


The All Topics page shows the Table of Contents. It is a hierarchy of all help topics arranged in a tree The tree branches can be expanded to browse topics that can be displayed using a single mouse click.

Search

The Search page allows locating local topics, remote documents, and other documents given a search query. Links to search hits are displayed along with a summary of topic contents. Search scope controls a subset of documentation being searched. Multiple search scopes can be configured, each defining a custom set of resources from among local documentation, additional local search engines or remote engines on the web.

Bookmarks



The Bookmarks shows topics marked as personal bookmarks.

Toolbar

The toolbar of the Help view contains the following buttons. The available buttons depends on the currently displayed page.

Back
: Displays the page or the topic that was displayed immediately prior to the current display.

Forward
: Displays the page or the topic was displayed immediately after the current display.

Collapse All
: Collapses the tree expansion state of all topics on the page.

Show All Topics
: When filtering of workbench elements is enabled, the buttons enables displaying topics for disabled elements.

Show Result Categories
: Switches between sorting search results by relevance and by logical containers for example books.

Show Result Descriptions
: Displays short description of each search hit, when available.

Menu
: Provides menu items that allow you switching help view pages.

Icons

The following icons can appear in the Navigator view.

Icon	Description
	Book (closed)
	Book (closed)
	Topic (container)
	Topic
	Search of local help
	Search of remote info−center
	Search of web engine
	Open link in a help window
	Bookmark document
	Bookmarked document

Related concepts

Views

Related tasks

Opening views
Moving and docking views
Creating fast views

Related reference

Help window

Help window

This window displays help on using the Workbench. If you are reading this document, you've probably found this view already.

Here is what the Help view looks like, when using embedded browser:

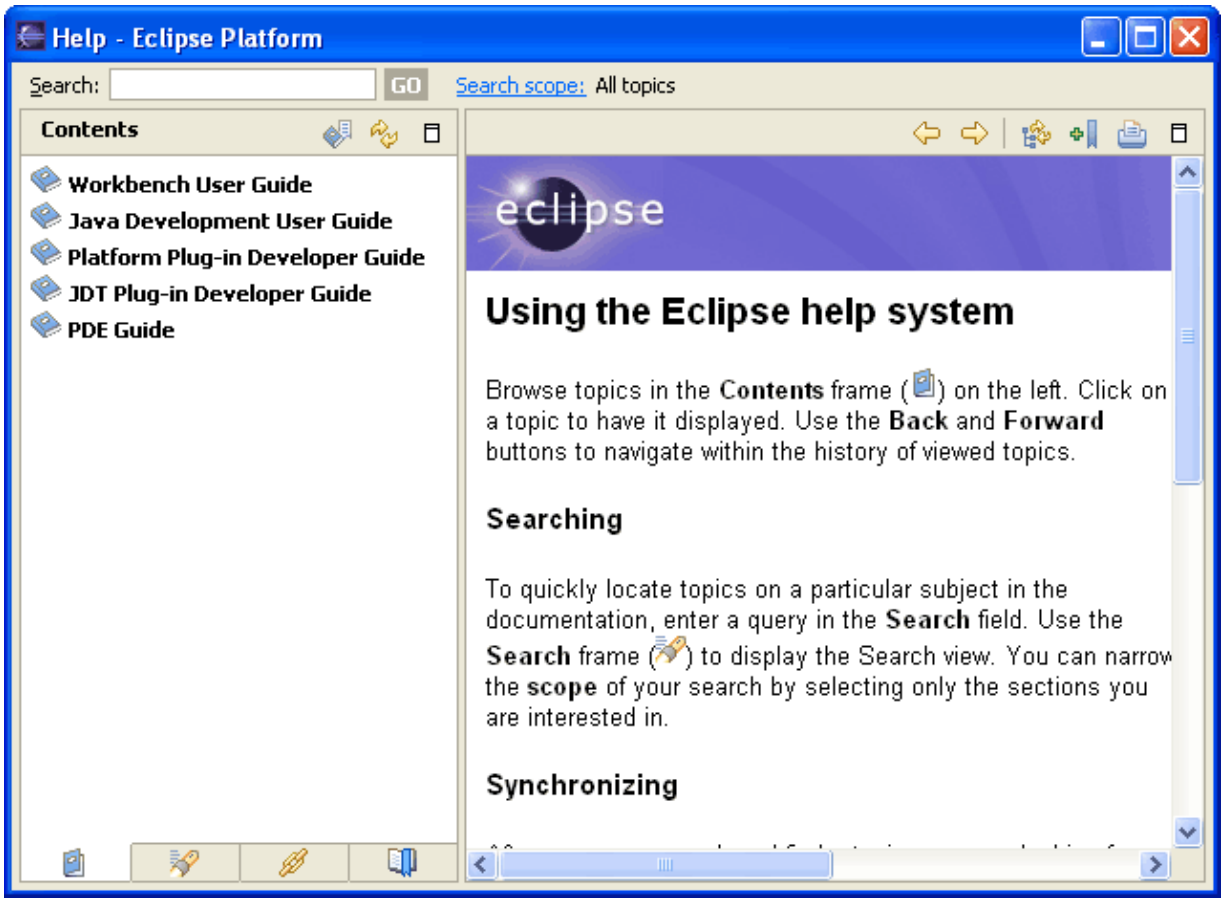


Table of Contents (Bookshelf)

This list contains all the available sets of information (or books). Click a book to see its topics tree.

Contents Tab

This tab displays an outline of the available Help topics for the selected book.

Search Results Tab

If you run a search of the documentation, the results list will be displayed on this tab.

Links Tab

If you invoke the context sensitive (F1) help, all the related links for that context are displayed in a list on this tab.

Bookmarks Tab

This list displays topics that were bookmarked using Bookmark Document button.

Search

Type a search query in the field and click Go. Click Advanced Search to enter a longer query or to filter the results by available books.

Search scope

Click the link to change the documentation set to be searched.

Go Back

Click this button to display the previously displayed page.

Go Forward

You can use this button if you have just used the Go Back button. Click it to display a previously displayed page.

Refresh / Show Current Topic and Show in Table of Contents

Click one of these buttons to match the navigation tree up to the topic currently shown in the content frame.

Bookmark Document

This button adds the currently displayed document to the Bookmarks view.

Print Page

This command prints the currently displayed topic.

Maximize and Restore

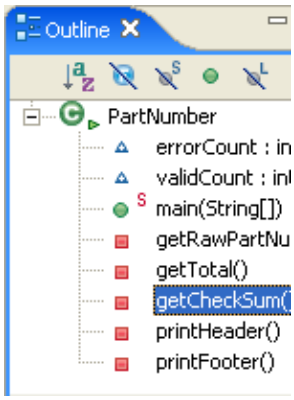
The Maximize buttons cause navigation or content frame to be maximized. When a frame is maximized the Restore button is available than can be used to restore the frame to its original size.

Related reference

Workbench User Guide

Outline view

The Outline view displays an outline of a structured file that is currently open in the editor area, and lists structural elements. The contents of the Outline view are editor specific. In the example below, which is for a Java source file, the structural elements are classes, fields, and methods. The contents of the toolbar are also editor specific.



Related concepts

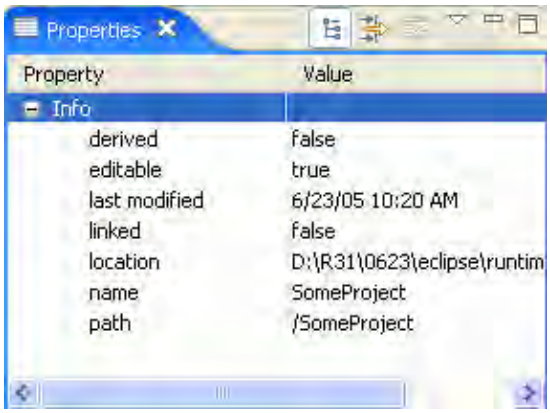
Views
Perspectives
Resources

Related tasks

Opening views
Moving and docking views
Creating fast views

Properties view

The properties view displays property names and values for a selected item such as a resource. Here is an example:



Toolbar buttons allow you to toggle to display properties by category or to filter advanced properties. Another toolbar button allows you to restore the selected property to its default value.

To see more detailed information about a resource than the Properties view gives you, right−click the resource name in one of the navigation views and select Properties from the pop−up menu.

Help system

The online help system lets you browse, search, and print system documentation. The documentation is organized into sets of information that are analogous to books. The help system also supplies a text search capabilities for finding the information you need by keyword and context−sensitive help for finding information to describe the particular function you are working with.

You can interact with help system in the workbench using Help view, or in external help browser window.

The help browser

When you first open the help browser by selecting **Help > Help Contents** in the Workbench, the first view shown in the help window is called Contents. Contents displays various groupings of documentation for the product. Click on one of the links to expand the navigation tree for that set of documentation. Browse the tree and click on topics to display them.

Search

If you want to find a particular piece of information in the online help, use Search. The help system will search the entire information set, or only a part of it, to find topics that meet the criteria you specify.

The Help view

You can open a Help view by The Help view by selecting **Help > Dynamic Help** or by selecting **Help > Search** in the Workbench., A help view will open turning to the Related Topics or to the Search page. You can use links at the bottom of the help view to turn to other pages. Browse topics and click on links to display them.

Context−sensitive help

If you are working through a task and encounter part of the interface that you do not understand, use context−sensitive help.. Bring focus to the interface widget in question by clicking on it or using the Tab key, and then press `F1` (`Ctrl+F1` on GTK+, and `Help` key on Carbon). A Help view will display description of the widget in focus, and list related topics. Leave the Help view open to see information on other interface elements when you focus on them.

When working in a dialog, like wizards, or preferences, and request context−sensitive help, a help view will be shown in a small window adjacent to the dialog.

An alternative way of presenting context−sensitive information are infopops. You can use help preference page to choose to use infopops instead of Help view for displaying context−sensitive help. If configured, a context help key, such as `F1`, will open an infopop that will provide you with information about the interface and, if appropriate, links to more information.

Note: On the some installation of KDE, Ctrl+F1 is the key binding for "Switch to Desktop 1". To make context−sensitive help work with GTK+ on KDE, then that binding must be removed. Try using the "KDE Default for 4 Modifier Keys" key binding scheme.

Related concepts

Help View

Related tasks

Accessing and navigating online help
Searching online help
Accessing context−sensitive help

Accessing and navigating online help

The help browser lets you browse, search, and print online documentation for the product. To open the help browser, select **Help > Help Contents** in the Workbench. This opens the help browser to the bookshelf, which shows the major sets of documentation available.

To navigate online help:

1. Select the desired link on the bookshelf.
2. Expand the topic tree to find the information you are looking for. To view a topic, click the link in the topic tree.
3. Most topics provide a list of links to related topics at the bottom. Follow these links to learn more.
4. Use the **Go Forward** and **Go Back** buttons. These behave the same way back and forward buttons work in an Internet browser, taking you to topics you have already looked at.
5. To synchronize the navigation frame with the current topic, click the **Refresh / Show Current Topic** button  or **Show in Table of Contents** button  . This is helpful if you have followed several links to related topics in several files, and want to see where the current topic fits into the navigation path.

As an alternative to browsing the information this way, use the search engine. Type a query into the **Search** field at the top of the browser and click **Go**.

To see context sensitive help from the Workbench put focus on a particular widget and press F1. For more details, refer to the list of related topics below.

To show documentation about capabilities that are disabled in the application, select the **Show All Topics** button  . When you choose to show all topics in the table of contents, the headings for documentation about any disabled activities are shown in the table of contents and also appear in search results.

Maximizing help views

To increase the space available for viewing the help content, the frame that displays content can be maximized. To maximize the frame, click the **Maximize** button  in the toolbar, or double click the toolbar. To return the frame to its original size, click the **Restore** button  in the toolbar or double click the tool bar again. Similarly, you can maximize the frame showing navigation, by double clicking its toolbar.

Printing online help

To print a topic from the online help:

1. Select the topic in the navigation frame.
2. Click the **Print** button  in the Help toolbar.
3. Select the desired printer settings, and click **Print**.

Related concepts

Online help system

Related tasks

Searching online help

Accessing context sensitive help

Searching online help

The help system includes a powerful text search engine that runs simple or complex queries on the documentation to help you find the information you are looking for.

To search the online help:

1. Select ***Help*** **>** ***Search***. type the term or terms for which you want to search.
2. Click **Go** or press Enter. The result set will be shown below.
3. To view the content of a topic in the result set, select it. Hits within the selected topic are highlighted.

Tip: You can also search local documentation from inside help browser by typing the term in the search field of the toolbar. The results will be shown in the Search view. Click a result to open the topic in the contents area.

Refining the search results in the help view

If the result set is very large, the information you are looking for might not appear in the top 10 or 15 results. You can then refine the search to reduce the number of results.

To refine a search:

1. Click the **Search Scope** link. to expand search scope section.
2. Click the **Advanced Settings** link. Search Scope preference dialog opens.
3. Click **Local Help** from the list.
4. Select **Search only the following topics** button to narrow down the search scope.
5. In the working set content tree, **select** the topics to which you want to narrow the search.
6. Click **OK**, to activate the changes and return to search page in the help view.
7. Click **Go** again. The results will be shown in the Search Results view in the Help browser.

Extending the search scope

If you cannot locate information in the local help, you can extend search scope to remote info−center or search engines.

To enable search engines:

1. Click the **Search Scope** link. to expand search scope section. The list of search engine is displayed.
2. Select the ones that contain information you are looking for.

In addition to search engines provided, you may define additional search engines.

To define a new search engine:

1. Click the **Search Scope** link. to expand search scope section.
2. Click the **Advanced Settings** link. Search Scope preference dialog opens.
3. Click **New**.
4. Select the search engine type.
5. Click ***OK***.

6. Provide a name and a description
7. Select engine specific settings and scope below. For the remote search engines, accessed using URL, fill in a full URL to query the engine. Use `{expression}` in the place of search expression.

Defining multiple search scopes

By default, changing search scope modifies the search scope named "default". You can define multiple search scope. They will be saved, allowing to quickly change search scope to one of them.

To define a new search scope:

1. Click the current search scope name, beside the **Search Scope** link. .Search Scope Sets dialog appears.
2. Click ***New***.
3. Type a name, and confirm.
4. Select the newly created search scope.
5. Click ***OK***. The new search scope becomes current.

Changes to the search scope affect current search scope.

Local search query syntax

Remember the following search expression rules:

- Unless otherwise stated, there is an implied AND between all search terms. In other words, topics that contain all the search terms will be returned. For example:

  ```
  Java project
  ```

  returns topics that contain the word *Java* and the word *project*, but does not return topics that contain only one of these words.
- Use OR before optional terms . For example:

  ```
  applet OR application
  ```

  returns topics that contain the word *applet* or the word *application* (or both).
- Use NOT before terms you want to exclude from search results. For example:

  ```
  servlet NOT ejb
  ```

  returns topics that contain the word *servlet* and do not contain the word *ejb*. **Note:** NOT only works as a binary operator (that is, "NOT servlet" is not a valid expression).
- Use ? for a single−character wildcard and * for a multi−character wildcard. For example:

  ```
  par?
  ```

  returns topics that contain *part* or *park*, but not *participate*. On the other hand:

  ```
  par*
  ```

returns topics that contain *part*, *park*, *participate*, *pardon*, and so on. **Note:** The search engine does not accept terms with a wild card at first character position.

- Use double quotation marks around terms you want treated as a phrase. For example:

  ```
  "creating projects"
  ```

  returns topics that contain the entire phrase *creating projects*, and not *creating* or *project* on its own.
- Punctuation acts as term delimiters. For example:

  ```
  plugin.xml
  ```

  returns hits on topics that contain *plugin.xml*, *plugin*, and *xml*, which is likely broader than you want. If you want to find just those topics containing *plugin.xml*, use double quotes, as in:

  ```
  "plugin.xml"
  ```
- The search engine ignores character case. For example:

  ```
  Workbench
  ```

  returns topics that contain 'workbench', 'Workbench', 'WorkBench', and 'WORKBENCH'.
- The following stop words are common English words which will be ignored (not searched for) if they appear in the search expression: a, and, are, as, at, be, but, by, in, into, is, it, no, not, of, on, or, s, such, t, that, the, their, then, there, these, they, to, was, will, with.
- The search engine does "fuzzy" searches and word stemming. If you enter *create*, it will return hits on topics that contain *creates*, *creating*, *creator*, and so on. To prevent search engine from stemming terms, enclose them in double quotes.

Search index generation

The first time you search the online help, the help system might initiate an index−generation process. This process builds the indexes for the search engine to use. It may take several minutes, depending on the amount of documentation and whether prebuilt indexes are installed. Results of the search will be available upon completion of the indexing process.

Each time you add or modify the documentation set (for example, when you install a new feature or update an existing one), the index will be updated to reflect the new information set.

Related tasks

Accessing and navigating online help

Accessing context−sensitive help

To access the context−sensitive help for a given widget:

1. Select the widget by putting focus on it (see the table below).
2. Press `F1` key. On GTK windowing system use `Ctrl+F1`, and on Carbon windowing system use `Help` key instead of `F1`.

This displays a description of the selected widget, and usually, a list of links to related information, in the help view. If you want more information than what is shown in the description, click a link to one of the related topic, or one of the search results appearing in the dynamic help section in the help view.

Depending on help preference settings, requesting context−sensitive help may display context help in an *infopop* instead of help view. You can dismiss the infopop by clicking outside it, or by pressing **Esc**. If you want more information than what is shown in the infopop, click a link to the related information. This opens the Help browser to the selected topic and closes the infopop. The list of related links stays in the left hand frame, so you do not have to press `F1` again if you are interested in other topics related to the same widget.

To access context sensitive help for a widget, you must put focus on it and then press `F1`. The following table shows how to put focus on some of the different kinds of widgets.

Widget type	How to select it
Menu item (for example ***Save*** in the ***File*** menu)	Let your mouse pointer rest over it so that it is highlighted, but don't click on it.
Field	Put the cursor in the field.
Button (for example **Cancel** button on a Dialog)	Tab until the button is in focus.
List	Click an item in the list.
View or pane	Click the title bar of the View or pane.
checkbox	Click the checkbox (which changes its state) or use the Tab key to bring it into focus.

Note: Context−sensitive help via `F1` is unavailable from toolbar buttons. Instead, let your mouse pointer hover over a toolbar button to view tooltip help for buttons.

Related tasks

Accessing and navigating online help

Watch/Edit

CVS provides a notification scheme which allows a group of developers to know if somebody is working on a given file. This facility is known as *watches*. By setting a *watch* on a file, you can have CVS notify you via email if someone else starts to *edit* this file. This mechanism is notification based only; the file is not locked in any way on the server, and several people are allowed to edit the same file at the same time.

In addition to watch list notification, *edit* on its own is useful for discovering if others are also editing that file. This is because when you edit a file, you will be informed if someone else is already editing it.

Normally with CVS clients, an explicit *edit* would need to be issued by the user. With Team CVS support however, an *edit* is automatically issued by the client when you start to modify a file. In addition, when *editing* a file, Team CVS provides you with the list of people already editing a file. This allows you to find out who is working on a file before you start to edit it.

- Related concepts

Team programming with CVS

- Related tasks

Finding out who's working on what: watch/edit

- Related reference

CVS

Finding out who's working on what: watch/edit

CVS provides a notification scheme which allows you to know if someone is modifying a file that you care about. This facility is known as *watches*. By setting a *watch* on a file, you can have CVS notify you via email (or other) if someone else starts to *edit* this file.

There are two parts to CVS watches: *watch*, and *edit*. The first, *watch*, is how you specify which files you wish to be notified about. The second, *edit*, is how you inform the CVS server (and thus others) that you are about to modify a file.

Edit is useful on its own without ever setting up any watches and lots of people work this way. This is because when you edit a file, you will be told immediately if someone else is already editing that file. Since most people just want to know up front that they may have to merge their changes on commit, *edit* on its own is sufficient for most. Another advantage to using just *edit* is that it doesn't require any administrative changes to the server, where as *watch* does. All that *watches* gives above this is the email notification that some file you are watching is being modified.

For these reasons, *edit* is supported natively by Team CVS where as *watch* isn't.

Setting up Watches

As mentioned, you can't set watches in Team CVS. If you are interested in doing this, you should consult your cvs documentation. In brief though, this is what's involved:

1. First, you or your CVS administrator will need to modify the CVSROOT/notify file. Consult the CVS documentation on watches for details on how to configure this file.
2. Next, you will need to perform a command line "cvs watch add <filename>" for each file that you wish to watch. If <filename> is a directory name, then all files within that directory will be watched.

Setting up a Project for Watch/Edit

Watches and editing are optional in CVS. To use this facility, you must turn on this option in the ***Team > CVS > Watch/Edit*** preferences page. Select "Configure projects to use Watch/Edit on checkout", accept the preference dialog, and then checkout your project. All the files in the project will be checked out read−only. This tells the CVS client which files are being edited by you and which aren't (writable files are being edited). If you've already checked out the project before you turned on this option, you can either check it out again or enable the "Use Watch/Edit for this project" option on the project's CVS properties page. Either of these operations will make the files in the project read−only.

Editing

Although typical CVS clients require you to perform an explicit edit, Team CVS automatically issues an edit as soon as you start to modify a file. This support is built deep into Eclipse, so typing in a text editor, performing Java refactoring, etc., will all issue a CVS edit for you. You can also perform an explicit edit via the ***Team > Edit*** context menu on a resource.

When an *edit* is issued, you will be informed immediately if someone is already editing that file. In addition, everyone who is *watching* that file will be notified by the CVS server via email etc. Since watches simply

give you email notification, *edit* without ever setting up watch lists is still a useful (and popular) workflow.

If you prefer, you can turn off automatic issuing of edits. This means you will need to manually perform a Team > Edit for each file you are working on. To use this work mode, turn on the ***Team > CVS > Watch/Edit*** preference "Edit the file without informing the server".

Finally, you can see the list of editors of a file at any time by selecting ***Team > Show Editors*** from the context menu of that file.

Unediting

Just as you can tell CVS that you are editing a file, there also needs to be a way of telling CVS that you are no longer editing that file. This is referred to as *unedit*. This way, if someone checks the editors list for a file, they'll know if someone is still working on that file. This happens in one of two ways:

- When you commit an edited file, an unedit is automatically issued.
- If you discover that you want to back out of the changes to a file, you can explicit unedit the file. In addition to notifying the server, an explicit unedit will revert the file to its base (i.e. the workspace will then contain the copy of the file before you started modifying it).

Related concepts

Watch/Edit

Team programming with CVS

Related reference

CVS

Accessibility features in Eclipse

Accessibility features help people with a physical disability, such as restricted mobility or limited vision, or those with special needs to use software products successfully. These are the major accessibility features in Eclipse:

- Eclipse uses Microsoft Active Accessibility (MSAA) APIs to render user interface elements accessible to assistive technology.
- You can operate all features using the keyboard instead of the mouse. See the related task.
- You can use screen−reader software such as Freedom Scientific's JAWS ™ and a digital speech synthesizer to hear what is displayed on the screen. You can also use voice recognition software, such as IBM ViaVoice ™ to enter data and to navigate the user interface.
- You can magnify what is displayed on your screen in the graphical views.
- Any fonts or colors defined by Eclipse can be set using the ***Window*** **>** ***Preferences*** dialog**.** See the related link.

Note: The Accessibility features mentioned in this document apply to the Windows operating system.

Related tasks

Navigating the user interface using the keyboard

Related reference

Keys
Font and color settings in Eclipse

Navigating the user interface using the keyboard

The user interface is navigable using the keyboard. The Tab key is used to iterate through the controls in a particular scope (for example, a dialog or a view and its related icons). To navigate to the main controls for the Workbench window or to tab out of views that use the Tab key (such as editors) use Ctrl+Tab.

Menus

Most menus are assigned mnemonics for each entry which allow you to select them by typing the underlined letter instead of the mouse. You can also select an item by moving through the menus and sub−menus with the arrow keys.

The various menus available can be accessed using the keyboard in the following ways:

- F10 accesses the menus on the main menu bar.
- Shift+F10 pops up the context menu for the current view. (Note: this shortcut is actually dependent on your window manager, but for most people it should be Shift+F10.)
- Ctrl+F10 will open the pull down menu for the current view if there is one. For editors, Ctrl+F10 will open the menu for the marker bar on the left of the editor area.
- Alt+mnemonic will activate the Workbench menu for a particular entry (e.g., Alt+W will bring down the Window menu).
- Microsoft Windows only: Pressing Alt will give focus to the menu bar.

Controls

Mnemonics are assigned to most control labels (e.g., buttons, checkboxes, radio buttons, etc.) in dialog boxes, preference pages, and property pages. To access the control associated with a label, use the Alt key along with the letter that is underlined in the label.

Navigation Context

Navigation context is saved for the packages, navigator views, Workbench preferences and properties dialogs. The expanded state of the tree in the packages and resource views are saved between Workbench sessions. The selected page for the preferences and properties dialog is saved between invocations of the dialog but are not saved between workbench invocations.

Cycling Editors, Views and Perspectives

To switch between editors, views and perspectives, the workbench provides a cycling function that is invoked by Ctrl and a function key. All of these cycling functions recall the last thing selected to allow for rapid cycling back and forth between two items. The cycling functions are

- Ctrl+F6 − Cycle to Editor
- Ctrl+F7 − Cycle to View
- Ctrl+F8 − Cycle to Perspective

Also, Ctrl+E can be used to activate the editor drop−down, and Ctrl+PageUp and Ctrl+PageDown can be used for switching between the open editors.

Accelerators

Many of the actions in Eclipse have an accelerator assigned to them. For additional information on accelerators, see Keys.

Help system

You can navigate the help system by keyboard using the following key combinations:

- Pressing Tab inside a frame (page) takes you to the next link, button or topic node.
- To expand/collapse a tree node, press Right/Left arrows.
- To move to the next topic node, press Down arrow or Tab
- To move to the previous topic node, press Up arrow or Shift+Tab
- To display selected topic, press Enter.
- To scroll all the way up or down press Home or End.
- To go back press Alt+Left arrow; to go forward press Alt+Right arrow.
- To go to the next frame, or toolbar press Ctrl+Tab (Ctrl+F6, if using Mozilla, or Mozilla based browser).
- To move to previous frame, press Shift+Ctrl+Tab. (Shift−Ctrl+F6, if using Mozilla, or Mozilla based browser).
- To move to the frame displaying topic content press Alt+K (when using embedded help browser on Windows, or Internet Explorer).
- To move to Contents tab, press Alt+C
- To move to Search Results tab, press Alt+R
- To move between tabs, press Right/Left arrows.
- To switch view, select a tab and press Enter.
- To switch and move to a view, select a tab and press Up arrow.
- To move to the search entry field, press Alt+S
- To print the current page or active frame, press Ctrl+P.
- To find a string in the current page or active frame, press Ctrl+F (when using embedded help browser on Windows, or Internet Explorer).

Most labels of controls on help system pop−up dialogs have mnemonics are assigned to them. To access the control associated with a label, use the Alt key along with the letter that is underlined.

Related concepts

Accessibility features in Eclipse
Changing the key bindings
Keys
Online help system

Related reference

Font and color settings in Eclipse

Keys

The function of the keyboard can be extensively customized in Eclipse. Within Eclipse, key strokes and key sequences are assigned to invoke particular commands.

Key Strokes, Key Sequences, and Key Bindings

A 'key stroke' is the pressing of a key on the keyboard, while optionally holding down one or more of these modifier keys: `Ctrl`, `Alt` (`Option` on the Macintosh), `Shift`, or `Command` (only on the Macintosh.) For example, holding down `Ctrl` then pressing `A` produces the key stroke `Ctrl+A`. The pressing of the modifier keys themselves do not constitute key strokes.

A 'key sequence' is one or more key strokes. Traditionally, Emacs assigned two or three key stroke key sequences to particular commands. For example, the normal key sequence assigned to `Close All` in emacs is `Ctrl+X Ctrl+C`. To enter this key sequence, one presses the key stroke `Ctrl+X` followed by the key stroke `Ctrl+C`. While Eclipse supports key sequences of arbitrary lengths, it is recommended that keyboard shortcuts be four key strokes in length (or less).

A 'key binding' is the assignment of a key sequence to a command.

Schemes

A 'scheme' is a set of bindings. Eclipse includes two schemes:

- Default
- Emacs (extends Default)

The *Default* scheme contains a general set of bindings, in many cases recognizable to users as traditional key sequences. For instance, `Ctrl+A` is assigned to `Select All`, and `Ctrl+S` is assigned to `Save`.

The *Emacs* scheme contains a set of key bindings familiar to users of Emacs. For instance, `Ctrl+X H` is assigned to `Select All`, and `Ctrl+X S` is assigned to `Save`.

It is important to understand why the *Emacs* scheme says that it 'extends Default'. The *Emacs* scheme is not a complete set of bindings like the *Default* scheme. Rather, it borrows from the *Default* scheme where possible, only defining explicit Emacs−style bindings where they vary from the *Default* scheme. Generally, only well known commands like `Select All`, `Save`, etc. have specific Emacs key sequences associated with them.

The user decides which scheme they are most comfortable using by changing the 'Scheme' setting on the keys preference page. If the user chooses the *Default* scheme, all *Emacs* bindings are ignored. If the user chooses the *Emacs* scheme, explicit Emacs−style key sequence assignments take precedence over any conflicting assignments in the *Default* scheme.

Contexts

Key bindings can vary based on the current context of Eclipse.

Sometimes the active part might be a Java file editor, for instance, where a different set of key sequence assignments may be more appropriate than if the active part was an html file editor. As a specific example, typically `Ctrl+B` is assigned to `Build` in a context such as Java file editing, while `Ctrl+B` is assigned to `Make Text Bold` in a context such as HTML file editing. This context is usually determined by the active part, but it can be influenced by the active window or dialog as well. If the active part does not choose a particular context, the workbench will set the active context to *In Windows*.

Eclipse includes nine different contexts. They are:

- In Dialogs and Windows
- In Windows (extends In Dialogs and Windows)
- In Dialogs (extends In Dialogs and Windows)
- Editing Text (extends In Windows)
- Editing Java Source (extends Editing Text)
- Debugging (extends In Windows)
- Debugging Java (extends Debugging)
- In Console
- Editing Ant buildfiles

Much like configurations, contexts can extend other contexts. For example, the *Editing Java Source* context borrows key bindings from the *Editing Text* context, which in turn borrows key bindings from the *In Windows* context.

Note: It is not recommended to promote a key binding to a context which it extends. For example, it is not recommended to move an *Editing Text* key binding to the *In Dialogs and Windows* context. This may have unexpected results.

It is possible for some key bindings to work in dialogs. Those key bindings are assigned to the *In Dialogs and Windows* context. One example of such a key binding is the key binding for "cut". It is possible to change these key bindings. For example, it is possible to have Ctrl+X as cut in dialogs, but Ctrl+W as cut in windows.

Platform and Locale

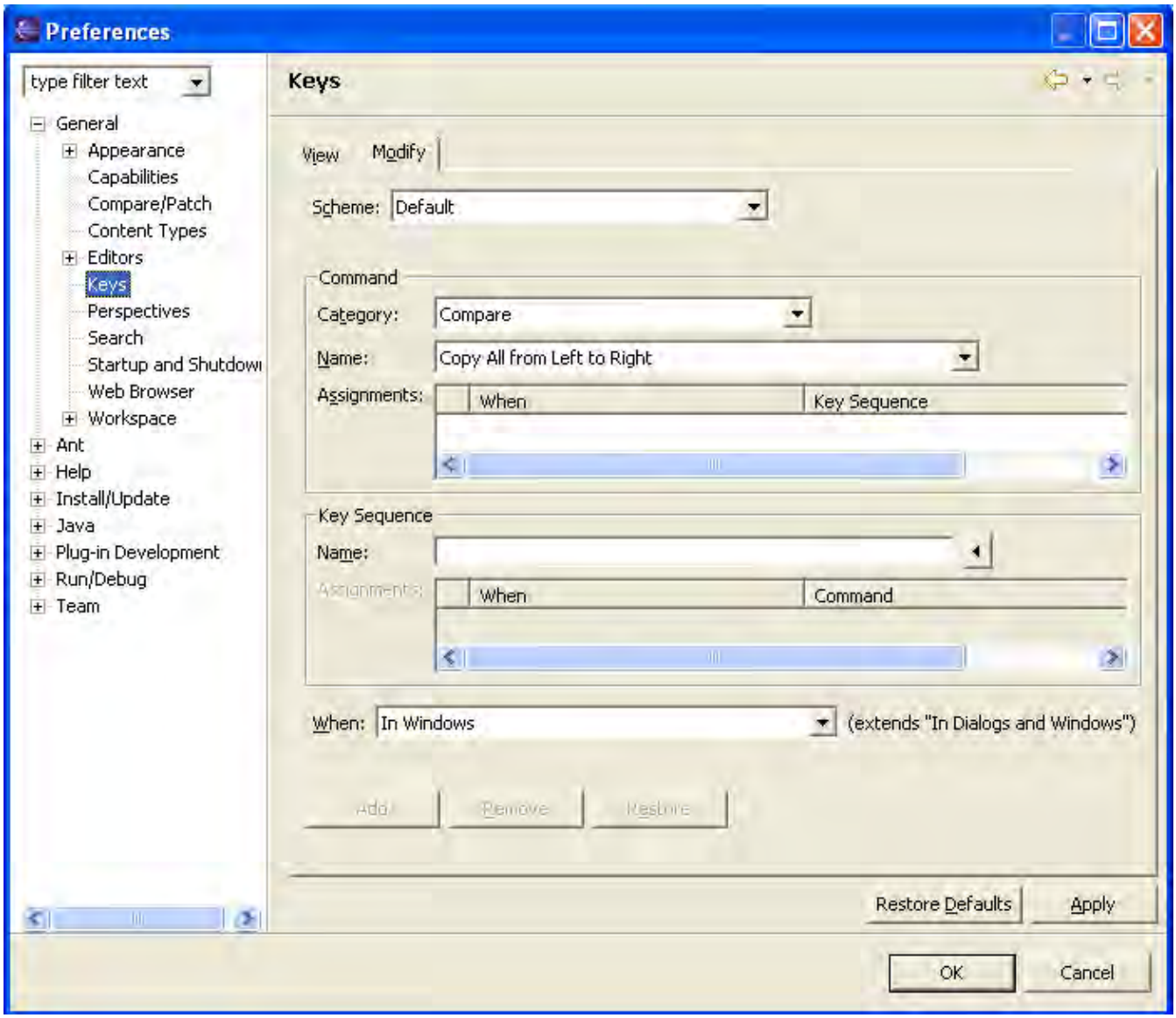
Key bindings also vary by platform and locale. On the Macintosh platform, `Command+S` is assigned to `Save`, instead of the usual `Ctrl+S`. On Chinese locales (zh), `Alt+/` is assigned to `Content Assist`, instead of the usual `Ctrl+Space`.

The current platform and locale is determined when Eclipse starts, and does not vary over the course of an Eclipse instance.

Customizing Key bindings

With multi−stroke key sequences, schemes, and contexts, there are a lot of things to keep in mind when customizing key bindings. To make things easier, all key customization is done on the Keys preference page.

Select **Window** > **Preferences** > **General** > **Keys** for the Keys preference page.



In this example, we have chosen to select *Emacs* as the scheme, and have selected the command `Close` from the list of commands. Information on this command, along with its current key bindings, is shown.

Notice that `Close` has three key sequences assigned to it: `Ctrl+F4` and `Ctrl+W` in the *Default* scheme and `Ctrl+X K` in the *Emacs* scheme. Both are assigned in the *In Windows* context. Thus, if the user sets the scheme to *Default*, `Ctrl+F4` and `Ctrl+W` will be assigned to `Close` while `Ctrl+X K` will not. If the user sets the scheme to *Emacs*, however, `Ctrl+X K` will be assigned to `Close`. As well, because the *Emacs* scheme also borrows key bindings from the *Default* scheme, `Ctrl+F4` and `Ctrl+W` will also be assigned to `Close` provided that these key binding have not been assigned to another command in the *Emacs* scheme. In this example, "Ctrl+W" is bound to `Cut` in the Emacs scheme.

Below the list of key sequences assigned to `Close`, there is a place to add or remove key bindings. By default, it selects the context as *In Windows*.

We type in the key sequence `Ctrl+W`, and the 'Add' button becomes enabled. Also, a list of all the commands to which the key sequence `Ctrl+W` is already assigned is show below the Add button. We can see that `Ctrl+W` is currently assigned to the command `Cut` in the *In Windows and Dialogs* context. We click the

'Add' button to assign `Ctrl+W` to `Close`.

The Dynamic Nature of Key bindings

Key bindings are provided by plug−ins, and in Eclipse, plug−ins can be added or removed. This can cause key bindings declared by these plug−ins to be added or removed. Eclipse stores custom key bindings in a way to This compensate for this. Consider the example above where `Ctrl+Alt+W` was assigned to `Cut` in the *Emacs* scheme. Consider the user installs a new plug−in which assigns `Ctrl+Alt+W` to a particular command. Eclipse will preserve the user's assignment to `Cut`.

Conflict Resolution

There are only a finite number of simple, common key strokes available to assign to a multitude of commands. We have seen that scheme, context, platform, and locale all partition key sequence assignments into domains where they don't conflict with one another. Consider the case for `Ctrl+B` above if contexts did not exist. One plug−in would assign `Ctrl+B` to `Build`, the other plug−in would assign `Ctrl+B` to `Make Bold Text`. How would Eclipse properly resolve this conflict?

Though conflicts are drastically reduced by employing the above mechanisms, they can still occur. Two plug−ins, independent of one another, could assign the same key sequence to different commands with the same context, scheme, platform, and locale. Consider if a plug−in assigned `Ctrl+F4` in the *In Windows* context and *Default* scheme to one of its commands. This directly conflicts with Eclipse assigning `Ctrl+F4` to the close command in the same context and scheme.

This is a conflict. It wouldn't be proper to invoke both commands, nor would it be proper to simply choose one of the two commands to receive the key stroke. The only proper thing to do is to ignore both key bindings, making `Ctrl+F4` effectively useless in this context and scheme.

These types of conflicts can be resolved by the user explicitly assigning the key sequence to one of the commands.

Another type of conflict can be caused by multiple−key stroke key sequences. For example, in the *Emacs* scheme, there are many multiple−key stroke key sequences beginning with the key stroke `Ctrl+X`. `Ctrl+X K` is assigned to `Close`. `Ctrl+X H` is assigned to `Select All`.

As previously mentioned, the *Emacs* scheme borrows key bindings from the *Default* scheme. In the default scheme, `Ctrl+X` is assigned to `Cut`. Though the *Emacs* scheme doesn't explicitly redefine `Ctrl+X`, pressing `Ctrl+X` is required as part of many of its key bindings. In the *Emacs* scheme, when one presses `Ctrl+X`, one is half way to entering one of many possible assigned key sequences. One would not expect the `Cut` action to be invoked at this time.

For this type of conflict, the rule is that the `Ctrl+X` key sequence assigned to `Cut` would be ignored. Otherwise, it would not be possible to complete many of the key bindings in the *Emacs* configuration.

Related concepts

Accessibility Features in Eclipse
Changing the key bindings
Online help system

Related reference

Font and color settings in Eclipse

Fonts and colors in Eclipse

Eclipse uses the fonts and colors provided by the operating system as much as possible. On Windows the platform color and font settings are found on the **Properties > Colors and Fonts** page. The font used by most widgets in Eclipse is the one set in the Message Box settings of the properties. However, operating systems do not provide enough colors to handle all of the extra information that colors and fonts provide in Eclipse.

Fonts

There are 4 main fonts in use by the Eclipse platform. They are:

Banner Font
: Used in PDE editors, welcome pages and in the title area of many wizards. For instance the New Project wizard uses this font for the top title,

Header Font
: Used as a section heading. For instance the Welcome page for the Eclipse Platform uses this font for the top title,

Text Font
: Used in text editors.

Dialog Font
: Used in dialogs.

These fonts can be set via the **General > Appearance > Colors and Fonts** preference. As well as these 4 fonts there are several other secondary font settings. These default to the text font. They can be found on the Colors and Fonts preference page:

- **Compare Text Font**
- **Console Text Font**
- **CVS Console Font**
- **Debug Console Font**
- **Detail Pane Text Font**
- **Java Compare Text Font**
- **Java Editor Text Font**
- **Memory Views Table Font**
- **Part Title Font** (optional: used by some presentations)
- **View Message Font** (optional: used by some presentations)

Colors

Eclipse uses colors as an information enhancement in many places. Whenever possible the operating system color settings are used, but in cases where the operating system settings are not enough, Eclipse defines other colors. All of these colors can be adjusted via the following preference pages:

- **General > Appearance > Colors and Fonts > Basic** (Error text, hyperlink text, active hyperlink text)
- **General > Search** (Foreground for potential matches)
- **Run/Debug > Console** (Standard Out, Standard Error, Standard In)
- **Run/Debug** (Variable Views changed value, Memory View unbuffered lines)
- **Ant** (Error, Warning, Information, Verbose, Debug)

- **Java > Editor** (Line number, matching brackets, current line, print, find scope, hyperlink, selection foreground, selection background)
- **Java > Editor,** select the **Syntax** Tab (Javadoc HTML tags, Javadoc keywords, Javadoc links, Javadoc others, keyword 'return', keywords excluding 'return', Method names, Multi line comment, Operators and brackets, Others, Single−line comment, Strings, Task Tags)
- **Java > Editor > Code Assist** (completion proposal background, completion proposal foreground, method parameter background, method parameter foreground, completion overwrite background, completion overwrite foreground)
- **Plug−in Development > Editors** (Text, Processing instructions, Constant strings, Tags, Comments)
- **Team > CVS > Console** (Command line, Message, Error)

WIN Accessibility and the Windows Color Dialog

For color selection, Eclipse uses a dialog provided by the operating system. On windows, the color selection dialog does not respond properly to assistive technology. When you first get into the dialog, focus is on one of the basic colors, but the dialog provides no indication of this through assistive technology. You can select colors in Eclipse with this dialog in the following way:

1. Select to customize the color of something in Eclipse, for example the color of Error Text in your Workbench Colors and Fonts Basic preferences.
2. In the color selection dialog, tab twice to go from the Basic Color matrix to the Define Custom Colors button and press Enter.
3. You can now enter the basic colors using an HSL or RGB specification according to the following definitions. See the Windows Color Dialog Reference for a tables and values for these colors.

Related tasks

Accessibility Features in Eclipse
Navigating the user interface by using the keyboard

Related reference

Keys
Windows Color Dialog Reference

Windows Color Dialog Reference

Below are the tables that show the colors of the fonts in the matrix of the Windows color dialog and the color settings that each of these correspond to.

Windows Color Dialog Color Matrix

salmon	pale yellow	pale green	spring green	pale turquoise	deep sky blue	pale rose	pink
red	yellow	apple green	light green	aqua	turquoise	pale slate blue	magenta
chocolate	pumpkin	lime	teal	dark turquoise	medium slate blue	maroon	rose
dark red	dark orange	green	sea green	blue	medium blue	purple	blueviolet
dark brown	saddle brown	dark forest green	dark teal	navy	midnight blue	dark purple	dark blueviolet
black	olive	dark olive	grey	light teal	light grey	dark purple	white

Settings for Default Colors in the Windows Color Dialog

Color	***Hue***	***Saturation***	***Lumination***	***Red***	***Green***	***Blue***
apple green	60	240	120	128	255	0
aqua	120	240	120	0	255	255
black	160	0	0	0	0	0
blue	160	240	120	0	0	255
blueviolet	180	240	120	128	0	255
chocolate	0	80	90	128	64	64
dark blueviolet	180	240	60	64	0	128
dark brown	0	240	30	64	0	0
dark forest green	80	240	30	0	64	0
dark olive	40	80	90	128	128	64
dark orange	20	240	120	255	128	0
dark purple	200	240	30	64	0	64
dark purple	200	240	30	64	0	64
dark red	0	240	60	128	0	0
dark teal	120	240	30	0	64	64
dark turquoise	140	240	60	0	64	128
deep sky blue	140	240	120	0	128	255
green	80	240	60	0	128	0
grey	160	0	120	128	128	128
light green	90	240	120	0	255	64
light grey	160	0	181	192	192	192

light teal	120	80	90	64	128	128
lime	80	240	120	0	255	0
magenta	200	240	120	255	0	255
maroon	220	240	60	128	0	64
medium blue	160	240	75	0	0	160
medium slate blue	160	240	180	128	128	255
midnight blue	160	240	30	0	0	64
navy	160	240	60	0	0	128
olive	40	240	60	128	128	0
pale green	80	240	180	128	255	128
pale rose	220	240	180	255	128	192
pale slate blue	160	81	151	128	128	192
pale turquoise	120	240	180	128	255	255
pale yellow	40	240	180	255	255	128
pink	200	240	180	255	128	255
pumpkin	13	240	150	255	128	64
purple	200	240	60	128	0	128
red	0	240	120	255	0	0
rose	220	240	120	255	0	128
saddle brown	20	240	60	128	64	0
salmon	0	240	180	255	128	128
sea green	100	240	60	0	128	64
spring green	100	240	120	0	255	128
teal	120	240	60	0	128	128
turquoise	133	240	90	0	128	192
white	160	0	240	255	255	255
yellow	40	240	120	255	255	0

Running Eclipse

After you install (unzip) the Eclipse driver in a directory (such as c:\eclipse), start the Workbench by running the Eclipse executable file found in the top level install directory. The executable file is called `eclipse.exe` on Windows systems and `eclipse` on Linux systems. **Note:** the following discussion describes setting up on Windows systems. Setup on Linux is analogous.

If you do not specify otherwise, Eclipse will prompt you for a workspace directory. The default location for this directory will be a child of your user home directory called "workspace" (for example, c:\Documents and Settings\My Name\workspace). This workspace directory is used as the default content area for your projects as well as for holding any required metadata. For shared or multi−workspace installs you should explicitly state the location of your workspace rather than using the default. In addition to simply entering a different location at the prompt you may also use the −data command line argument.

Setting a specific location for the workspace with −data

To use the `−data` command line argument, simply add `−data` *your_workspace_location* (for example, `−data c:\users\robert\myworkspace`) to the **Target** field in the shortcut properties, or include it explicitly on your command line.

Setting the Java VM using −vm

It is recommended that you explicitly specify which Java VM to use when running Eclipse. This is achieved with the `−vm` command line argument (for example, `−vm c:\jre\bin\javaw.exe`). If you don't use `−vm`, Eclipse will use the first Java VM found on the O/S path. When you install other products, they may change your path, resulting in a different Java VM being used when you next launch Eclipse.

Advanced Topics in Running Eclipse

The Eclipse executable and the platform itself offer a number of execution options of interest to people developing or debugging parts of Eclipse. This is a list of the commonly used options, for a full list see the Eclipse runtime options page in the Platform Plug−in Developer Guide. The general form of running the Eclipse executable is:

```
eclipse [platform options] [−vmargs [Java VM arguments]]
```

Eclipse Startup Parameters

Command	Description	Since
−arch *architecture*	Defines the processor architecture on which the Eclipse platform is running. The Eclipse platform ordinarily computes the optimal setting using the prevailing value of Java `os.arch` property. If specified here, this is the value that the Eclipse platform uses. The value specified here is available to plug−ins as BootLoader.getOSArch(). Example values: "x86", "sparc", "PA−RISC", "ppc".	2.0
−application *applicationId*	The application to run. Applications are declared by plug−ins supplying extensions to the org.eclipse.core.runtime.applications extension point. This argument is typically not needed. If specified, the value overrides the value supplied by the configuration. If not specified, the Eclipse Workbench is run.	1.0
−configuration *configurationFileURL*	The location for the Eclipse Platform configuration file, expressed as a URL. The configuration file determines the location of the Eclipse platform, the set of available plug−ins, and the primary feature. Note that relative URLs are not allowed. The configuration file is written to this location when the Eclipse platform is installed or updated.	2.0
−consolelog	Mirrors the Eclipse platform's error log to the console used to run Eclipse. Handy when combined with `−debug`.	1.0
−data *workspacePath*	The path of the workspace on which to run the Eclipse platform. The workspace location is also the default location for projects. Relative paths are interpreted relative to the directory that Eclipse was started from.	1.0

-debug [*optionsFile*]	Puts the platform in debug mode and loads the debug options from the file at the given location, if specified. This file indicates which debug points are available for a plug−in and whether or not they are enabled. If a file location is not given, the platform looks in the directory that eclipse was started from for a file called ".options". Both URLs and file system paths are allowed as file locations.	1.0
-dev [*classpathEntries*]	Puts the platform in development mode. The optional classpath entries (a comma separated list) are added to the runtime classpath of each plug−in. For example, when the workspace contains plug−ins being developed, specifying `-dev bin` adds a classpath entry for each plug−in project's directory named `bin`, allowing freshly generated class files to be found there. Redundant or non−existent classpath entries are eliminated.	1.0
-keyring *keyringFilePath*	The location of the authorization database (or "key ring" file) on disk. This argument must be used in conjunction with the `-password` option. Relative paths are interpreted relative to the directory that Eclipse was started from.	1.0
-nl *locale*	Defines the name of the locale on which the Eclipse platform is running. The Eclipse platform ordinarily computes the optimal setting automatically. If specified here, this is the value that the Eclipse platform uses. The value specified here is available to plug−ins as BootLoader.getNL(). Example values: "en_US" and "fr_FR_EURO".	2.0
-nosplash	Runs the platform without putting up the splash screen.	1.0
-os *operatingSystem*	Defines the operating system on which the Eclipse platform is running. The Eclipse platform ordinarily computes the optimal setting using the prevailing value of Java `os.name` property. If specified here, this is the value that the Eclipse platform uses. The value specified here is available to plug−ins as BootLoader.getOS(), and used to resolve occurrences of the `$os$` variable in paths mentioned in the plug−in manifest file. Example values: "win32", "linux", "hpux", "solaris", "aix".	1.0
-password *password*	The password for the authorization database. Used in conjunction with the `-keyring` option.	1.0

−perspective *perspectiveId*	The perspective to open in the active workbench window on startup. If this parameter is not specified, the perspective that was active on shutdown will be opened.	1.0
−plugincustomization *propertiesFile*	The location of a properties file containing default settings for plug−in preferences. These default settings override default settings specified in the primary feature. Relative paths are interpreted relative to the directory that eclipse was started from.	2.0
−product *productId*	The ID of the product to run. The product gives the launched instance of Eclipse its personality, and determines the product customization information used. This replaces −feature, which is still supported for compatibility.	3.0
−refresh	Option for performing a global refresh of the workspace on startup. This will reconcile any changes that were made in the file system since the platform was last run.	1.0
−showlocation	Option for displaying the location of the workspace in the window title bar. In release 2.0 this option only worked in conjunction with the −data command line argument.	2.0
−vm *vmPath*	The location of Java Runtime Environment (JRE) to use to run the Eclipse platform. If not specified, the JRE is at `jre`, sibling of the Eclipse executable. Relative paths are interpreted relative to the directory that eclipse was started from.	1.0
−vm*args* *args*	When passed to the Eclipse, this option is used to customize the operation of the Java VM used to run Eclipse. If specified, this option must come at the end of the command line. The given arguments are dependant on VM that is being run.	1.0

All arguments following (but not including) the −vmargs entry are passed directly through to the indicated Java VM as virtual machine arguments (that is, before the class to run). **Note:** If an Eclipse startup argument, such as −data, is provided after the Java vm arguments (−vmargs), Eclipse will not start and you will receive a "JVM terminated. Exit code=1" error.

Running on Different VMs

Running Eclipse on J9

When running Eclipse on J9 version 1.5, it is recommended that you use the following VM options:

```
eclipse.exe [eclipse arguments] −vm path_to_j9w.exe
            −vmargs −ms:32 −mm:2048 −mo:32768 −moi:32768 −mca:32 −mco:128 −mx:2000000
```

When running Eclipse on J9 version 2.0, the default arguments chosen by J9W should be suitable. However, to override the parameters which are automatically set internally by the Eclipse executable, you must specify −vmargs with no following arguments as follows:

```
eclipse.exe [eclipse arguments] −vm path_to_j9w.exe −vmargs
```

Please refer to the J9 VM documentation and help for further information.

Running Eclipse on the IBM Developer Kit, Java(TM) Technology Edition VM

The default VM settings for IBM Developer Kit, Java(TM) Technology Edition 1.3 Linux work well for initial exploration, but are not sufficient for large scale development. For large scale development you should modify your VM arguments to make more heap available. For example, the following setting will allow the Java heap to grow to 256MB:

```
−vmargs −Xmx256M
```

Upgrading Eclipse

If you are upgrading to a newer release of Eclipse from an older release, there are simple steps to follow to migrate your workspace to the new release. Your workspace is the directory on disk that contains all of your project files, as well as meta−data such as preferences you may have customized. The steps to follow for upgrading depend on whether or not you used the "−data" command line argument when starting Eclipse. The "−data" argument is recommended because it clearly specifies the location of your workspace.

Tip: It doesn't hurt to make a backup of your workspace before upgrading. After you've upgraded your workspace, you won't be able to use it again with an older version of Eclipse. If you ever want to go "back in time" to an earlier release, you'll need that backup!

The workspace chooser dialog allows the user choose the location of the workspace and is shown to the user on first startup in the absence of a −data argument. The default location provided by this dialog will be a "workspace" child of the user's home directory (for example, C:\Documents and Settings\UserName\workspace.)

New users with 3.1 should just leave this default, or specify some other location, but it's not recommended that they place it under the eclipse install directory. They should not copy the workspace, as the workspace can contain metadata with absolute file system paths, which will be invalid if the workspace is copied elsewhere.

In Eclipse 3.0 and earlier, Eclipse's default location for the workspace was under the Eclipse directory. For example, if the eclipse.exe was in D:\eclipse−SDK−3.0.1\eclipse, then the default workspace location would be D:\eclipse−SDK−3.0.1\eclipse\workspace.

To load a workspace created using 3.0 or earlier, they should just point to the old workspace directory using the workspace chooser, whether it's under the old install directory or elsewhere.

Users who use "−data"

If you were previously using the "−data" argument to start Eclipse, your upgrade path is much easier:

1. Install the new version of Eclipse in a new location, separate from any old version of Eclipse.
2. Start this new version of Eclipse, using the "−data" command line argument to point to your old workspace location.

See the example in the previous section for an illustration.

Adding third party plug−ins

If you have installed extra plug−ins in your Eclipse environment, you will need to add these new plug−ins to each new build or version of Eclipse you install. Before you do this, refer to the documentation for those plug−ins to ensure they are compatible with the version of Eclipse you are moving to. There are several ways to add these extra plug−ins to your new Eclipse install:

1. Copy the directories for each plug−in into the "plugins" directory of your new Eclipse version.

2. Use an Eclipse update site to re−install those extra plug−ins or features in the new version of Eclipse.
3. If you are using product extensions, simply copy the "links" directory into the new Eclipse version install directory. For more details on product extensions, see the documentation in the Platform Plug−in Developer Guide, under **Programmer's Guide > Packaging and delivering Eclipse based products > Product extensions**.

Working with perspectives

Perspectives define the initial set and layout of views in the Workbench window. They provide a set of functionality aimed at accomplishing a specific type of task or working with specific types of resources.

See the Related tasks links for more details.

Related concepts

Perspectives
Views
Fast views

Related tasks

Switching between perspectives
Specifying the default perspective
Opening perspectives
Changing where perspectives open
Configuring perspectives
Saving a user defined perspective
Deleting a user defined perspective
Resetting perspectives

Working with views and editors

Views and editors are the main visual entities which appear in the Workbench. In any given perspective there is a single editor area, which can contain multiple editors, and a number of surrounding views which provide context.

The Workbench provides a number of operations for working with views and editors. See the Related tasks links for more details.

Related concepts

Views
Editors
Fast views
Perspectives

Related tasks

Opening views
Moving and docking views
Rearranging tabbed views
Creating fast views
Opening files for editing
Associating editors with file types
Editing files outside the Workbench
Tiling editors
Maximizing a view or editor

Rearranging tabbed views

In addition to dragging and dropping (docking) views inside the Workbench, you can rearrange the order of views within a tabbed notebook.

1. Click on the tab of the view that you want to move and drag it to where you want it. A stack symbol  appears as you drag the view across other view tabs.
2. Release the mouse button when you have the view tab in the desired location. The view that you selected is now moved.

Related concepts

Views

Related tasks

Moving and docking views

Customizing the Workbench

Many aspects of the appearance and behavior of the Workbench can be customized to suit your individual needs. For example, you can:

- Rearrange where items appear in the main toolbar.
- Change the key bindings used by editors.
- Change the fonts and colors which are used.

See the Related tasks links for more details.

Related concepts

Views
Editors
Workbench

Related tasks

Rearranging the main toolbar
Changing the key bindings
Controlling single and double click behavior
Changing fonts and colors
Changing the placement of the tabs
Importing and Exporting Preferences

Changing the key bindings

The function of the keyboard can be extensively customized in Eclipse.

Select **Window > Preferences > General > Keys** for the Keys preference page, where you can assign key sequences to many of the commands in Eclipse.

Related concepts

Keys
Navigating the user interface by using the keyboard

Controlling single and double click behavior

You can control how the Workbench responds to single and double clicks. To do this:

1. On the main menu bar, click **Window > Preferences**.
2. Select the **General** category.
3. Select the behavior you want to change from the **Open mode** group.
4. Click **OK**.

The effect of these selections varies by view. For example, in one of the navigation views:

Double click
: Will cause a single click on a resource to select it, and a double click to open it in an editor.

Single click
: Will cause a single click on a resource to both select it and immediately open an editor on it.

The checkboxes under the **Single click** radio button further refine the single click behavior. Checking **Select on hover** will cause the resource to be selected if you hover over it with the mouse. Checking **Open when using arrow keys** will cause the resource to be opened if you use the arrow keys to navigate to it.

Related reference

Workbench

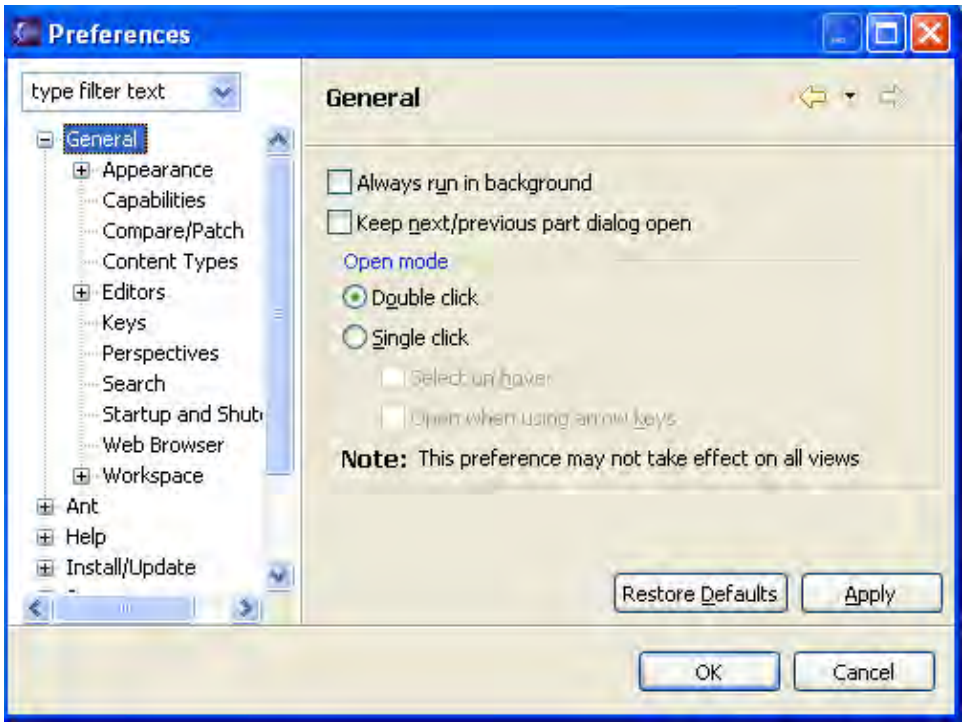
General

The term *Workbench* refers to the desktop development environment.
Each Workbench window contains one or more perspectives. Perspectives contain views and editors and control what appears in certain menus and tool bars. More than one Workbench window can exist on the desktop at any given time.

The following preferences can be changed on the General page.

Option	***Description***	***Default***
Always run in background	If this option is turned on, then the Workbench will perform certain actions in the background without disturbing the user.	Off
Keep next/previous part dialog open	If this option is turned on then the editor and view cycle dialogs will remain open when their activation key is let go. Normally the dialog closes as soon as the key combination is release.	Off
Open mode...	You can select one of the following methods for opening resources: • Double click − Single clicking on a resource will select it and double clicking on it will open it in an editor. • Single click (Select on hover) − Hovering the mouse cursor over the resource will select it and clicking on it once will open it in an editor. • Single click (Open when using arrow keys) − Selecting a resource with the arrow keys will open it in an editor. ***Note:*** Depending on which view has focus, selecting and opening a resource may have different behavior.	Double click

Here is what the General preferences page looks like:



Related reference

Workbench Window Layout

Workbench window layout

You can rearrange the layout of Workbench windows as follows:

- Drag views to different positions within the Workbench window.
- Drag views to the shortcut bar to create a fast view.
- Drag editors such that they are simultaneously visible beside, above, or below another editor.
- Resize views and editors by dragging the sashes which separate them.

Drop cursors

Drop cursors indicate where a view will dock when you release your mouse button. This indication is relative to the view or editor area underneath the cursor.

Drop Cursors

Cursor	Name	Description
	Dock above	The view will appear above the view underneath the cursor.
	Dock below	The view will appear below the view underneath the cursor.
	Dock to the right	The view will appear to the right of the view underneath the cursor.
	Dock to the left	The view will appear to the left of the view underneath the cursor.
	Stack	The view will appear as a tab in the same pane as the view underneath the cursor.

Fast views

From a view's title bar context menu, you can select Fast View to minimize the view as a button on the shortcut bar.

If a view is minimized in this way, you can click its button on the shortcut bar to bring it up in a fast view. To revert the fast view back to a docked view again, select the *Fast View* button on its title bar.

See Shortcut Bar

Double−Click

Double−clicking a view or editor's title bar maximizes the part in the Workbench window.

Title bar context menu and fast view toolbars

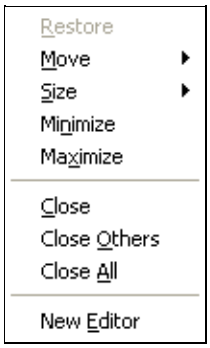
From the context menu of a view or editor's title bar, you can select how you want the view to appear within the Workbench window.

View and editor title bar context menu options

Option	Description
Restore	Restores the view to its originating (non−maximized/non−minimized) size and position within the Workbench.
Fast View (views only)	Minimizes the view and places a button for it in the shortcut bar. This option, when used in a fast view, will restore the view to its originating (docked) position in the Workbench. See Shortcut Bar
Restore	Restores the view or editor to its originating (non−maximized/non−minimized) size and position within the Workbench.
Move	Move the part or part group.
Size	Change the size of the part in the direction specified.
Maximize	Maximizes the part in the Workbench window.
Minimize	Minimizes the part in the Workbench window.
Close	Closes the part.
Close Others (editors only)	Closes all editors except the current editor.
Close All (editors only)	Closes all editors.

Here is what a view's context menu looks like:

Here is what an editor's context menu looks like:


Restore
Move
Size
Minimize
Maximize
Close
Close Others
Close All
New Editor

Changing fonts and colors

By default, the Workbench uses the fonts and colors provided by the operating system. However, there are a number of ways that this behavior can be customized.

Fonts

The Workbench lets you directly configure the following fonts:

Banner Font
: Used in PDE editors, welcome pages and in the title area of many wizards. For instance the New Project wizard uses this font for the top title.

Dialogue Font
: Used for widgets in dialogs.

Header Font
: Used as a section heading. For instance the Welcome page for the Eclipse Platform uses this font for the top title.

Text Font
: Used in text editors.

CVS Console Font
: Used in the CVS console.

Ignored Resource Font
: Used to display resources that are ignored from CVS.

Outgoing Change Font
: Used to display outgoing changes in CVS.

Console Font (defaults to text font)
: Used by the debug console.

Detail Pane Text Font (defaults to text font)
: Used in the detail panes of debug views.

Memory View Table Font (defaults to text font)
: Used in the table of the memory view.

Java Editor Text Font (defaults to text font)
: Used by Java editors.

Properties File Editor Text Font (defaults to text font)
: Used by Properties File editors.

Compare Text Font (defaults to text font)
: Used by textual compare/merge tools.

Java Compare Text Font (defaults to text font)
: Used by Java compare/merge tools.

Java Properties File Compare Text Font (defaults to properties file editor text font)
: Used by Java properties file compare/merge tools.

Part Title Font (defaults to properties file editor text font)
: Used for view and editor titles. ***Note:*** It is recommended that this font not be bold or italic because the workbench will use bold and italic versions of this font to display progress.

View Message Font (defaults to properties file editor text font)
: Used for messages in the view title bar (if present).

To change these fonts:

1. On the main menu bar, click **Window > Preferences**.
2. Expand the **General > Appearance** category and select **Colors and Fonts**.
3. Select the font you want to change.
4. Click **Change**.
5. Use the dialog which opens to select a font.
6. Click **OK**.

Note: You can also click **Use System Font** to set the font to a reasonable value chosen by the operating system. For example, on Windows this will use the font selected in the Display Properties control panel.

Plug−ins that use other fonts may also provide preference entries to allow them to be customized. For example, the Java Development Tools provide a setting for controlling the font used by the Java editor (**Colors and Fonts > Java > Java Editor Text Font**).

In addition to the above, some text is always displayed in the system font. For example, the navigator tree always does this. To change the font used in these areas, you can use the configuration tools provided by the operating system (for example, the Display Properties control panel on Windows, or the .Xdefaults file in Motif).

Colors

To set the colors used by the Workbench to display error text and hyperlink text:

1. On the main menu bar, click **Window > Preferences**.
2. Expand the **General > Appearance** category and select **Colors and Fonts**.
3. Select the color you want to change in the tree view and click the color bar on the right.
4. Use the dialog which opens to select a color.
5. Click **OK**.

Plug−ins that use other colors may also provide preference entries to allow them to be customized. For example, the searching support provides a preference for controlling the color used to display potential matches (**General > Search > Foreground color for potential matches**).

In general, the Workbench uses the colors that are chosen by the operating system. To change these colors you can use the configuration tools provided by the system (for example, the Display Properties control panel on Windows, or the .Xdefaults file in Motif).

Related concepts

Fonts and Colors in Eclipse

Importing and Exporting Preferences

Preference files can be both imported to and exported from the Workbench allowing you to easily share individual or group preferences.

The Import wizard can be used to import preferences from the local file system to the Workbench.

To import a preference file:

1. Select ***File*** > ***Import***.
2. In the Import wizard select ***Preferences*** and click ***Next***.
3. Click ***Browse...*** and locate the Preferences file on the local file system.
4. Select one of the following:
    - ***Import all*** to accept all of the preferences defined in the file.
    - ***Choose specific preferences to import*** to select only specified preferences defined in the file.
5. Click ***Finish***.

The Export wizard can be used to export preferences from the Workbench to the local file system.

To export a preference file:

1. Select ***File*** > ***Export***.
2. In the Export wizard select ***Preferences*** and click ***Next***.
3. Select one of the following:
    - ***Export all*** to add all of the preferences to the file.
    - ***Choose specific preferences to export*** to add only specified preferences to the file.
4. Click ***Browse...*** and locate the preferences file on the local file system.
5. Click ***Finish***

Note: If no changes have been made to the original preference settings the preferences file will be empty.

Working with projects, folders and files

There are three different types of resources in the workbench: projects, folders, and files. Projects are the largest structural unit used by the Workbench. Projects contain folders and files, and they can be opened, closed, or built. Folders can contain other folders and files. The Workbench provides a number of mechanisms for working with projects, folders and files. See the related tasks section for more details.

Folders and files directly below projects can be linked to locations in the file system outside of the project's location. These special folders and files are called linked resources.

Related concepts

Workbench
Resources
Resource hierarchies
Linked resources

Related tasks

Creating a project
Closing projects
Deleting projects
Creating a folder
Creating a file
Creating linked resources
Moving resources
Copying resources
Renaming resources
Deleting resources
Viewing resources properties

Navigating and finding resources

The Workbench provides a number of mechanisms for navigating and finding resources. See the Related tasks links for more details.

Related concepts

Resources
Navigator view
Search view

Related tasks

Finding a resource quickly
Searching for files
Searching for text within a file
Sorting resources in the Navigator view
Showing or hiding files in the Navigator view
Narrowing the scope of the Navigator view
Linking the Navigator view to the active editor

Bookmarks, tasks and other markers

Markers are objects that may be associated with Workbench resources. There are many uses of markers in the Workbench, including providing support for bookmarking resources or locations within resources, tracking ongoing tasks, or displaying error messages. See the related tasks section for more details.

Related concepts

Bookmarks
Tasks view
Markers

Related tasks

Creating a bookmark within a file
Creating a bookmark for an entire file
Deleting a bookmark
Adding line items in the Tasks view
Associating a task with a resource
Deleting tasks
Filtering the Task view
Automatically fixing problems

Working with local history

A local edit history of a file is maintained when you create or modify a file. Each time you edit and save the file, a copy is saved so that you can replace the current file with a previous edit or even restore a deleted file. You can also compare the contents of all the local edits. Each edit in the local history is uniquely represented by the date and time the file was saved. See the Related tasks links for more details.

Related concepts

Local history

Related tasks

Comparing resources with the local history
Replacing a resource with local history
Restoring deleted resources from local history
Setting local history preferences

Using the help system

The Workbench provides a number of ways to provide help information, including context sensitive help and extensive online documentations. See the Related tasks links for more details.

Related concepts

Online help system

Related tasks

Accessing context sensitive help
Accessing and navigating online help
Searching online help
Setting help fonts and colors for accessibility
Changing the web browser used by the help system

Setting help fonts and colors for accessibility

The help browser uses your operating system's settings for the font colors, styles, and sizes. Users with visual impairments may wish to change some of these settings to increase the readability of the documentation.

WIN In addition, on Windows platforms using Microsoft Internet Explorer, the help browser uses a component of Internet Explorer to display documentation, so changes you make to its display settings also affect the help display. To change the help browser's font and color settings:

1. Open Microsoft Internet Explorer.
2. Select **Tools > Internet Options**.
3. On the **General** page, click the **Colors**, **Fonts**, or **Accessibility** button.
4. Set the formatting options you desire.
5. Optionally, you can specify a cascading style sheet (CSS) to apply to the content.
6. Click **OK** and exit Internet Explorer.
7. Restart the Workbench. Open the Help perspective and browse the documentation to see the changes.

Note: The help system also uses the Icon font setting on the Display Properties **Appearance** tab.

For more information about creating a CSS, consult a CSS reference. The W3 Consortium (www.w3.org) has an extensive collection of information about CSS and links to valuable resources.

Related tasks

Accessing and navigating online help

Changing how help information is displayed

Help browser

The help system can be accessed in help view, or separate help window. The help window may have a form of a simple window with an embedded browser, or be one of the web browser available on your system.

If embedded web browser is supported on your system, help uses an embedded help browser to display help, whenever possible. When a modal window is showing on the screen, that would prevent interacting with the embedded browser, requesting help opens an external browser. If embedded browser is used on your system and you prefer to always use external browser, you may select this behavior in help preferences.

Context help

By default, context−sensitive help for workbench elements is displayed in the help view. Context−sensitive help for dialogs is displayed adjacent to the dialog, in a window similar to the help view. If you prefer to display context help in infopops when working in the workbench, in dialogs or both, change the selection in the help preference page.

Displaying topics

A topic selected in the help view can be displayed in place, in the help view, or in the editor area. You may indicate suitable way in the help preferences. On some platforms, with no embedded browser, the topics will always be displayed in an external browser.

Related reference

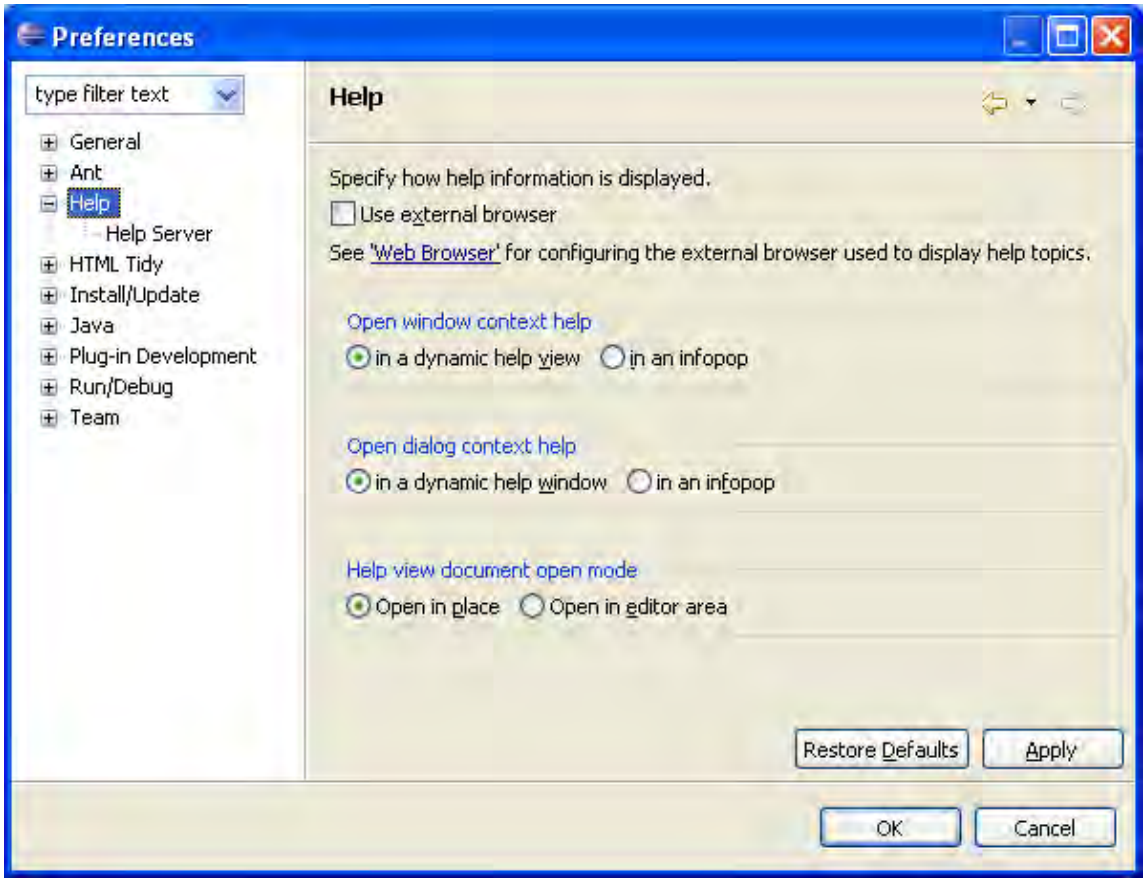
Help preference page

Help

On the Help preferences page, you can indicate how to display help information.

Option	Description	Default
Use external browsers	**If embedded web browser is supported on your system, help window uses an embedded help browser to display help contents, whenever possible, and this option is available. Select it, to force help to use external browsers. Use "Web Browser" preference page to select browser to use.**	**Off**
Open window context help	**This option allows you to determine whether the window context help will be opened in a dynamic help view or in an infopop.**	**in a dynamic help view**
Open dialog context help	**This option allows you to determine whether the dialog context help will be opened in a dynamic help section of help view or in an infopop.**	**in a dynamic help window**
Help view document open mode	**This option allows you to determine whether the documents selected in the help view will be opened in place or in the editor area.**	**Open in place**

Here is what the Help preferences page looks like:



Note: Selection performed on this page can affect how the help view is presented. If the selected browser is not fully compatible with Internet Explorer or Mozilla, or has JavaScript disabled, the help view shown in the

browser might be a simplified version.

Related tasks

Changing the browser used by the help system

Related reference

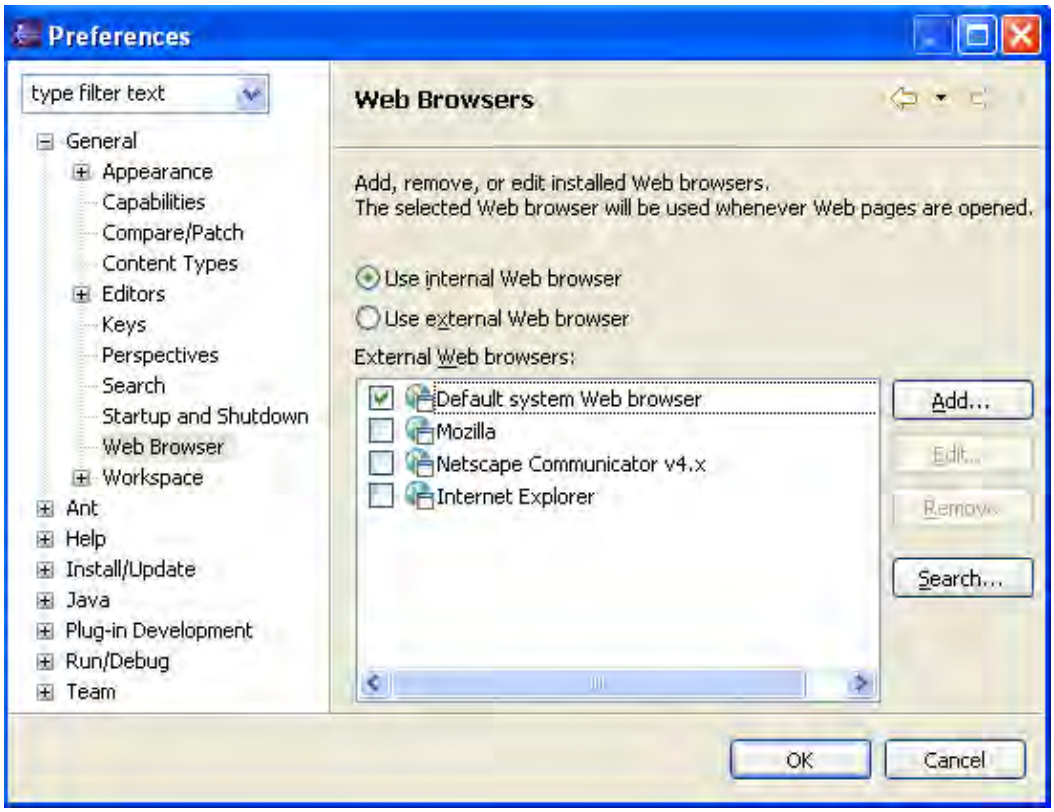
Help View
Web Browser

Web Browser Preference Page

The following preferences can be changed on the Web Browser preference page.

Option	Description	Default
Use internal Web browser	This option enables you to use an internal Web browser.	On
Use external Web browser	This option enables you to use an external Web browser. Select the required browser from the list of available external web browsers.	Off

Here is what the Web Browser preference page looks like:



Working in the team environment with CVS

The Workbench provides tools to manage, share and synchronize resources. The CVS standard is supported by default. See the Related tasks links below for more details.

Related concepts

Team programming with CVS

Related tasks

Working with a CVS repository
Working with projects shared with CVS
Synchronizing with the repository

Working with a CVS repository

In the CVS team programming environment, team members do all of their work in their own Workbenches, isolated from others. Eventually they will want to share their work. They do this via a CVS repository. The Workbench provides a number of mechanisms which support working with CVS repositories. See the Related tasks links for more details.

Related concepts

Team programming with CVS
CVS Repositories

Related tasks

Creating a CVS repository location
Discarding a CVS repository location
Refreshing the CVS Repositories view
Discovering branch and version tags

Working with projects shared with CVS

The Workbench provides a number of mechanisms which support working with projects shared via CVS. See the Related tasks links for more details.

Related concepts

Team programming with CVS
CVS Repositories
Branches
Synchronizing with a CVS repository

Related tasks

Sharing a new project using CVS
Project checked out with another CVS tool
Checking out a project from a CVS repository
Checking out a module from a CVS repository
Disconnecting a project from CVS
Setting the CVS keyword substitution mode
Enabling the CVS resource decorations

Disconnecting a project from CVS

Disconnect a project from CVS to disable the CVS operations that can be performed on the project and it resources and optionally to remove the CVS information (stored in the CVS folders) associated with the project.

To disconnect a project from CVS:

1. In one of the navigation views, select the project to be disconnected.
2. Select **Team > Disconnect** from the project's pop−up menu. The Confirm Disconnect from CVS dialog opens.
3. In the dialog, choose one of:
    a. **Delete the CVS meta information** − disables the CVS team menu operations and removes the CVS folders and their contents from the file system.
    b. **Do not delete the CVS meta information** − disables the CVS team menu operations but leaves the CVS meta information.
4. Click **Yes** to disconnect the project.

Related concepts

Team programming with CVS
Branches
Synchronizing with a CVS repository

Related tasks

Checking out a project from the CVS repository
Replacing resources in the Workbench
Branching
Synchronizing with the repository
Updating
Resolving conflicts
Merging from a branch

Related reference

CVS

Setting the CVS keyword substitution mode

CVS uses the keyword substitution mode of a file to differentiate binary files from ASCII files and to indicate what type of keyword substitution is to take place when files are committed and checked out.

To set the CVS keyword substitution mode:

1. In one of the navigation views, select the files or containing folders for which a change in keyword substitution mode is desired. **Note**: Ensure that any new files that are to be committed are added to CVS version control as the keyword substitution mode can only be set for files that are already under CVS control.
2. From the pop−up menu, select **Team > Change ASCII/Binary Property**. The Set Keyword Substitution Mode wizard will open.
3. The wizard contains a list of all founds found in the selection. On this page, you can do the following:
    - You can enter a file pattern to filter the list of files.
    - You can change the mode for files individually.
    - You can select multiple files and change the mode for the all using the drop down at the bottom of the page.
    - You can select multiple files and click ***Propose*** which determines the mode based on your preference settings.
    - You can check the *Show changes* options to see all the files whose mode has been changed in the wizard.
4. When you are done specifying the new modes for any files, enter the commit comment to be associated with any file commits. Files will need to be committed if changing the file type from Binary to ASCII results in a change in the file content due to line terminator adjustments. Click **Finish** to apply the changes.

Related concepts

Team programming with CVS
Branches
Synchronizing with a CVS repository

Related tasks

Creating a CVS repository location
Checking out a project from the CVS repository
Replacing resources in the Workbench
Branching
Synchronizing with the repository
Updating
Resolving conflicts
Merging from a branch

Related reference

CVS
File Content

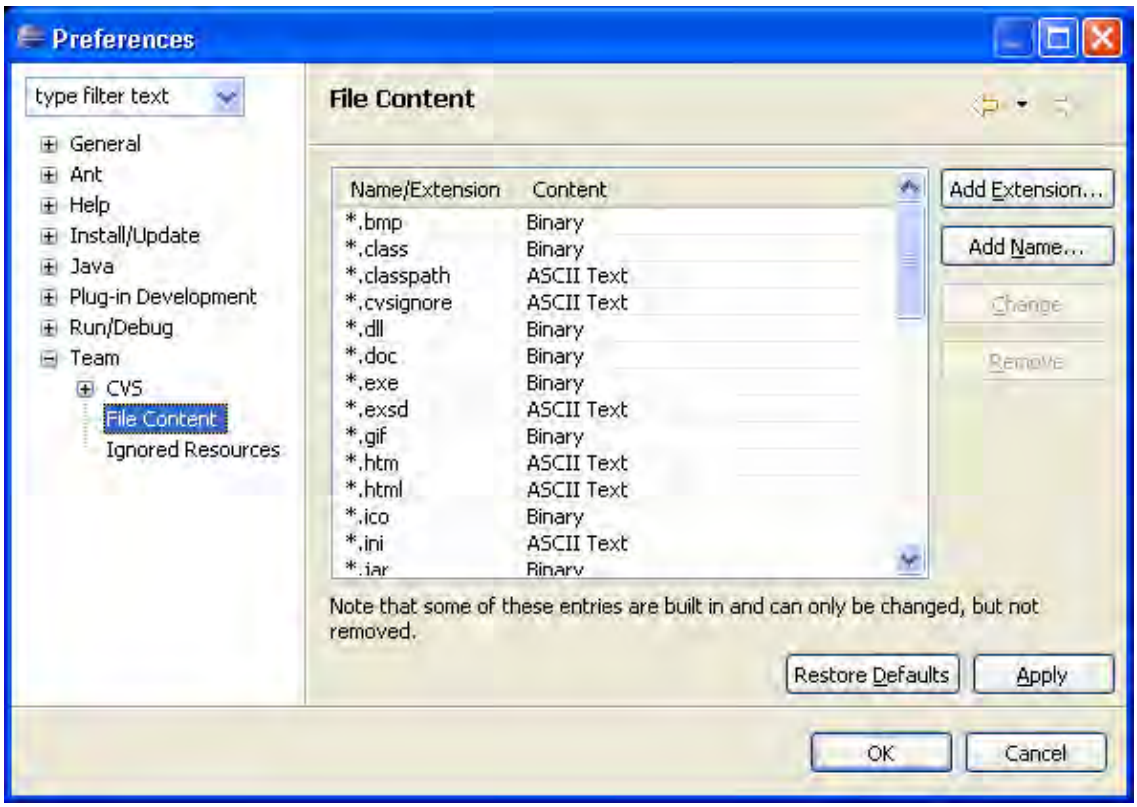
Team File Content

On the Team>File Content preference page, you can associate file names or extensions with the type of data the file contains. The two choices for file content type are ASCII and Binary. Repository providers such as CVS can then use this information to provide content type appropriate behavior. For example, for ASCII files, CVS ensures that line terminators conform to those of the OS platform.

Entries are added to the File Content page in two ways. The first is through contributions from workbench plug−ins. Tools integrated into the workbench provide the workbench with the file content types for file extensions specific to the tool. The workbench itself also defines the file content type for file extensions that are common and appear frequently in the workbench (e.g. html,.png, etc.).

The second method is for users to add file content types explicitly on the File Content preference page. To do so, the user can simply click on the ***Add Extension*** button and enter an extension or click on the ***Add Name*** button and add a name.. Following this, they can toggle the type associated with the name or extension by selecting the entry for the extension in the table and clicking ***Change***. Entries can be removed from the list by selecting them and clicking ***Remove***. Only those entries that were manually added can be removed. Entries contributed through the plug−in manifest can only be changed.

Here is what the Team File Content preference pages looks like:



Related tasks

Setting the CVS keyword substitution mode

Connecting and configuring CVS with SSH

Eclipse includes an SSH client for accessing a remote CVS server. The client supports both the SSH1 and SSH2 protocol versions. To use SSH2 with a CVS repository select the extssh connection type.

Different methods can be used to authenticate, depending on the level of functionality and security you want. User authentication methods used by the client by default are, in the following order: public−key, Keyboard−Interactive, and password authentication.

Authentication with Public Keys (keypair)

Public−key authentication allows you to connect to a remote CVS server without sending your password over the wire. This is a more secure authentication method than password authentication. Public−key authentication uses two keys, a private key that only you have−−it should be kept in a secure place and protected with a password. And the public key, which is placed on the server you wish to gain access to.

Eclipse supports generating both keys and you can copy the public key to the server from within Eclipse. In order to use public key authentication follow these steps:

1. If you already have a private/public key for the CVS server, simply open the Team > CVS > SSH2 Connection Method preference page and on the General tab add your private key to the list by selecting the Add Private Key button. Your key will be used when authenticating.
2. If you don't have a keypair yet, in the preferences page, go to the Key Management page.
3. Select either a Generate RSA key (if your server supports version 1 of the protocol) or Generate DSA key (for version 2).
4. A public key and private key will be generated. The public key will be shown in the read−only text area.
5. The next step is to copy the public key to your server. If your server is running an OpenSSH server than you can use the Export via sftp action. Otherwise you will have to copy and paste the public key into your remote ~/.ssh/authorized_keys file.
6. Now you have to provide a password for your private key, and save it on your computer.

So now that you have your keypairs generated and installed, the next time you create a CVS location to the server you shouldn't enter a password. When the connection is initialized, you will be prompted for the passphrase for your private key. This is the most secure method of authentication, as long as your private key is protected with a passphrase.

Note: exporting the public key is only supported on OpenSSH enabled SSH servers.

Authentication with Passwords

If your server is configured to support password authentication, then you can simply enter your password when you create a CVS repository location and that password will be used when authenticating with the server. This is the simplest authentication method, but isn't as secure as public keys.

Note: It is not recommended that you save your login passwords using Eclipse. The file that contains the file is not sufficiently protected against intruders and as such you can comprise the security of your system. If your company requires strict security practices then your should use keypair authentication.

Related concepts

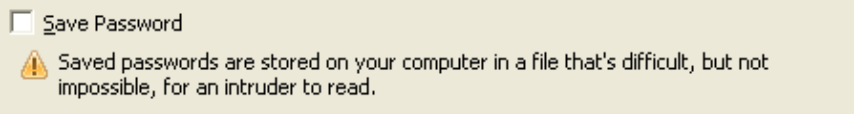
Creating a CVS repository location
Team programming with CVS

Related reference

CVS

Password Management

When creating a repository location, an option is provided at the bottom of the wizard to save the password between sessions.



In previous releases, passwords were automatically saved between sessions whereas now the user has full control over this behavior. There is a caveat to this though − the password file Eclipse uses to save the password between sessions is not very secure. It is difficult, but not impossible, for an intruder to read the file and find your passwords.

We recommend that, if you have strict security requirements, you don't save passwords using this mechanism. Instead, use keypair authentication provided by the *extssh* connection method.

You can manage the list of passwords that have been cached via the Team > CVS > Password Management preference page. The page displays the currently saved passwords and allows you to delete them.

Related concepts

Creating a CVS repository location
Connecting and configuring CVS with SSH
Team programming with CVS

Related reference

CVS

Setting the content type of a file extension

CVS repositories distinguish between files that contain ASCII data and those that do not. For ASCII files, additional helpful functionality is available to the user. This includes:

- Proper end of line conversion between the client and the server. This ensures that a Windows user and a UNIX user can work on the same file without introducing incompatibilities.
- Auto merging of conflicts. If a file contains both incoming and outgoing changes but none on the same line as another, then the file may be automatically merged when updating.

Certain file types invariably contain either ASCII data or binary data. For example, *.txt files usually contain ASCII data, and *.exe files usually contain binary data. Eclipse comes with a pre−defined set of file types, which you may add to or modify.

To set the content type associated with a file extension:

1. From the main menu bar, select **Window > Preferences** to open the Preferences dialog.
2. In the dialog, select the **Team > File Content** preference page. This page displays a list of file extensions and content type (ASCII or Binary) for file extensions whose content type is known.
3. To add a file extension, click the **Add Extension** button and enter the file extension in the text prompt that appears. Once **OK** is clicked, the extension will be added to the list with a content type of ASCII.
4. To change the content type for an existing file extension, select the file extension entry and click the **Change** button. This will toggle the type from ASCII to Binary or vice versa.
5. To remove a file extension, select the file extension entry and click the **Remove** button.

Related concepts

Team programming with CVS

Related reference

CVS

Filtering in the CVS Resource History View

Over time, the revision history for a file can grow drastically. As a result, it may be difficult to find the revision or revisions that you are looking for. Using the filtering mechanism of the CVS Resource History view can help.

To set a filter on the CVS Resource History view:

1. Select the **Filter History** action from the local toolbar of the CVS Resource History view.
2. Enter any author, comment, or date range that you would like the view to display.
3. Click **OK**.

The view will now display only those revisions that match the criteria you entered.

Tip: You can remove a date range filter by selecting the first item ("−−−") in the date and month fields.

Tip: You can choose whether the filter should match any or all of the criteria by selecting the appropriate radio button in the dialog.

Related concepts

Team programming with CVS

Related reference

CVS
CVS Resource History view

Sharing your workspace setup using Project Sets

Your workspace setup may consist of several projects from one or more repositories. Once you have setup your workspace, you can share it with others by exporting a Team Project Set. A project set is a text file that contains a pointer to each of the projects contained in the project set.When a project set is imported, these pointers are used to fetch the projects from the repository. Project sets can include any projects that are mapped to repository tooling that provides support for them, such as CVS. To export a project set:

1. Setup your workspace with all the projects you want to work on by checking them out of CVS or obtaining them in any manner appropriate for the repository tooling you are using.
2. From the ***File*** menu, choose ***Export***. The Export dialog will open.
3. In the Export dialog, choose Team Project set and click ***Next***. The dialog will now display all the projects that are eligible for export.
4. Check the projects you wish to include in the project set. Either browse for or type in the name of the file where you wish to save the project set and then click ***Finish***.

Now you have created a project set file. You can now share this file with others or use it yourself to recreate your workspace. To Import a project set:

1. From the ***File*** menu, choose ***Import***.This will open the Import dialog.
2. In the Import dialog, choose Team Project set and click ***Next***.
3. Browse for or type in the name of the file containing the project set and click Finish.
4. The projects contained in the project set will be fetched from the repository.

For CVS, you will be prompted to provide the authentication information (user name and password) for the repository as it is not included in the project set.

Related concepts

Team programming with CVS
CVS Repositories

Related tasks

Checking out a project from a CVS repository
Checking out a module from a CVS repository
Creating a CVS repository location
Sharing a new project using CVS

Related reference

CVS
CVS Repositories view
CVS Checkout wizard

Versioning

Resources can be versioned in order to capture a snapshot of their current state at one specific point in time. See the Related tasks links for more details.

Related concepts

Versions
Branches

Related tasks

Creating a version of a project
Enabling the CVS resource decorations
Versioning projects in the repository

Finding out who to blame with the Annotate command

Let's say you have found a bug on line 65 of a file and you don't understand the code. Who do you ask, or blame, for the change? Well you could start by looking at the resource history for the file, but that won't tell you who changed that particular line. This is why the Annotate command is useful. You can pick any ASCII file (see note on binary files) and get a listing of who changed what line.

The Show Annotation action is available form the following places: History View, Repository Explorer, Synchronize View, and the Resource and Packages View. When the annotate is run you will be able to:

- Step through changes in the Annotate View and the text editor will highlight the associated lines associated with the selected change.
- You can select a line in the text file and the Annotate View will select the change that is associated with that line.
- The History View will show the history for the opened file and highlight the revision of the currently selected change. This allow you to quickly see the commit comment for a particular change.

Only works with text files

The annotate command will only work with files that are marked as ASCII in the CVS repository. Also, the command will open a text file to show the changes even if the associated editor in the workbench is a non−text editor. For example, if you run annotate on a plugin.xml file a simple text editor will be opened instead of the full PDE editor.

Related concepts

Watch/Edit
Team programming with CVS

Related reference

CVS

Quick Diff: Showing changes in a text editor

Instead of using a compare editor, which will show changes between 2 or 3 files by showing each file side−by−side, you can enable quick diff support and see the changes within the text editor (e.g. any text editor based on the Eclipse text editor). This feature can be enabled via the General > Editors > Text Editors > Quick Diff preference page. You should select the Latest CVS Revision as the reference provider. This will annotate the text file with diffs against the latest revision in CVS. Here are the following scenarios that are useful:

1. Open a file and make changes to it. You will see the difference annotations marking the changes. Then if you run Team > Replace with latest. The annotations are removed and the file is clean.
2. Open a file and make changes to it. You will see the difference annotations marking the changes. Then if you run Team > Commit the annotations are removed and the file is clean.
3. If you synchronize the file with the server and a new revision is found on the server, the editor will update showing the incoming changes.

If you enable showing the differences in the overview ruler, you can, at a glance, get an idea of how many changes you have made to a file since your last commit. The differencing happens in a background thread to minimize the impact on actual editing of the file.

Note: this feature requires an active connection to your CVS server so that remote contents can be fetched when an editor is opened.

Related concepts

Watch/Edit
Team programming with CVS

Related reference

CVS

Changing CVS team settings

You can customize the settings related to a number of CVS team views and operations.

1. From the main menu bar, select **Window > Preferences**.
2. On the left side of the Preferences window, select the **Team** category.
3. You will see a number of general **Team** options. These options can be used by any Eclipse integrated repository. In our case, they are all applicable to CVS.
4. You will also see a **CVS** category. Select it. You will see various CVS options. If you expand the **CVS** category, you will see additional CVS preference pages.
5. Underneath the ***CVS*** category are several subcategories. For example, there are sections for ***Watch/Edit***, ***Console***, ***Label Decorations*** and ***SSH2*** among others.

In the Preferences dialog, you can search by keywords such as CVS or SSH2 in order to show only the relevant pages.

Changing CVS Project Settings

You can change CVS project settings in the properties dialog of a project.

- From the context menu of a view showing the project, select Properties
- Click on the CVS category

From this dialog, you can:

- Enable Watch/Edit on the project.
- Specify whether new or absent folders should be fetched when an update is performed.
- Change the repository location to which the project is associated.

Related concepts

Team programming with CVS

Related tasks

Synchronizing with the repository
Committing

Related reference

CVS Preference pages

CVS

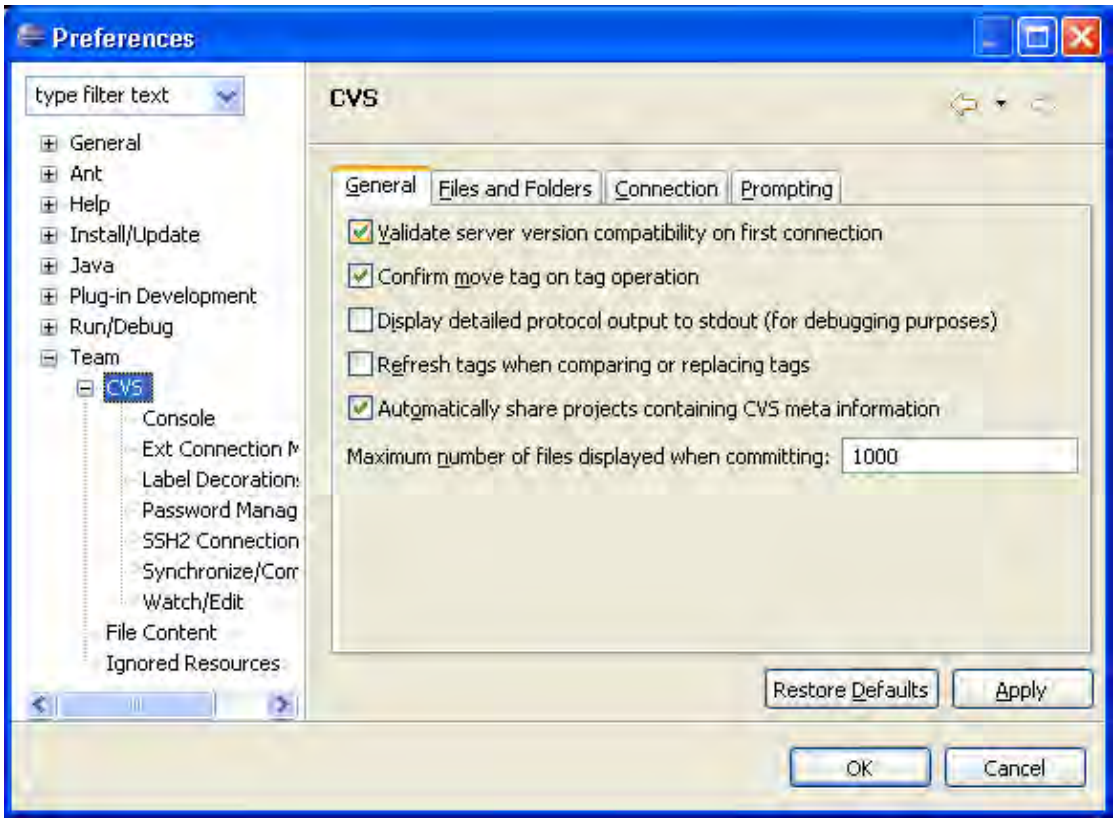
The following sections describe the preferences available in the tab groups of the CVS preferences page.

General

On the General tab group of the CVS preference page you can customize several aspects of the CVS Plug−in.

Option	Description	Default
Validate server version compatibility on first connection	Use this option to enable a query of the CVS server version on the first connection to determine server compatibility. The server version will be output to the console and if an incompatibility is detected a warning message will be logged when connecting.	Enabled
Confirm move tag on tag operation	Use this option to be prompted when the Move tag option is chosen when tagging.	Enabled
Display detailed protocol output to stdout	Use this option to display the communication trace between the Workbench and a CVS server.	Disabled
Refresh tags when comparing or replacing tags	Use this option to have the Compare with and Replace With tag dialogs automatically refresh the known tags by contacting the server.	Disabled
Automatically share projects containing CVS meta information	Use this option to have any imported project that was checked out from CVS using a different CVS tool automatically shared with CVS.	Enabled
Maximum number of files displayed when committing	Use this option to limit the number of files that get displayed in the commit dialog.	1000

This is what the General tab group of the CVS preference page looks like:

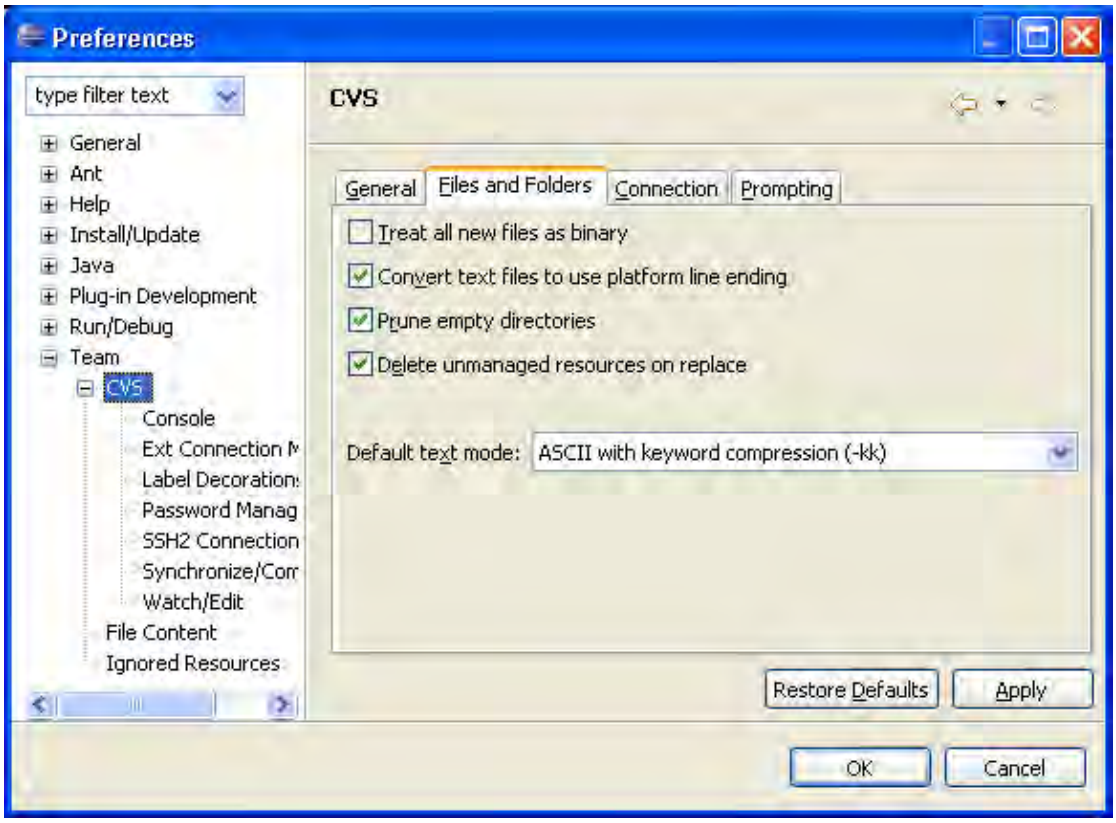


Files and Folders

On the File and Folders tab group of the CVS preference page you can customize several aspects of the CVS Plug−in.

Option	Description	Default
Treat all new files as binary	Use this option to override the file content settings and treat all new files as binary.	Disabled
Convert text files to use platform line ending	Use the option to convert the line endings of text files to the line ending used by the platform. This option can be disabled if you are checking out resources to a *nix drive that is mounted on a Windows machine.	Enabled
Prune empty directories	Use this option to specify the pruning of empty directories on update and in the synchronization view. Although pruned directories aren't shown in the workbench there is actually still an empty directory in the repository. This is helpful because CVS doesn't provide a client with the ability to remove directories from the server.	Enabled
Delete unmanaged resources on replace	Use this option to allows resources not under CVS control to be deleted when replacing with resources from the repository.	Enabled
Default text mode	Use this option to set the default keyword substitution for text files.	ASCII with keyword expansion −kkv

This is what the File and Folders tab group of the CVS preference page looks like:

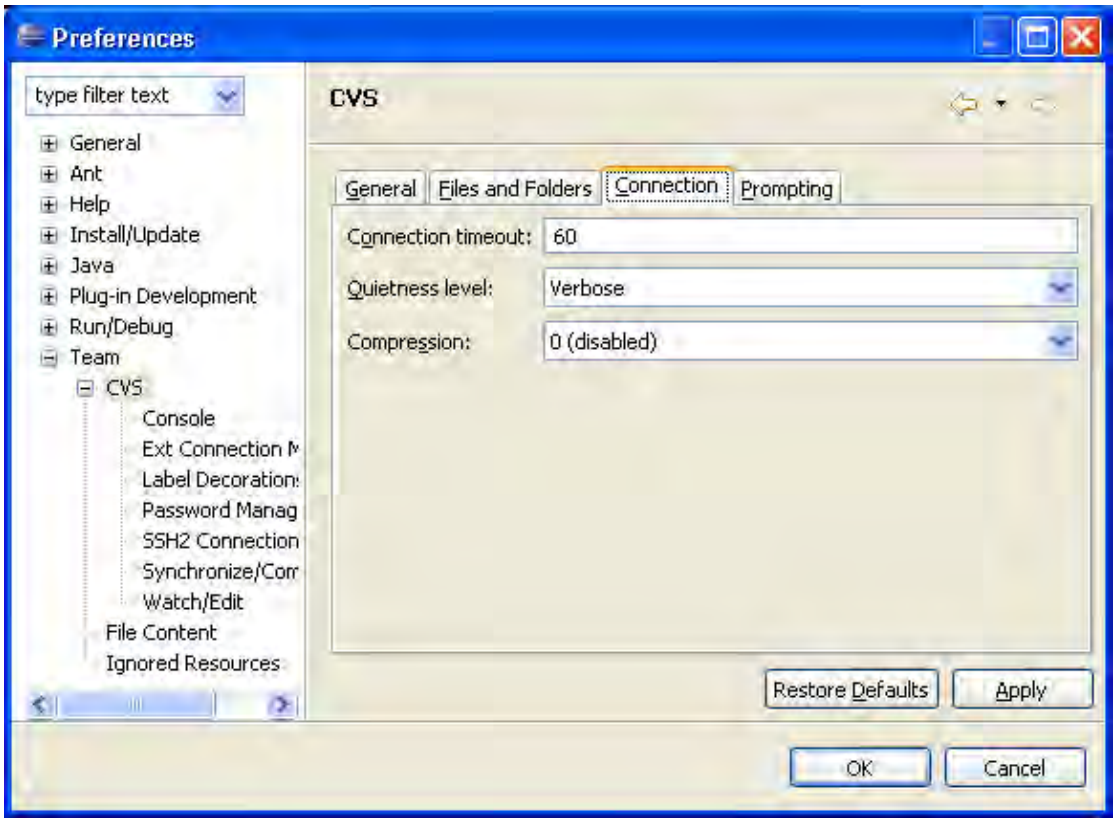


Connection

On the Connections tab group of the CVS preference page you can customize several aspects of the CVS Plug−in.

Option	Description	Default
Communication timeout	Use this option to configure the amount of time in seconds to wait before timing out from a connection to the CVS server.	60 seconds
Quietness level	Sets the amount of status information CVS prints for a command. In ***Somewhat quiet*** mode the printing of unimportant informational messages is suppressed. The consideration of what is important depends on each command. In ***Very quiet*** mode all output except what is absolutely necessary to complete the command is suppressed. In ***Very Quiet*** mode, some CVS servers may not communicate important information about errors that have occurred. You may want to consider using ***Somewhat quiet*** mode instead.	Verbose
Compression	Use this option to set the compression level used when sending files between the client and server.	0

This is what the Connection tab group of the CVS preference page looks like:



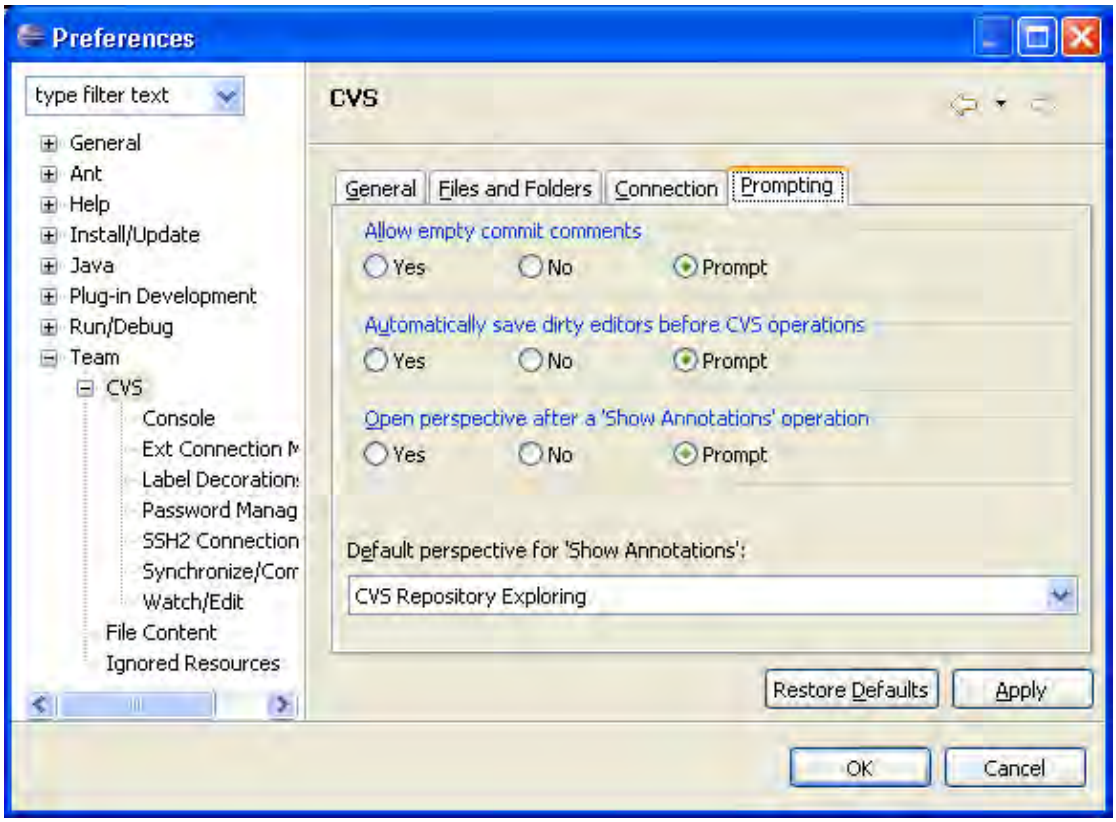
Prompting

On the Prompting tab group of the CVS preference page you can customize several aspects of the CVS Plug−in.

Option	Description	Default
Allow empty commit comments	Use the option to configure what happens if a commit comment is not provided when committing. • ***Yes:*** Allow the commit to occur without a comment. • ***No:*** Do not allow the commit to proceed until a comment is provided. • ***Prompt:*** Ask the user whether they want to allow the commit to proceed without a comment.	Prompt
Automatically save dirty editors before CVS operations	Use this option to configure what happens when there are open editors with unsaved changes when a CVS operation is performed. Options are: • ***Yes:*** Automatically save unsaved changes in open editors before each CVS operation. • ***No:*** Continue CVS operations even if there are unsaved changes in open editors. • ***Prompt:*** Ask the user what to do with unsaved changes in open editors.	Prompt
		Prompt

Open perspective after a 'Show Annotations' operation	Use this option to configure whether another perspective is opened when the Show Annotations operation is run. • ***Yes:*** Open the perspective chosen in the Default perspective for 'Show Annotations' preference. • ***No:***Do not open another perspective. • ***Prompt:*** Ask the user whether they want to open the perspective chosen in the Default perspective for 'Show Annotations' preference or remain in the current perspective.	

This is what the Prompting tab group of the CVS preference page looks like:



Related reference

Perspectives

Perspectives

A perspective defines the initial set and layout of views in the Workbench window. One or more perspectives can exist in a single Workbench window.

Perspectives can be opened either in the same (existing) Workbench window, hiding the current perspective, or in a new Workbench window

Perspectives define visible action sets, which you can change to customize a perspective. You can save a perspective that you build in this manner, making your own custom perspective that you can open again later.

The Workbench defines the *Resource* perspective by default. This perspective shows views relevant to resource management.

Related reference

Window Menu
Editor Area
Navigator View
Outline View
Tasks View

Window menu

This menu allows you to display, hide, and otherwise manipulate the various views, perspectives, and actions in the Workbench.

New Window

This command opens a new Workbench window with the same perspective as the current perspective.

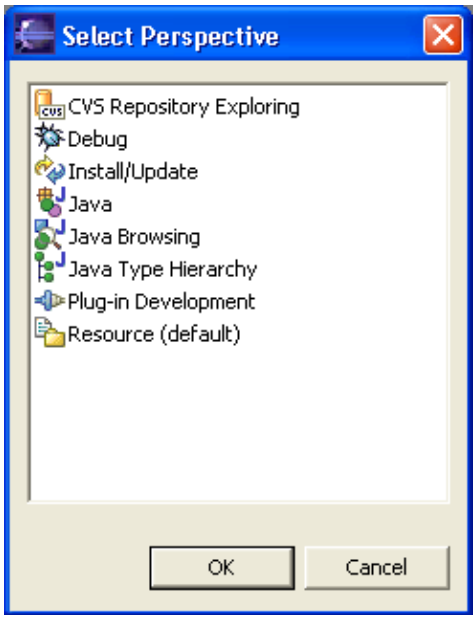
New Editor

This command opens an editor based on the currently active editor. It will have the same editor type and input as the original.

Open Perspective

This command opens a new perspective in this Workbench window. This preference can be changed on the **General** > **Perspectives** preference page. All of the perspectives that are open within the Workbench window are shown on the shortcut bar.

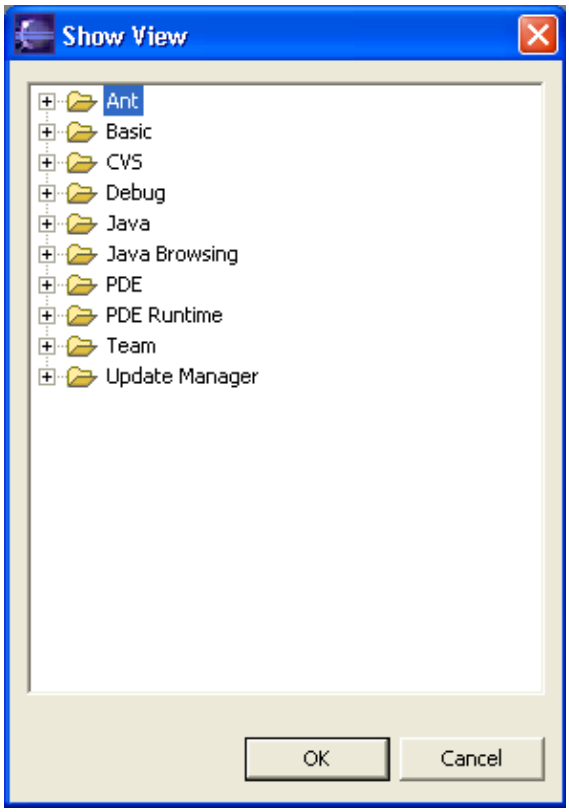
The perspectives you will likely want to open are listed first. This list is dependent on the current perspective. From the **Other...** submenu you can open any perspective.



Show View

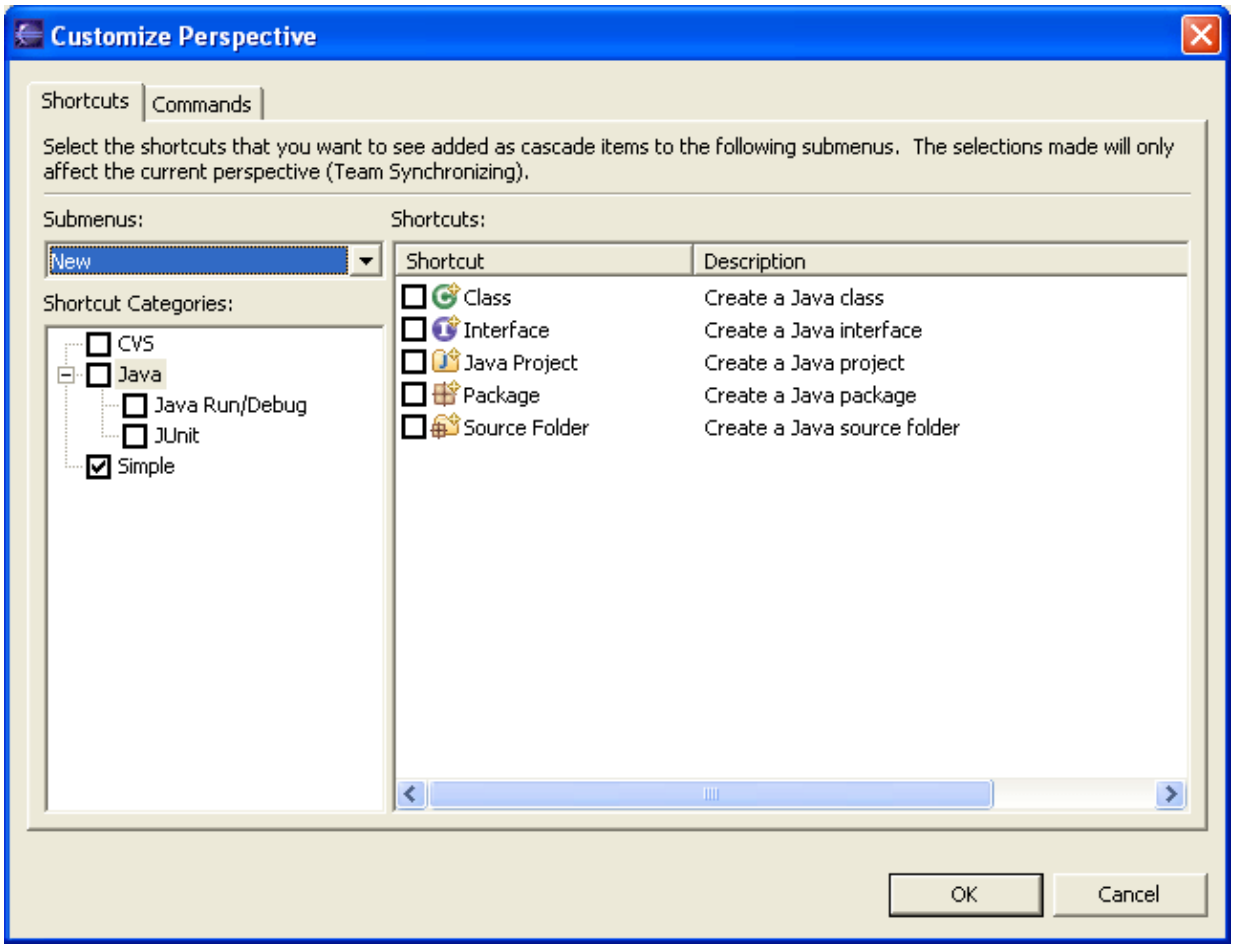
This command displays the selected view in the current perspective. You can configure how views are opened on the **General** > **Perspectives** preference page. Views you are likely to want to open are listed

first. This list is dependent on the current perspective. From the **Other...** submenu you can open any view. The views are sorted into categories in the Show View dialog.



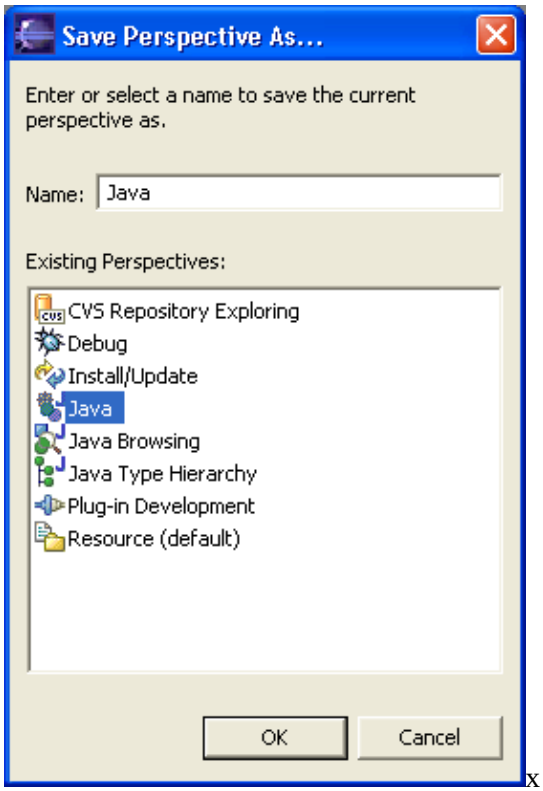
Customize Perspective

Each perspective includes a predefined set of actions that are accessible from the menu bar and Workbench toolbar.



Save Perspective As

This command allows you to save the current perspective, creating your own custom perspective. You can open more perspectives of this type using the **Window > Open Perspective > Other** menu item once you have saved a perspective.



x

Reset Perspective

This command changes the layout of the current perspective to its original configuration.

Close Perspective

This command closes the active perspective.

Close All Perspectives

This command closes all open perspectives in the Workbench window.

Navigation

This submenu contains shortcut keys for navigating between the views, perspectives, and editors in the Workbench window.

- **Show System Menu**: Shows the menu that is used for resizing, closing or pinning the current view or editor.
- **Show View Menu**: Shows the drop down menu that is available in the toolbar of the active view.
- ***Maximize active view or editor:*** Causes the active part to take up the entire screen, or if it already is, returns it to its previous state.
- **Activate Editor**: Makes the current editor active.
- **Next Editor**: Activates the next open editor in the list of most recently used editors.
- **Previous Editor**: Activates the previous open editor in the list of most recently used editors.

- ***Switch to editor:*** Shows a dialog that allows switching to opened editors. Shows a dialog that allows switching to opened editors.
- **Next View**: Activates the next open view in the list of most recently used views.
- **Previous View**: Activates the previous open view in the list of most recently used editors.
- **Next Perspective**: Activates the next open perspective in the list of most recently used perspectives.
- **Previous Perspective**: Activates the previous open perspective in the list of most recently used perspectives.

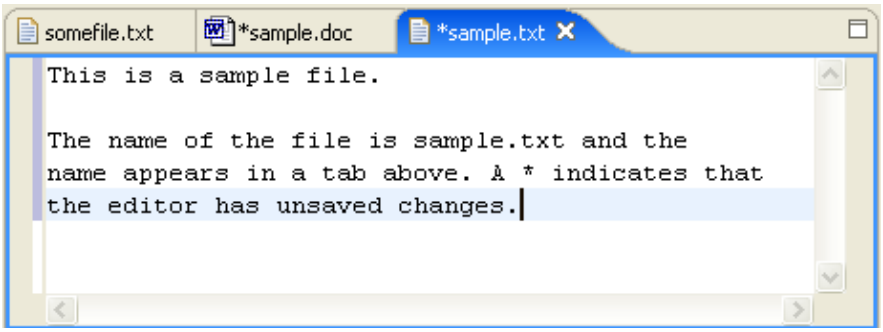
Preferences

This command allows you to indicate your preferences for using the Workbench. There are a wide variety of preferences for configuring the appearance of the Workbench and its views, and for customizing the behavior of all tools that are installed in the Workbench.

Editor area

The editor area is where you modify the contents of files in the Workbench.

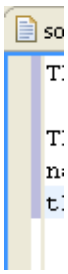
Here is what the editor area looks like when multiple files are open and a text file is being edited:



Marker bar

The marker bar is the vertical bar located at the left of the editor area.

Here is what the marker bar looks like:



Markers

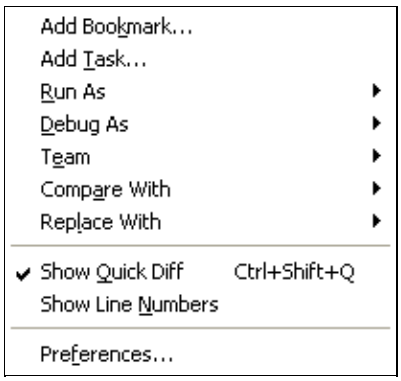
Markers are displayed in the marker bar, to the left of the text editor.

Depending on the type of file displayed in the editor area, three kinds of markers may be displayed:

- Bookmarks
- Task markers (for associated tasks)
- Debugging breakpoints

You can create and associate a marker with a specific line in a file by accessing the context menu from the marker bar, which is directly to the left of that line.

Here is what the context menu of the marker bar looks like:

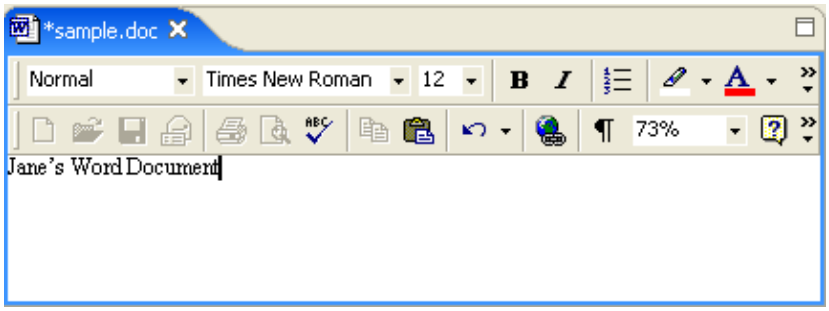


Types of editors

The Workbench uses three types of editors:

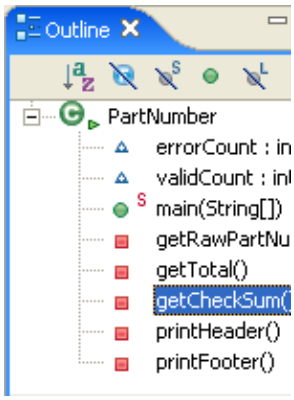
- Internal: These editors are launched inside the editor area in the Workbench window.
- External: You can go outside the Workbench in the file system, edit a Workbench file outside the Workbench, and save the edited file. For example, imagine that you add an SGML file to the Workbench. Later, you go into the file system and open the file in an SGML editor, then save the file. The edited SGML file is still represented in the Workbench, even though you did not edit the file in the Workbench. If you associate a file type with an external editor in the Workbench (**General > Editors > File Associations** preferences page), then the Workbench will launch this external editor.
- ActiveX: On Microsoft Windows platforms, the Workbench makes use of ActiveX controls for applications that allow for them. For example, Microsoft Word supports being embedded as an OLE document. Thus if you have a *.doc* file in the Workbench, and Word is registered as the editor for *.doc* files in your operating system, then opening the file will launch Word as an OLE document within the Workbench editor area. Notice how OLE documents also add such features as menus and toolbar buttons.

The following illustrates Microsoft Word embedded as an OLE document:



Outline view

This view displays an outline of a structured file that is currently open in the editor area, and lists structural elements. The contents of the outline view are editor−specific. In the example below, which is for a Java source file, the structural elements are classes, variables, and methods. The contents of the toolbar are also editor−specific.



`

Restoring deleted files from the repository

The CVS plug−in allows you to delete files from the CVS repository. If you delete a managed file and commit the deletion to the server, the file is deleted from the current branch or version. Although the file has been deleted from the current branch, the file and its previous revisions are actually still on the server. The CVS plug−in provides a tool to help restore a file to your workspace that has previously been deleted from the CVS repository:

1. Select a folder managed by CVS and from the context menu select ***Team*** **>** ***Restore from Repository***.
2. The repository will be searched for deleted files and the list will be shown in a dialog.
3. Select a file in the left most pane to display the revisions of the file that are available in the repository.
4. Select a revision in the right−most pane to view the contents in the bottom text pane.
5. Check the revision that you would like to restore.
6. Click ***Finish*** once you have checked off a revision for each file you wish to restore.
7. After restoring the file if you want to actually restore the file to the current branch you have to add then commit the file back into the repository. If the filename does not change, revision history will be maintained for the restored file.

Related concepts

Team programming with CVS

Related tasks

Synchronizing with the repository
Committing

Reverting a branch to a previous version

It is often useful to revert the contents of a branch to those of a specific version. For example, if your current branch contains changes that you no longer want to release you can revert all or a portion of a project to the contents of any version.

1. Checkout into your workspace the contents from the branch that you want to revert.
2. Select ***Compare With > Another Branch or Version*** on the resource(s) that you want to revert.
3. From the tag selection dialog box select the version to which you want to revert the branch.
4. When the compare editor opens, review the differences that are shown and ensure that they are what you expected.
5. Select the root folder in the compare view and from the context menu select ***Override and Update***. After the operation is completed the folder or project you compared against will have exactly the same contents as the remote revision.
6. You can verify this by performing another comparison against the version. (***Note***: That the CVS preference to ***Consider file contents in comparisons*** should be enabled for this to work.)
7. The comparison should report that there are no changes.

Once your workspace contains the new contents, run your tests then commit the changes to the branch.

Related concepts

Team programming with CVS

Related tasks

Synchronizing with the repository
Committing

Running the CVS command−line client outside of Eclipse

Compatibility

Because the Eclipse CVS plug−in stores its meta information in a format that is compatible with the command−line CVS client you should be able to use a CVS command line client against Eclipse workspace files on disk. The metadata is stored in CVS/ sub−directories but you rarely see them within Eclipse. They are marked as private which causes them to be hidden from view. If you open a (non−Eclipse) file explorer you will see that these directories and their contents appear on the file system.

Don't forget to refresh!

Whenever you use external tools to modify workspace files, you must perform a ***Refresh*** from within Eclipse to make the workspace aware of the changes. If you get a *resource out of sync* error in Eclipse it is a sign that there are resources in Eclipse that have been modified outside of Eclipse. One solution is to perform a refresh (available from a resource's popup menu) on any resources or projects that where modified outside of Eclipse. There is also a preference to refresh automatically.

Caveats

1. Deleted folders

You may encounter unexpected behavior when using the command−line CVS client in conjunction with deleted folders. Eclipse's CVS support keeps track of deleted folders and their contents so that, on the next synchronization, the Synchronize view can properly report on the changes. This information is kept outside of the CVS meta folder structure. This is because in CVS you normally inform the repository of deletions prior to deleting them locally, which is a different workflow than we like to support in the Synchronization view. Thus it is recommended that you do not use the command−line CVS client while you have pending deletions to commit. In some circumstances it could cause the Synchronize view to display incorrect contents, although it will not cause any lost work.

2. CVS directories appear in the workbench

When you use the command−line CVS the CVS folders can sometimes appear in one of the navigation views. There are some cases where CVS folders are not hidden from the UI as the user would expect. For instance, CVS folders will appear if a user imports a CVS project into Eclipse before the CVS plug−in is loaded. To avoid this, open the CVS Repositories view (thus loading the CVS plug−in) before importing CVS projects into Eclipse.

3. The 'extssh' connection method

The *extssh* connection method is unique to Eclipse CVS and doesn't exist in the command−line CVS client. If you are using *extssh* to connect to a SSH2 repository and would like to use the command line client for some CVS operations you can configure the *ext* connection method to use *extssh* when inside Eclipse.

1. Create a new repository location that uses the *ext* connection method. The repository path, host, and user should be identical to those in the *extssh* location.

2. Open the ***Team>CVS>Ext Connection Method*** preference page.
3. Enable ***Use another connection method type to connect*** and choose *extssh*.
4. Now when you use the *ext* connection method inside Eclipse, *extssh* will be used instead of an external client.

Related tasks

Changing the properties of a CVS repository location
Creating a CVS repository location
Using projects checked out with another CVS tool

Crash recovery

The Workbench periodically saves a snapshot in order to reduce the risk of losing data due to crashes.

- Saved data is never lost as it is written to disk immediately (upon save).
- Unsaved data in open editors may be lost, depending on the editor implementation.
- Bookmarks and tasks might be lost.
- If a crash occurs during CVS synchronization, the Workbench may be out of sync. You can check by performing the synchronize operation again.
- Previously−created projects are never lost.

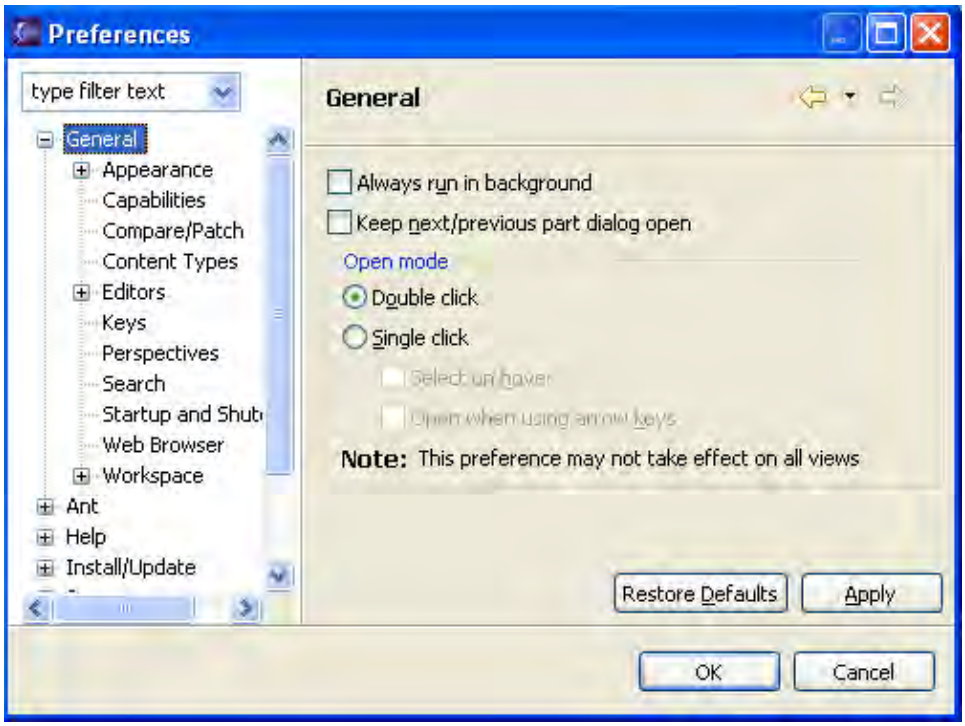
Developers of plug−ins can also choose to participate in this lifecycle and save the state of their plug−ins.

Preferences

The Preferences dialog is the dialog used to set user preferences. The Preferences dialog pages can be searched using the filter function. To filter by matching the page title, simply type the name of the page you are seeking and the available pages will be presented below. The filter also searches on keywords such as appearance and java. The history controls allow you to navigate through previously viewed pages. To step back or forward several pages at a time, click the drop down arrow and a list of the most recently viewed preference pages will appear.

The Preferences dialog can be found from the main workbench **Window** menu under **Window > Preferences**. Preference pages contributed by plug−ins will be found in this dialog.

Here is what the preferences dialog looks like:



Related reference

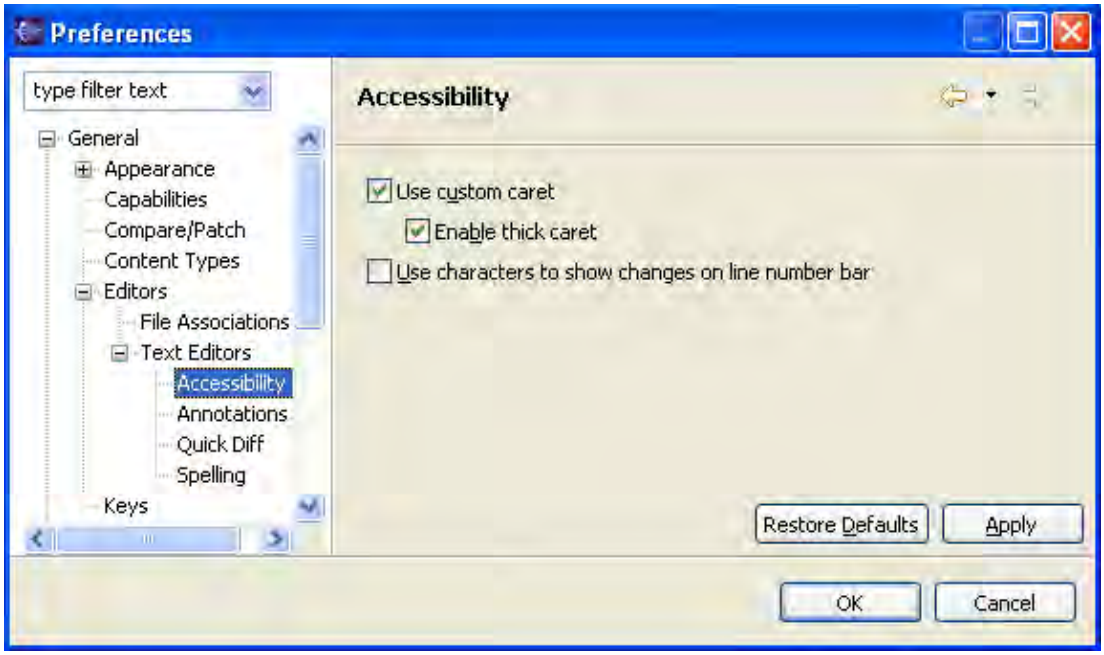
Workbench window layout

Accessibility Preference Page

The following preferences can be changed on the Accessibility page.

Option	Description	Default
Use custom caret	This option replaces the original caret with a custom caret and shows a different caret for Overwrite and Insert mode.	On
Enable thick caret	This option replaces the original caret with a more visible, thicker caret.	On
Use characters to show changes on line number bar	Quick Diff shows the changes in a vertical ruler using colors. Color blind persons can enable this option to show the differences with different characters in the line number ruler.	Off

Here is what the Accessibility preference page looks like:

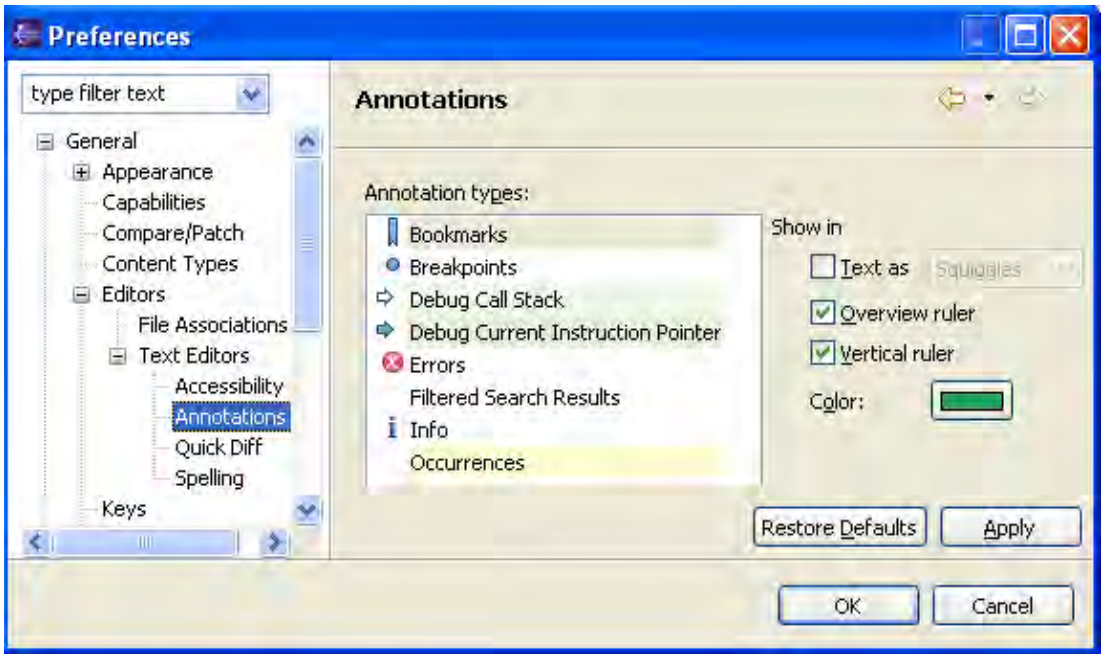


Annotations preference page

The following preferences can be changed on the Annotations preference page.

Option	Description
Show in Text as	This option controls whether the selected annotation type is shown in the text. The corresponding text will be underlined with squiggles or highlighted.
Show in Overview ruler	This option controls whether the overview ruler on the right side of the text editor is shown.
Show in Vertical ruler	This option controls whether the selected annotation type is shown in the vertical ruler.
Color	This option controls the color for the selected annotation type.

Here is what the Annotations preference page looks like:

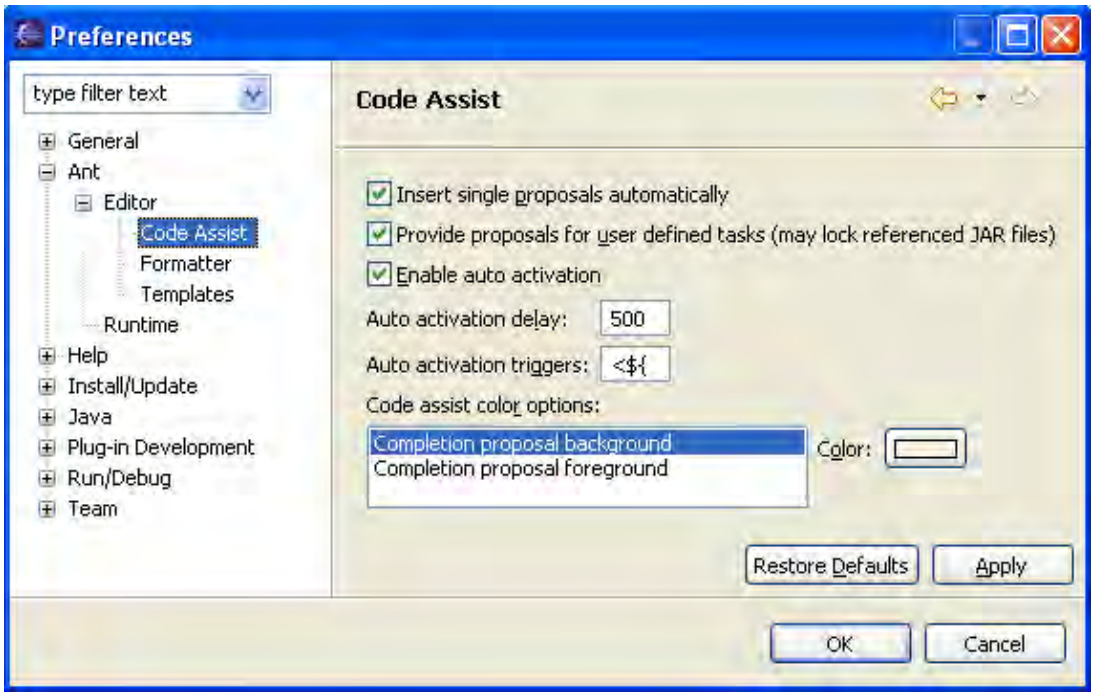


Ant Code Assist

The following preferences can be changed on the Ant Code Assist preference page.

Option	Description	Default
Insert single proposals automatically	This option allows you to automatically insert single proposals.	On
Provide proposals for user defined tasks (may lock referenced JAR files)	This option allows you to provide proposals for user defined tasks.	On
Enable auto activation	This option enables and disables auto activation.	On
Auto activation delay	This option allows you to set the auto activation delay.	500
Auto activation triggers	This option allows you to set the auto activation triggers.	<${
Code assist color options	This option allows you to set the color for the completion proposal background and foreground.	

Here is what the Ant Code Assist preference page looks like.



Related concepts

Ant support

Related reference

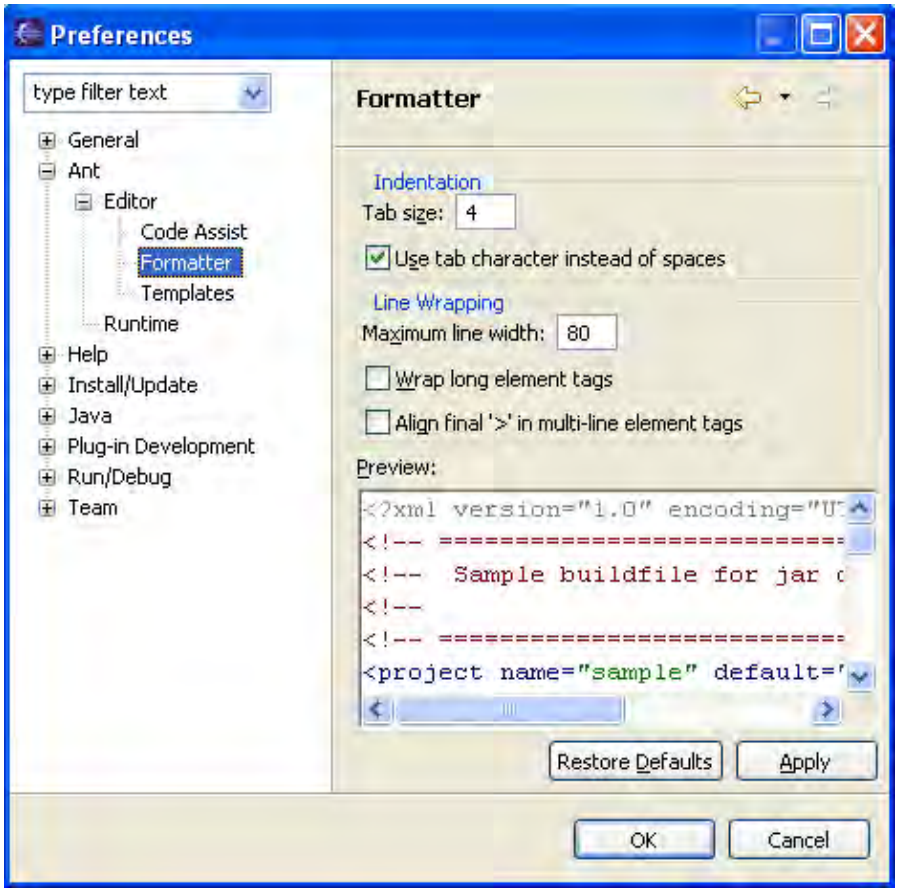
Ant Editor
Ant preferences
Ant runtime preferences

Ant Formatter

The following preferences can be changed on the Ant Formatter preference page.

Option	Description	Default
Indentation − Tab size	This option controls how many spaces are used to display tabs in the Ant editor.	4
Line Wrapping − Maximum line width	This option allows you to set the maximum line width for the Ant editor.	80
Line Wrapping − Wrap long element tags	This option allows you to wrap long element tags. The Preview pane will display a sample buildfile with this option.	Off
Line Wrapping − Align final '>' in multi−line element tags	This option allows you to align the final '>' in multi−line element tags. The Preview pane will display a sample buildfile with this option.	Off

Here is what the Ant Formatter preference page looks like.



Related concepts

Ant support

Related reference

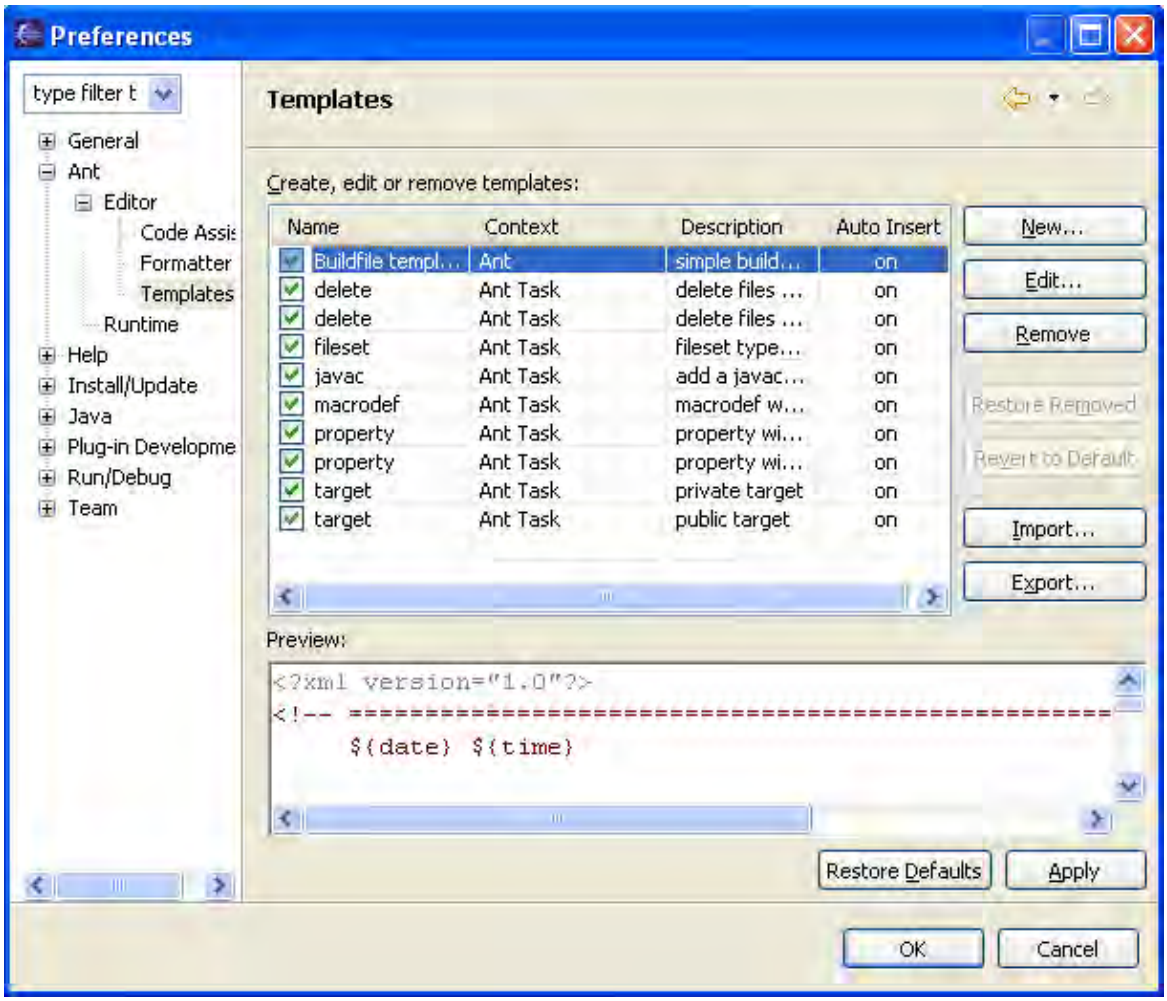
Ant Editor

Ant preferences
Ant runtime preferences

Ant Templates

The Ant Templates preference page can be used to create, edit or remove templates. You can also import templates as well as restore removed templates and revert to the default templates. The preview pane allows to see what your new or edited template will look like before applying your changes.

Here is what the Ant Templates preference page looks like.



Related concepts

Ant support

Related reference

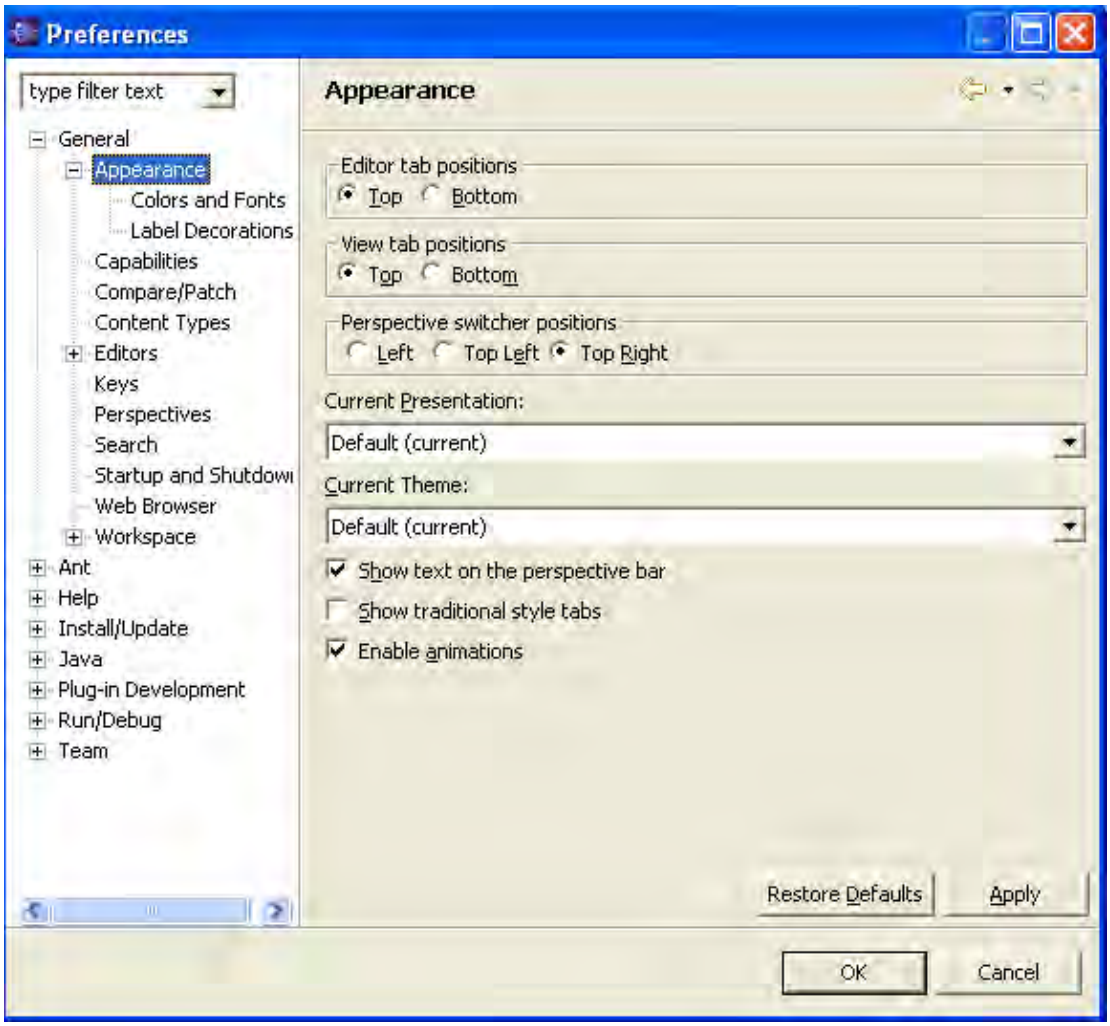
Ant Editor
Ant preferences
Ant runtime preferences

Appearance

The following preferences can be changed on the Appearance page.

Option	***Description***	***Default***
Editor tab positions	Specify either top or bottom to indicate where you want tabs for stacked editors to appear.	Top
View tab positions	Specify either top or bottom to indicate where you want tabs for stacked views to appear.	Top
Perspective switcher positions	Specify the location of the perspective switcher bar	Top right
Current presentation	Specify the currently active presentation (look and feel).	Default (current)
Current theme	Specify the currently active theme (color and font set).	Default (current)
Show text on perspective bar	Specify whether labels should be shown in the perspective bar as well as icons.	Enabled
Show traditional style tabs	Specify whether traditional (square) tabs should be used in place of the curved tabs	Disabled
Enable animations	Enable/disable the feature where views animate to their location when closed or opened	Enabled

Here is what the Appearance preferences page looks like:

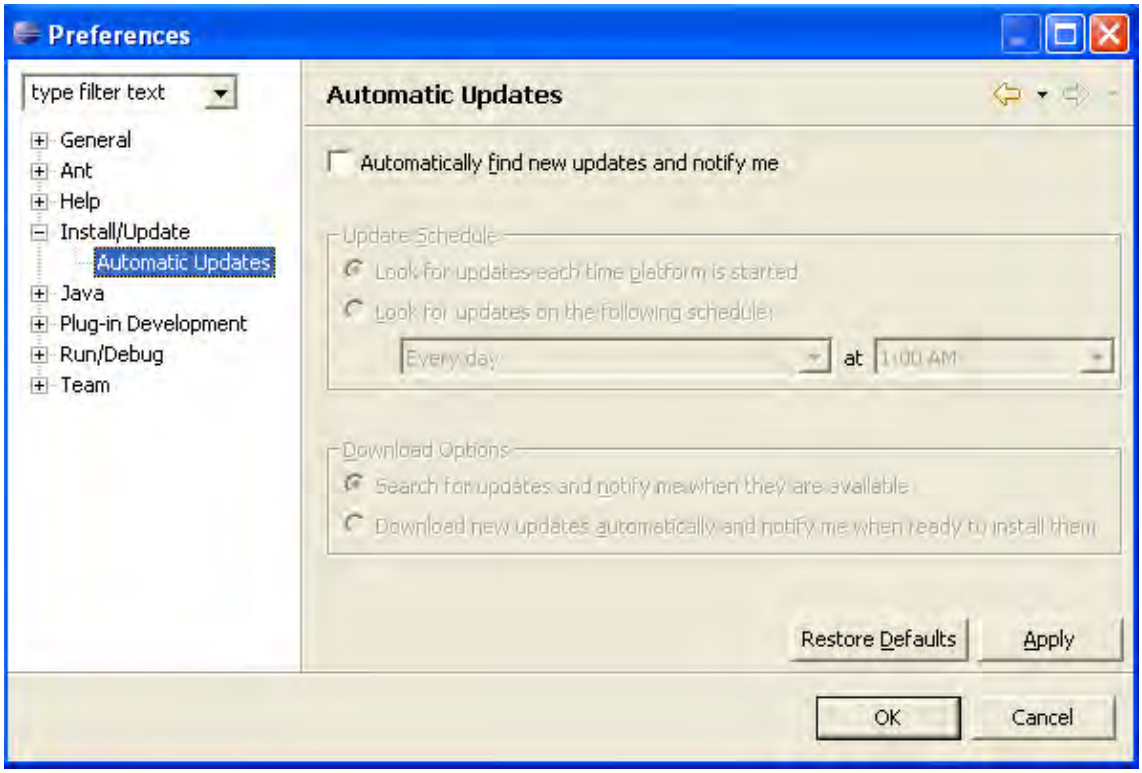

Preferences
type filter text
General
Appearance
Colors and Fonts
Label Decorations
Capabilities
Compare/Patch
Content Types
Editors
Keys
Perspectives
Search
Startup and Shutdowi
Web Browser
Workspace
Ant
Help
Install/Update
Java
Plug-in Development
Run/Debug
Team
Appearance
Editor tab positions
Top
Bottom
View tab positions
Top
Bottom
Perspective switcher positions
Left
Top Left
Top Right
Current Presentation:
Default (current)
Current Theme:
Default (current)
Show text on the perspective bar
Show traditional style tabs
Enable animations
Restore Defaults
Apply
OK
Cancel

Automatic Updates

The following preferences can be changed on the Automatic Updates page:

Option	Description	Default
Automatically find new updates and notify me	When selected, Update manager will automatically search for update, as defined by the update schedule	Off
Update Schedule	Look for updates on each startup, or once a day or some day a week, at a predefined time.	on startup
Download Options	This option allows you to choose between having Eclipse search for updates and notifying you of them once they are available or having Eclipse automatically download new updates and asking you to install them.	on startup

Here is what the Automatic Updates preference page looks like:



Related tasks

Updating Eclipse with the update manager

Related reference

Help Menu

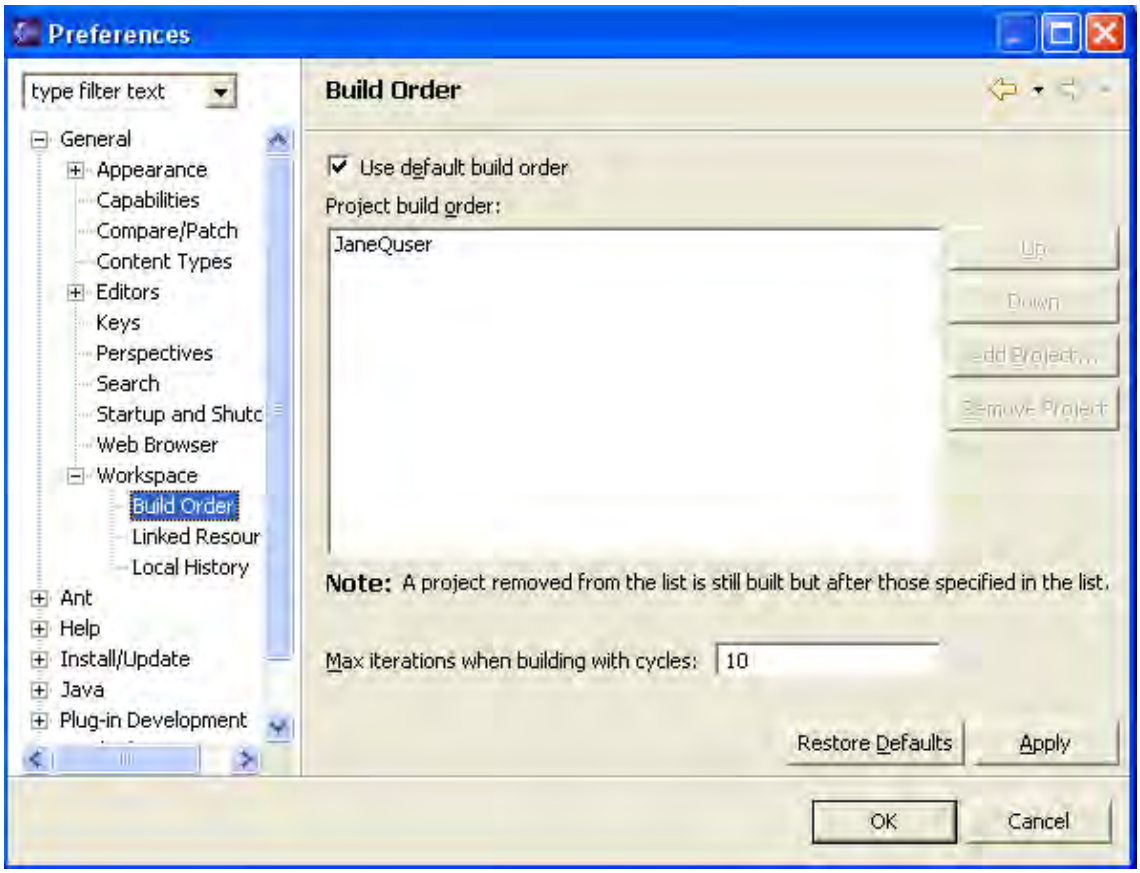
Build Order

Often the order in which projects are built is important. For example, if one project requires the Java classes which were defined in another project, the first project must be built after its prerequisite classes have been built. The Workbench allows users to explicitly define the order in which projects are built. Alternatively, users can let the platform compute the build order by interpreting project references as prerequisite relationships. The build order is applied for both building the entire workspace or for a group of projects.

You can change this order on the Build Order preferences page.

Option	Description	Default
Use default builder order	This option allows the platform to computes the build ordering. Turning off this option enables access to the projects list, the ordering of which can be manipulated.	On
Project build order	This option allows you to select projects and use the **Up** and **Down** buttons to change the build order. Add and remove projects in the build order using the **Add Project** and **Remove Project** buttons. Projects removed from the build order *will* be built, but they will be built after all projects in the build order are built.	
Max iterations when building with cycles	This preference allows you to deal with build orders that contain cycles. Ideally, you should avoid cyclic references between projects. Projects with cycles really logically belong to a single project, and so they should be collapsed into a single project if possible. However, if you absolutely must have cycles, it may take several iterations of the build order to correctly build everything. Changing this preference will alter the maximum number of times the workbench will attempt to iterate over the build order before giving up.	10

Here is what the Build Order preference page looks like:



Related reference

Builds
Project menu

Builds

Builders create or modify workspace resources, usually based on the existence and state of other resources. They are a powerful mechanism for enforcing the constraints of some domain. For example, a Java builder converts Java source files (.java files) into executable class files (.class files), a web link builder updates links to files whose name/location have changed, etc. As resources are created and modified, builders are run and the constraints are maintained. This transform need not be one to one. For example, a single .java file can produce several .class files.

Auto−build vs. Manual Build

There are two distinct user work modes with respect to building: relying on Auto−build, or user initiated manual building.

Users who do not need fine−grained control over when builds occur may simply choose to turn on auto−building. With auto−building on, builds occur after every set of resource changes (e.g., saving a file, importing a ZIP, ...). Auto−building is efficient because the amount of work done is proportional to the amount of change done. The benefit of auto−building is that your derived resources (e.g., Java .class files) are always up to date. Auto−building is turned on/off via the **Build automatically** option on the **General > Workspace** preference page.

Users needing more control over when builds occur can turn off auto−building and manually invoke builds. This is sometimes desirable in cases where, for example, you know building is of no value until you finish a large set of changes. In this case there is no benefit to paying the cost of auto−building. Builds can be invoked manually in numerous ways, for example, by selecting **Build Project** from a project's context menu.

The disadvantage of manual building is that the problems that were generated to indicate build errors quickly become out of date until you build. In addition, it is very important that you remember to manually build before relying on build output (e.g. before running your Java program).

Building and Cleaning

Builds work incrementally based on a previous built state. They will apply the transforms of the configured builders only on the resources that have changed since that previous state was computed (i.e., since the last build). Auto−building always uses incremental building for efficiency.

A clean build (**Project > Clean**) discards any existing built state. The next build after a clean will transform all resources according the domain rules of the configured builders.

Depending on the user's needs, build and clean can be done over a specific set of projects or the workspace as a whole. Specific files and folders cannot be built separately.

Project menu

The Project menu allows you to perform actions (builds or compilations) on projects in the Workbench.

Open Project

This command opens the currently selected project or projects. The selected projects must currently be closed for this command to be available.

Close Project

This command closes the currently selected project or projects. The selected projects must be currently open for this command to be available. Closing a project will remove all of that project's state from memory, but the contents on disk are left untouched.

Build All

This command performs an incremental build on all projects in the Workbench. That is, it builds (compiles) all resources in the Workbench that are affected by any resource changes since the last incremental build. This command is only available if auto−build is turned off. Auto−build is turned off via the ***Build Automatically*** menu option or from the **General > Workspace** preference page.

Build Project

This command performs an incremental build on the currently selected project. That is, it builds (compiles) all resources in the project that are affected by any resource changes since the last build. This command is only available if auto−build is turned off. Auto−build is turned off via the ***Build Automatically*** menu option or from the **General > Workspace** preference page.

Build Working Set

This menu allows you to performs an incremental build on a working set. That is, it builds (compiles) all resources in the working set that are affected by any resource changes since the last build. This command is only available if auto−build is turned off. Auto−build is turned off via the ***Build Automatically*** menu option or from the **General > Workspace** preference page.

Clean

This command discards all previous build results. If autobuild is on, then this invokes a full build.

Build Automatically

This command allows you to toggle the auto build preference. The auto−build preference is also located on the **General > Workspace** preference page.

Properties

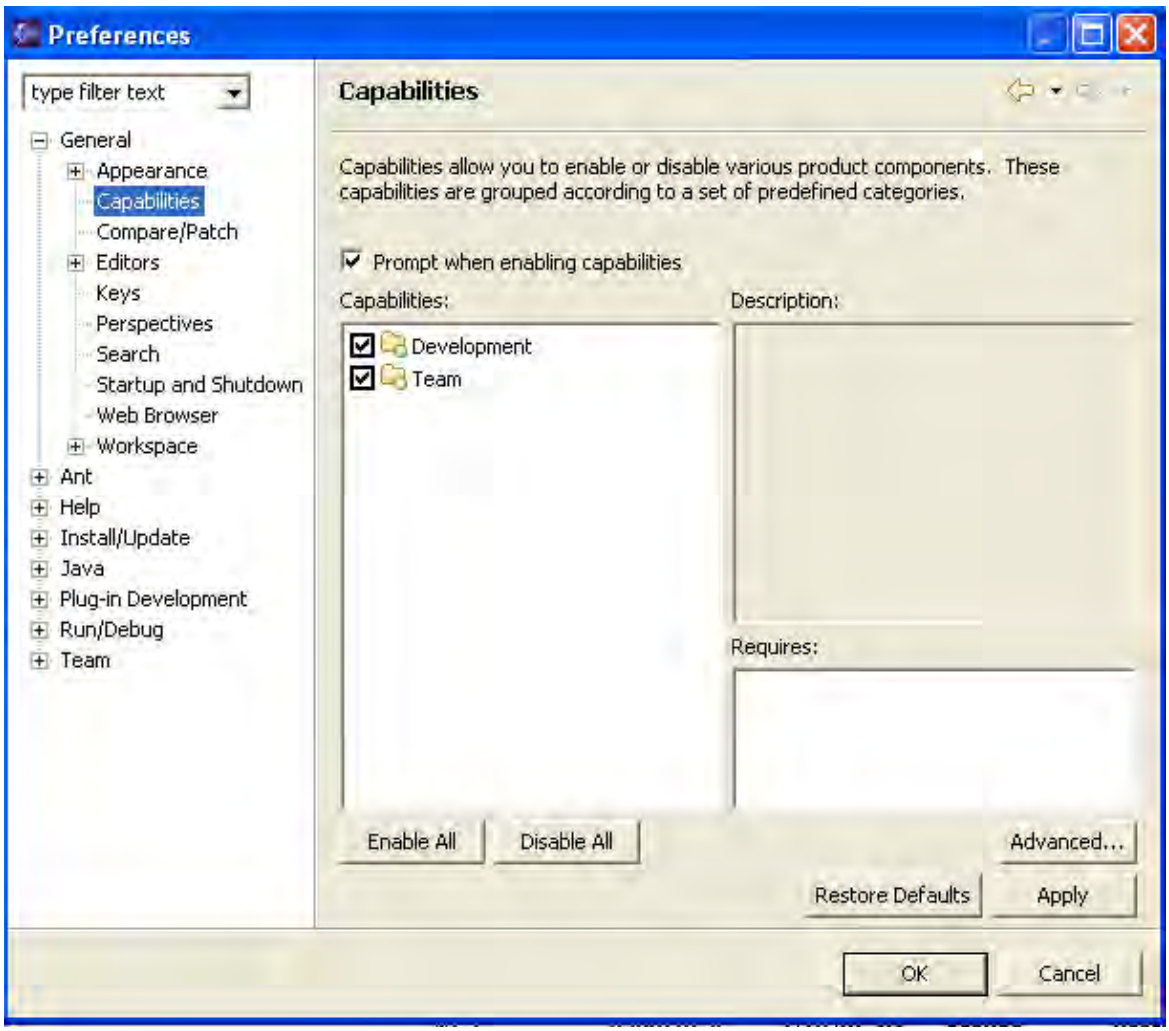
This command opens a dialog showing the properties of the selected project or of the project that contains the selected resource.

Capabilities

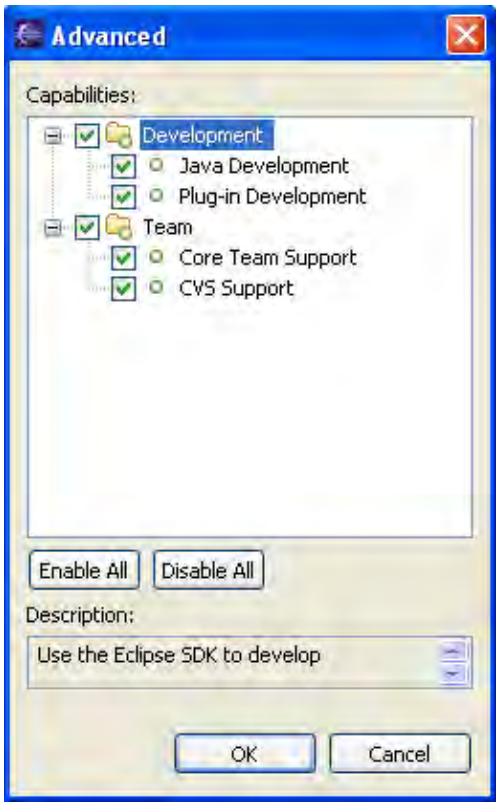
The Capabilities preference page allows you to enable or disable various product components such as Java development and plug−in development. By default, the page only shows general categories of behavior. If you would like configure fine−grained capabilities you should use the "Advanced" dialog.

Note: Some capability selections have dependencies on other capabilities, disabling a required capability while leaving dependant capabilities enabled will only result in them becoming re−enabled. This is the case when deselecting Java Development and Core Team Support.

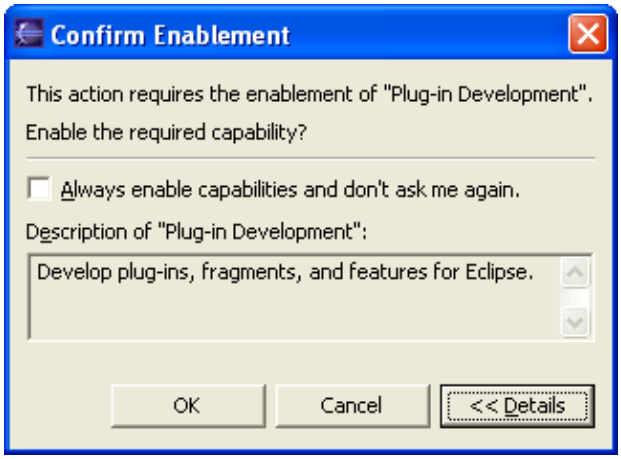
Here is what the Capabilities preference page looks like:



The advanced dialog appears as follows:



When attempting to enable an action after its capability has been disabled or has yet to be enabled in the preferences page, the following ***Confirm Enablement*** prompt will appear verifying that you do indeed want to enable the required capability. Click ***Details*** to display a description of the capability.



Note: This dialog only appears if the "Prompt when enabling capabilities" preference has not been disabled.

Colors and Fonts

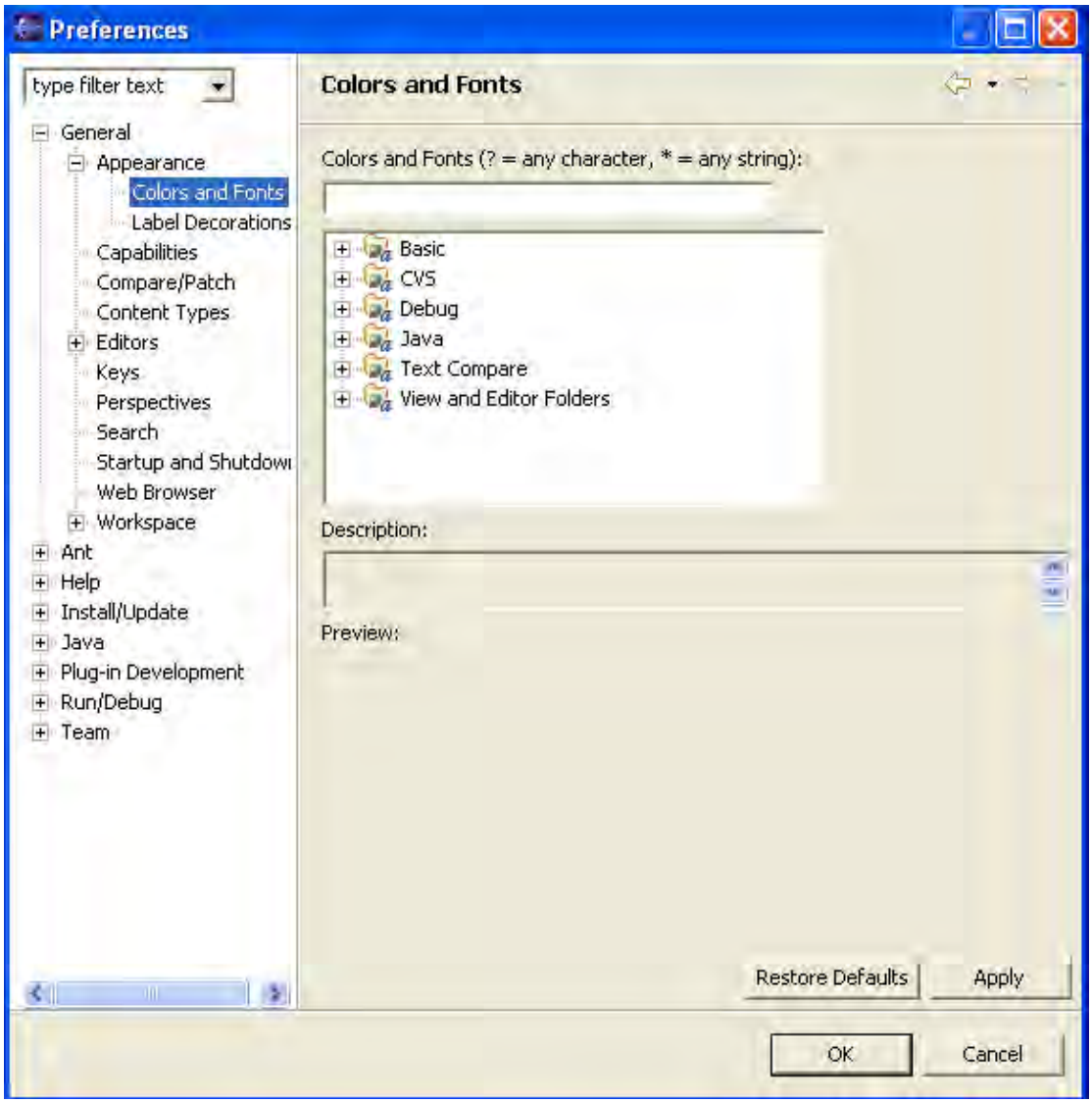
Many of the fonts and colors and used by eclipse components can be set using the Colors and Fonts preference page.

A tree is used to navigate among and show a short preview of the various colors and fonts. The current face (but not size) of any font is previewed in its label. Colors are previewed in the icon associated with its label. Additionally, some categories (Workbench in particular) provide a more detailed preview of their contributions. This preview is shown below the description area if available.

Font settings can be changed either by selecting the font from the list and clicking ***Use System Font*** to choose the Operating System font setting or by clicking ***Change*** to open up a font selection dialog. ***Reset*** can be used to return to the default value.

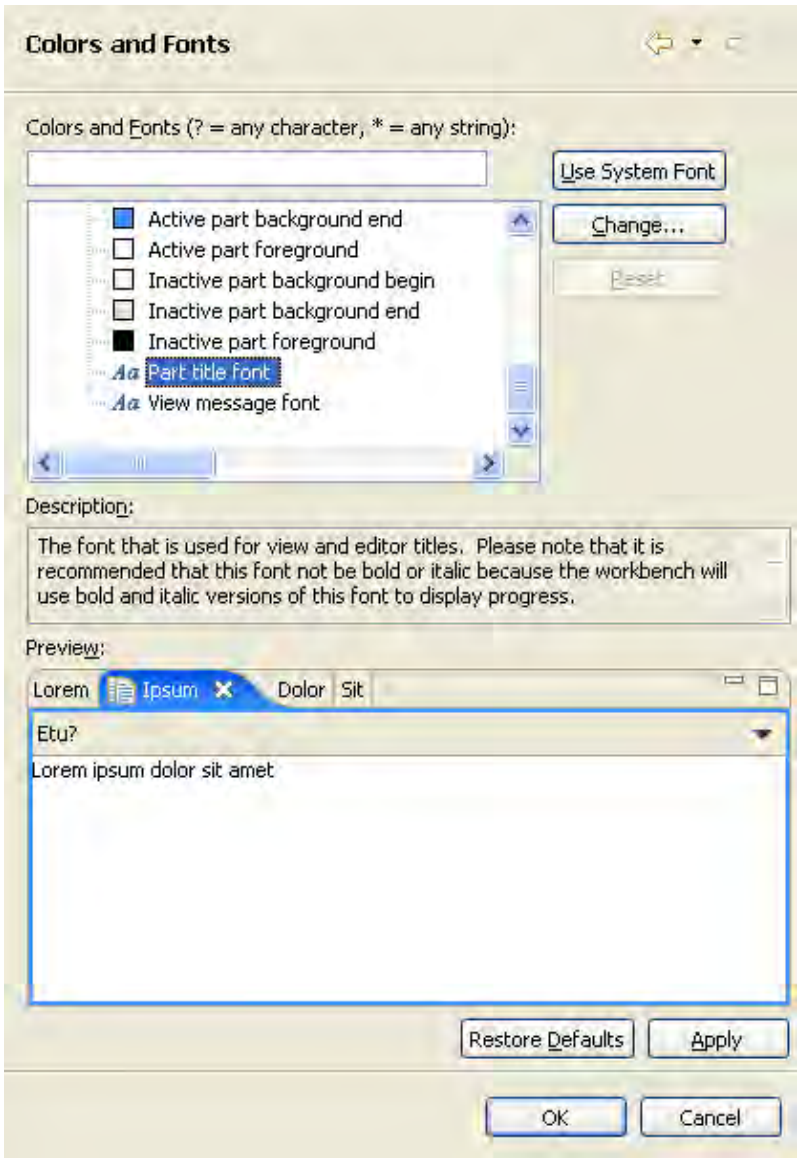
Color settings can be changed by clicking ***color*** to the right of the tree area when a color is selected. ***Reset*** can be used to return to the default value.

Here is what the Colors and Fonts preference page looks like:



The Colors and Fonts text field can be used to filter the contents. Simply type in an entry and any matching results will remain in the tree view.

Descriptions and previews are provided when the Workbench colors and font settings are selected.



Related tasks

Changing fonts and colors

Compare/Patch

The following preferences can be changed on the Compare/Patch page.

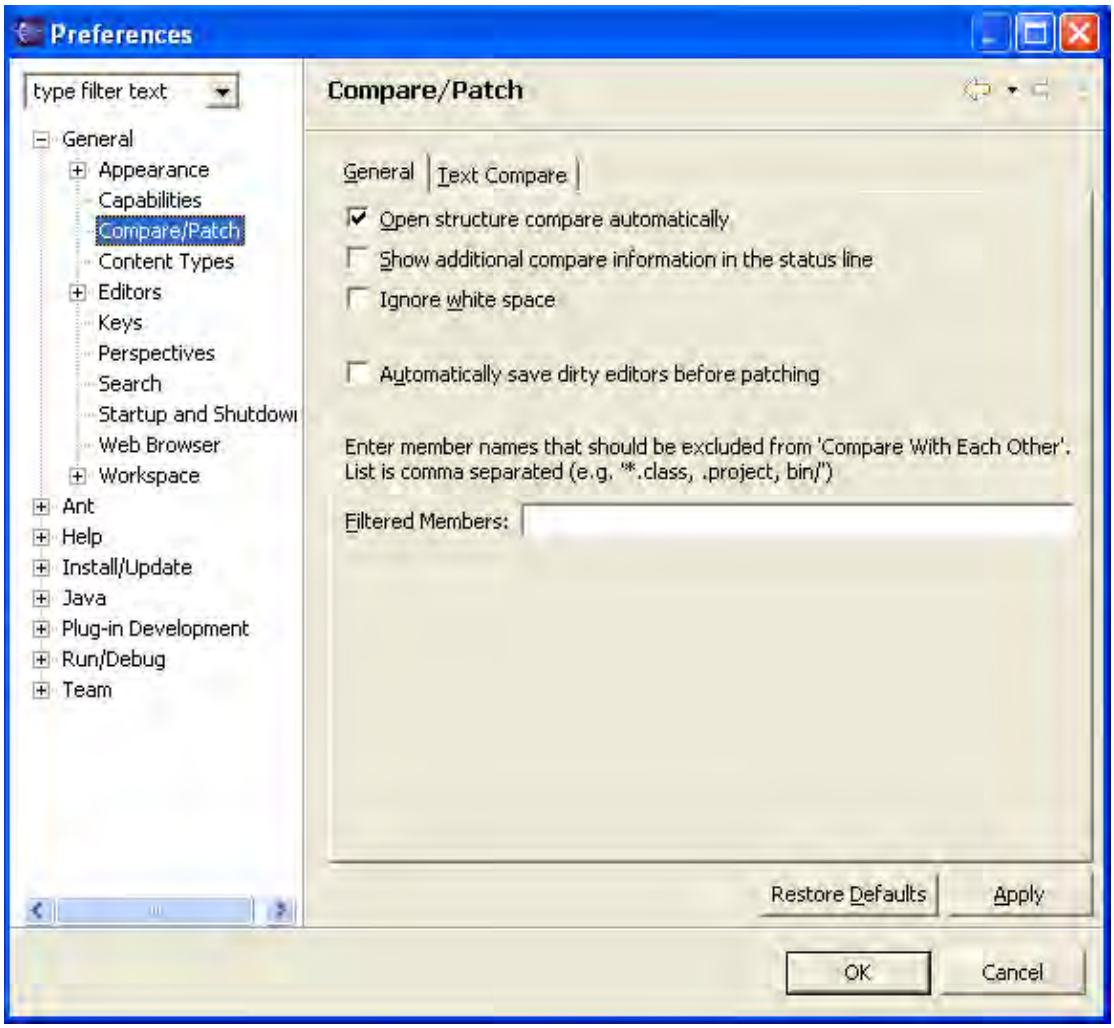
General options

Option	Description	Default
Open structure compare automatically	This option controls whether a structure compare is automatically performed whenever a content compare is done. Turn this option off if you don't want to see the structural differences.	On
Show additional compare information in the status line	If this option is on, additional information about a change is shown in the status line. Turn this option on if you are interested in additional information about a change.	Off
Ignore white space	This option controls whether or not whitespace change are shown in the compare viewer. Turn this option on if you want to see changes in whitespace.	Off
Automatically save dirty options before patching	This option controls whether any unsaved changes are automatically saved before a patch is applied. Turn this option on if you want to save changes automatically.	Off
Filtered Members	This option allows you to filter members that should be excluded from 'Compare With Each Other'. ***Note:*** The names in the list must be separated by a comma.	

Text Compare options

Option	Description	Default
Synchronize scrolling between panes in compare viewers	The two comparison viewers will "lock scroll" along with one another in order to keep identical and corresponding portions of the code in each pane side−by−side. Turn this option off if you do not want the compare viewers to lock scroll.	On
Initially show ancestor pane	Sometimes you want to compare two versions of a resource with the previous version from which they were both derived. This is called their *common ancestor*, and it appears in its own comparison pane during a three way compare. Turn this option on if you want the ancestor pane to always appear at the start of a comparison.	Off
Show pseudo conflicts	Displays pseudo conflicts, which occur when two developers make the same change (for example, both add or remove the exact same line of code or comment). Turn this option on if you want pseudo conflicts to appear in compare browsers.	Off
Connect ranges with single line	Controls whether differing ranges are visually connected by a single line or a range delimited by two lines.	On

Here is what the Compare preference page looks like:



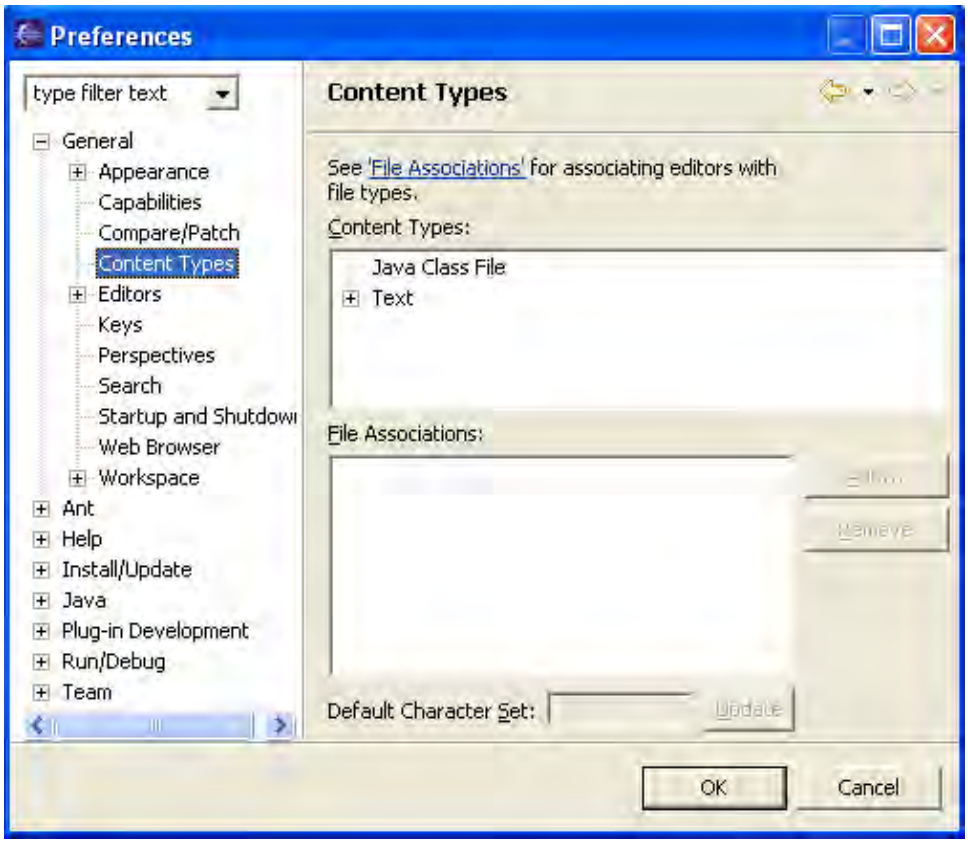
Related reference

Compare Editor
CVS Synchronization View

Content Types

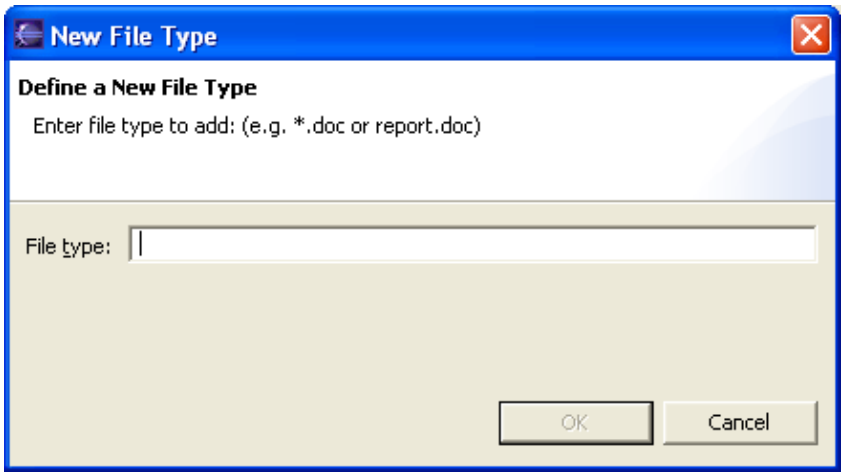
The Content Types preference page allows you to edit content types, their associated file names and character sets as well as associate arbitrary file names or file extensions with content types. A content type acts as a description of a certain class of files (for instance, XML files). Eclipse uses this description in various scenarios such as editor look up and file comparing.

To access the Content Types preference page select ***Window*** > ***Preferences*** > ***General*** > ***Content Types***. Here is what the Content Types preference page looks like:

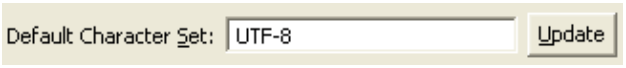


By selecting a content type in the topmost tree you can alter the file names and extensions that are associated with it. ***Note***: Certain items will be marked as "locked". An item is locked if it is one of the associations provided by the plug−in that declares the content type. In other words, only user−contributed associations may be removed.

Adding an association is as simple as clicking ***Add...***. A dialog will be shown prompting you to enter the file name or extension.



In addition to adding and removing file names or extensions, you may also set the default character set for a given content type. To do this, simply enter the character set name in the provided field and click ***Update***.

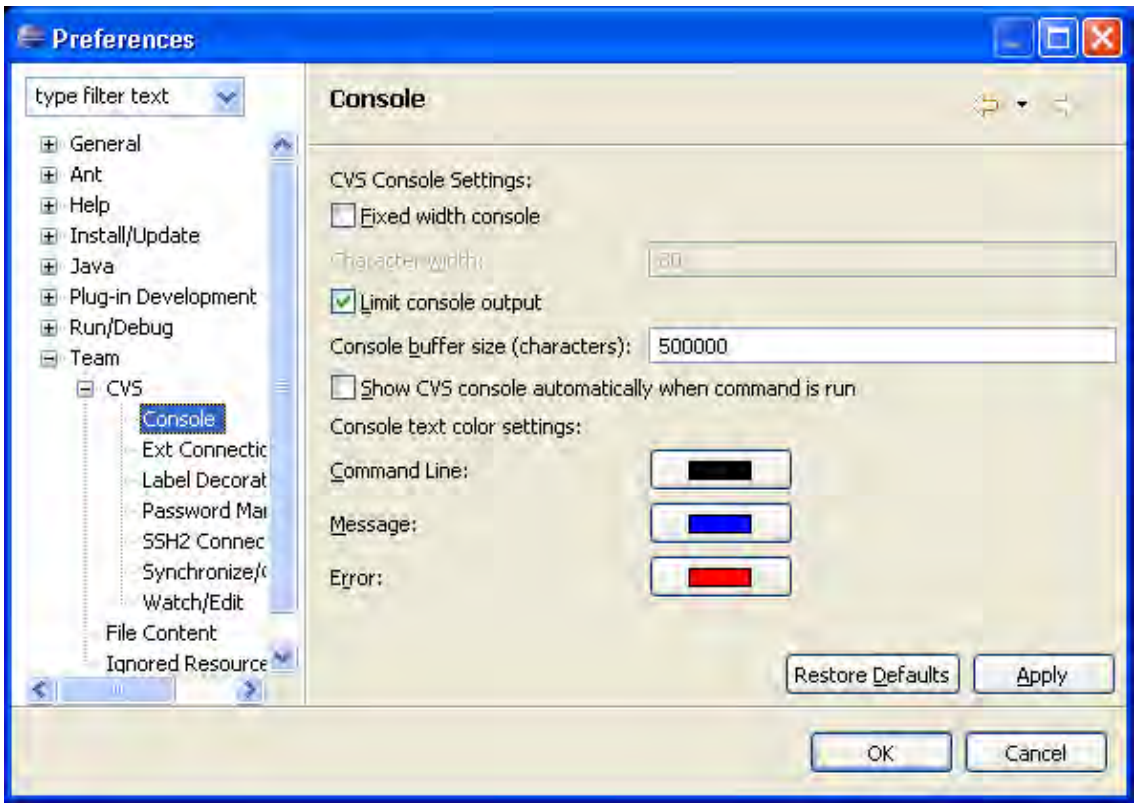


CVS Console

The following preferences can be changed on the CVS Console preference page.

Option	Description	**Default**
Fixed width	Use this option to fix the width of console lines. Enabling this option allows the width to be specified. The default width is 80.	Disabled
Limit console output	Use this option to limit the number of characters to be buffered by the console. The default buffer size is 500000.	Enabled
Show CVS output in Console view	Use this option to show the output of CVS commands in the Console view. Enabling this option may show useful information but will slow down command operation.	Disabled
Console text color settings	Use these options to change the colors for the text shown in CVS Console. • Command line text *(black)* • Message text *(blue)* • Error text *(red)*	

Here is what the preference page looks like:

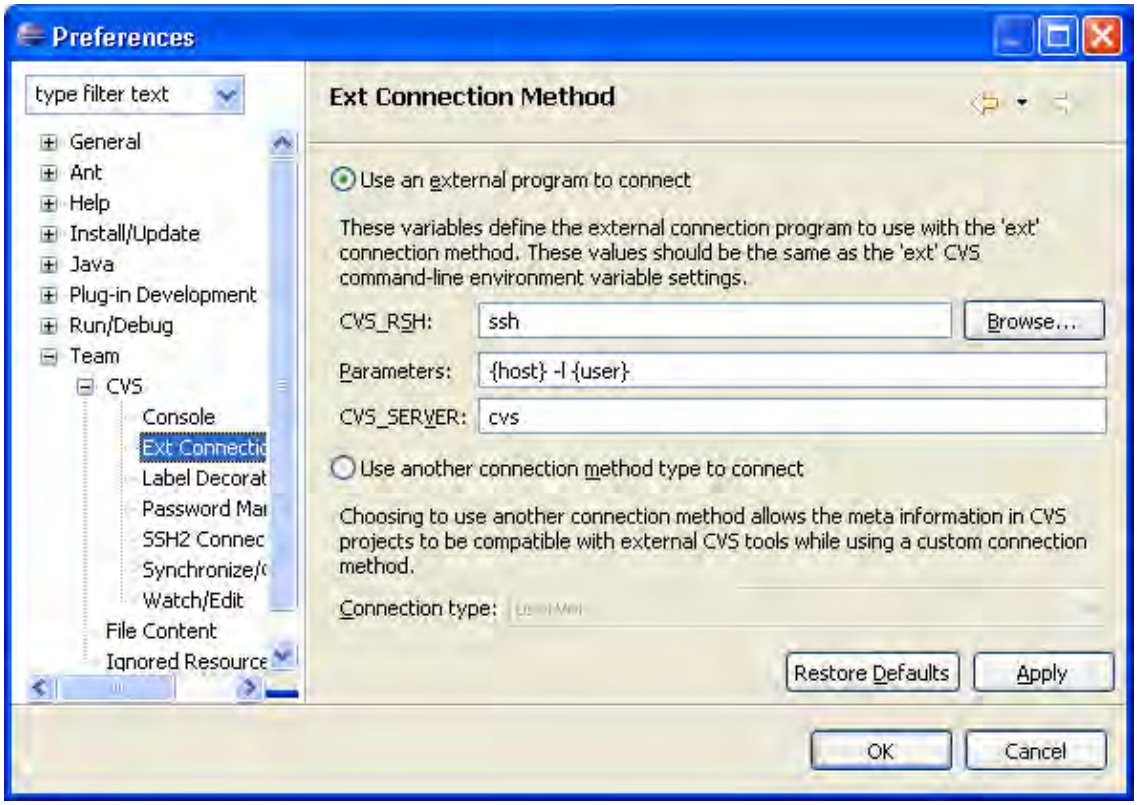


CVS Ext Connection Method

The following preferences can be changed on the CVS Ext Connection Method preference page.

Use external program vs. Use internal connection method	This page allows you to configure the ext connection method to use an external program or another connection method to to connect to a server. The later option is provided to allow custom connection method such as extssh to remain compatible with external CVS client tools.	Use external program
CVS_RSH	Use this option to configure the program that will be called to connect to the remote CVS server. The RSH command is invoked with following calling pattern: *CVS_RSH Parameters CVS_SERVER*	ssh
Parameters	Use this option to configure the parameters passed to the CVS_RSH program. The default parameter pattern is {host} −l {user}. It can be tailored using the {host}, {user}, {password} and {port} variables.	{host} −l {user}
CVS_SERVER	Use this option to configure the name of the remote CVS server program to run. Change this setting only if the remote CVS server binary name is different than the default.	cvs
Connection type	Use this option to set the connection method to be used for repository locations that use the ext connection method, if the option to use another connection method is enabled.	

Here is what the preference page looks like:

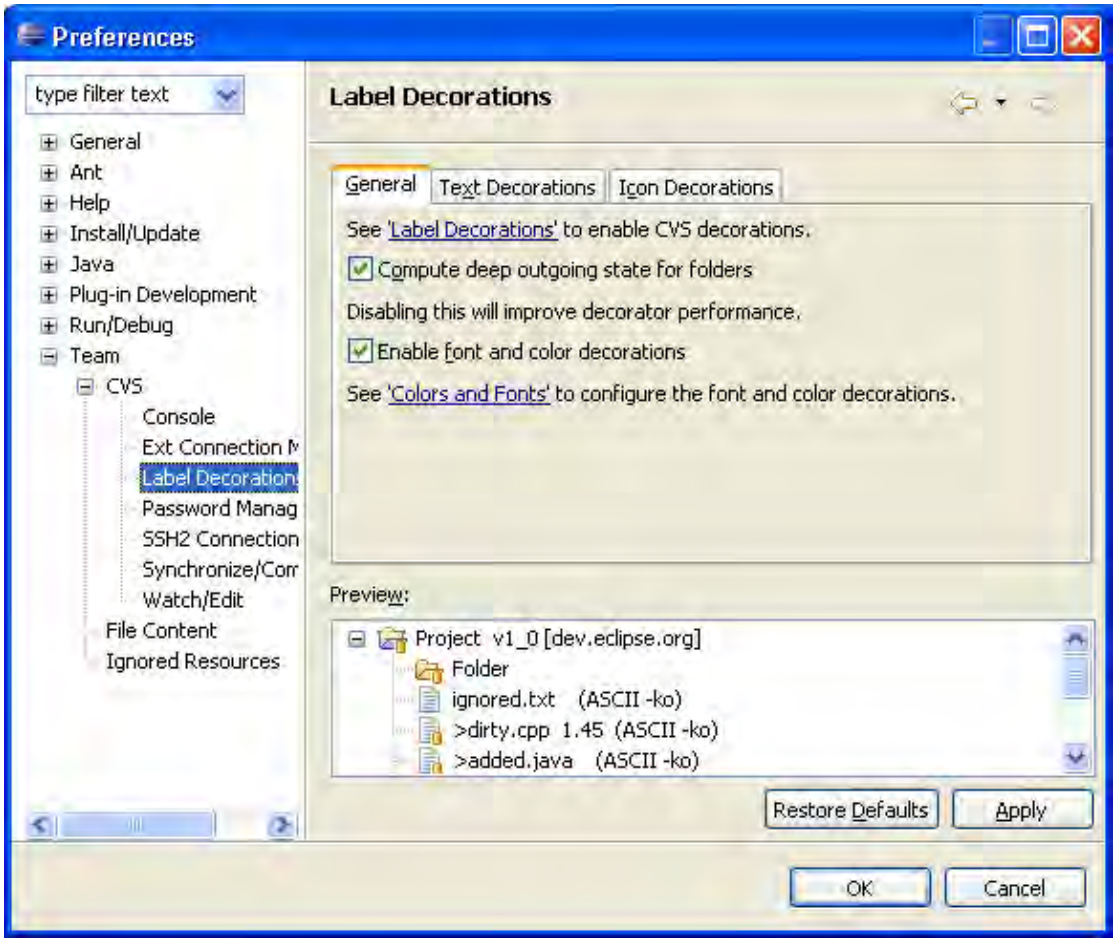


CVS Label Decorations

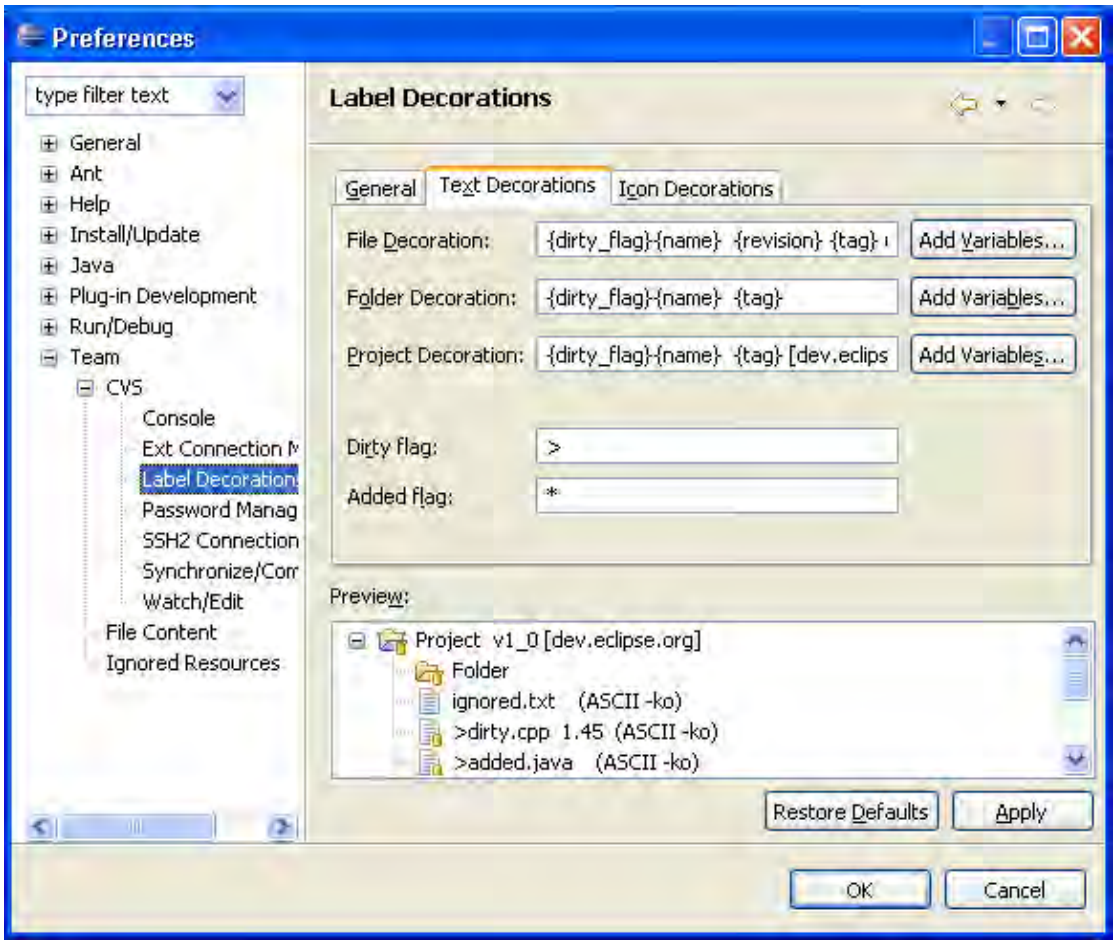
The following preferences can be changed on the CVS Label Decorations preference page.

General	Use the options on this page to configure general preferences about the decorators: • ***Compute deep outgoing state for folders:*** Use this option to configure if the outgoing indicators on folders should be calculated. Disabling this option improves the performance of the decorators because calculating the dirty state for folders requires computing the dirty state for all child resources. (*enabled by default*) • ***Enable font and color decorations:*** Use this option to enable font and color decorations. The colors and fonts used for outgoing changes, ignored resources, etc. can be configured on the ***General > Appearance >Colors and Fonts*** preference page
Text	Use the options on this page to configure how CVS information will be added to text labels.
Icons	Use the options on this page to configure the which icons can be used as overlays to show CVS specific information in views.

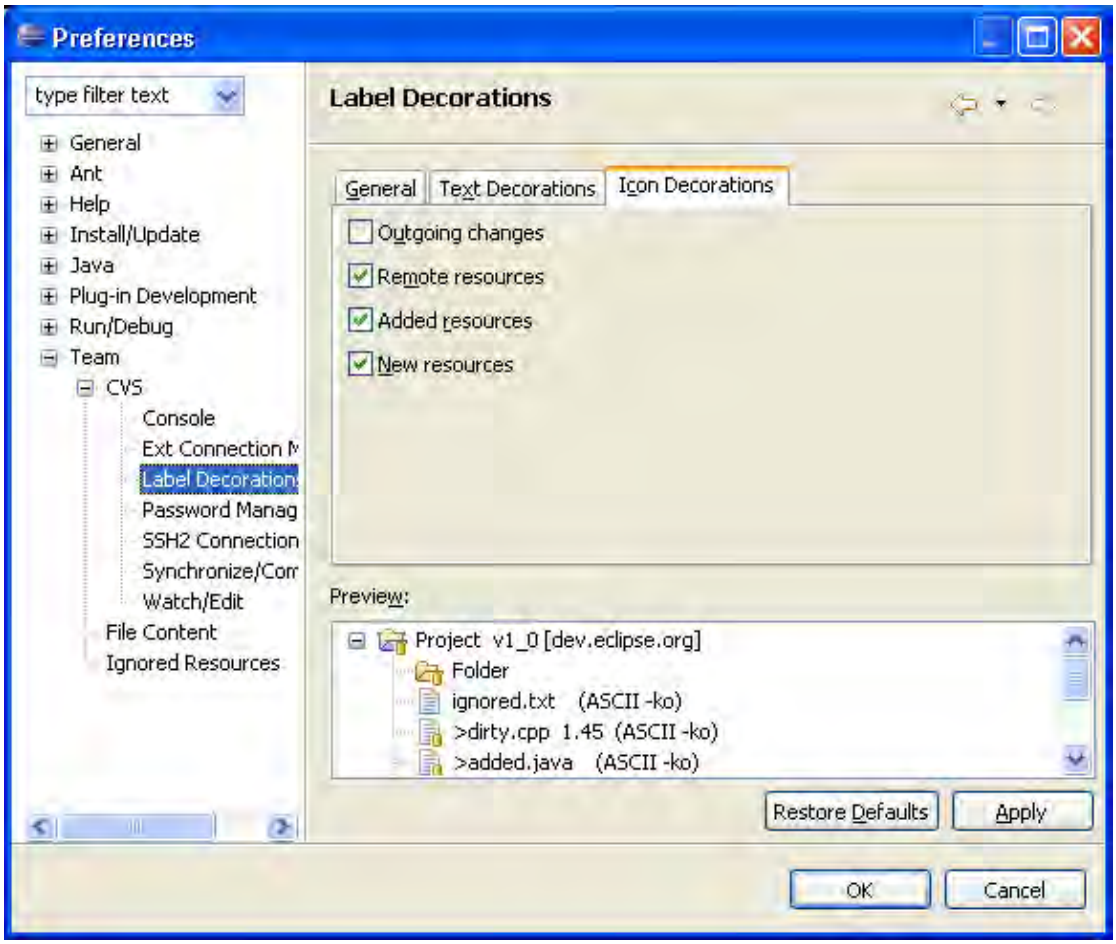
This is what the General tab group of the CVS Label Decorations preference page looks like:



This is what the Text tab group of the CVS Label Decorations preference page looks like:

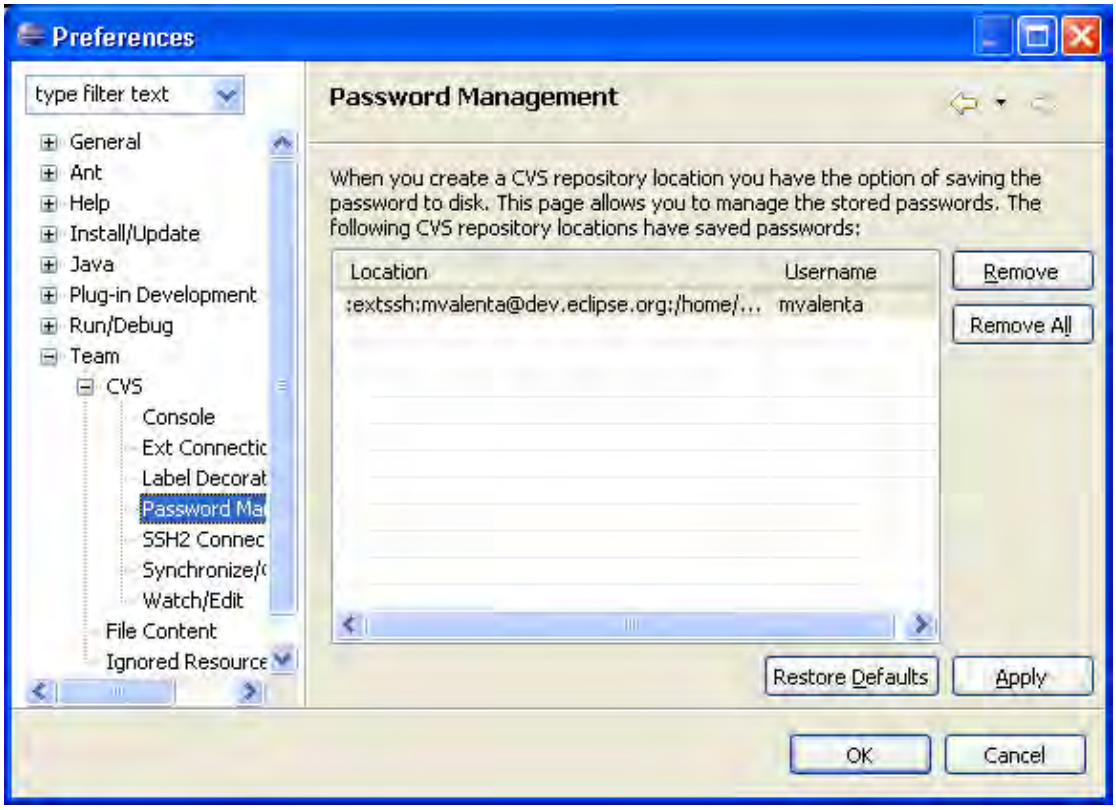


This is what the Icons tab group of the CVS Label Decorations preference page looks like:


Preferences
type filter text
General
Ant
Help
Install/Update
Java
Plug-in Development
Run/Debug
Team
CVS
Console
Ext Connection M
Label Decoration
Password Manag
SSH2 Connection
Synchronize/Com
Watch/Edit
File Content
Ignored Resources
Label Decorations
General
Text Decorations
Icon Decorations
Outgoing changes
Remote resources
Added resources
New resources
Preview:
Project v1_0 [dev.eclipse.org]
Folder
ignored.txt (ASCII -ko)
>dirty.cpp 1.45 (ASCII -ko)
>added.java (ASCII -ko)
Restore Defaults
Apply
OK
Cancel

CVS Password Management

This preference page allows you to see which repository locations have passwords cached in the keyring file and also allows you to purge those passwords.Only those passwords that you have explicitly indicated should be saved will appear in the list. Here is what the preference page looks like:

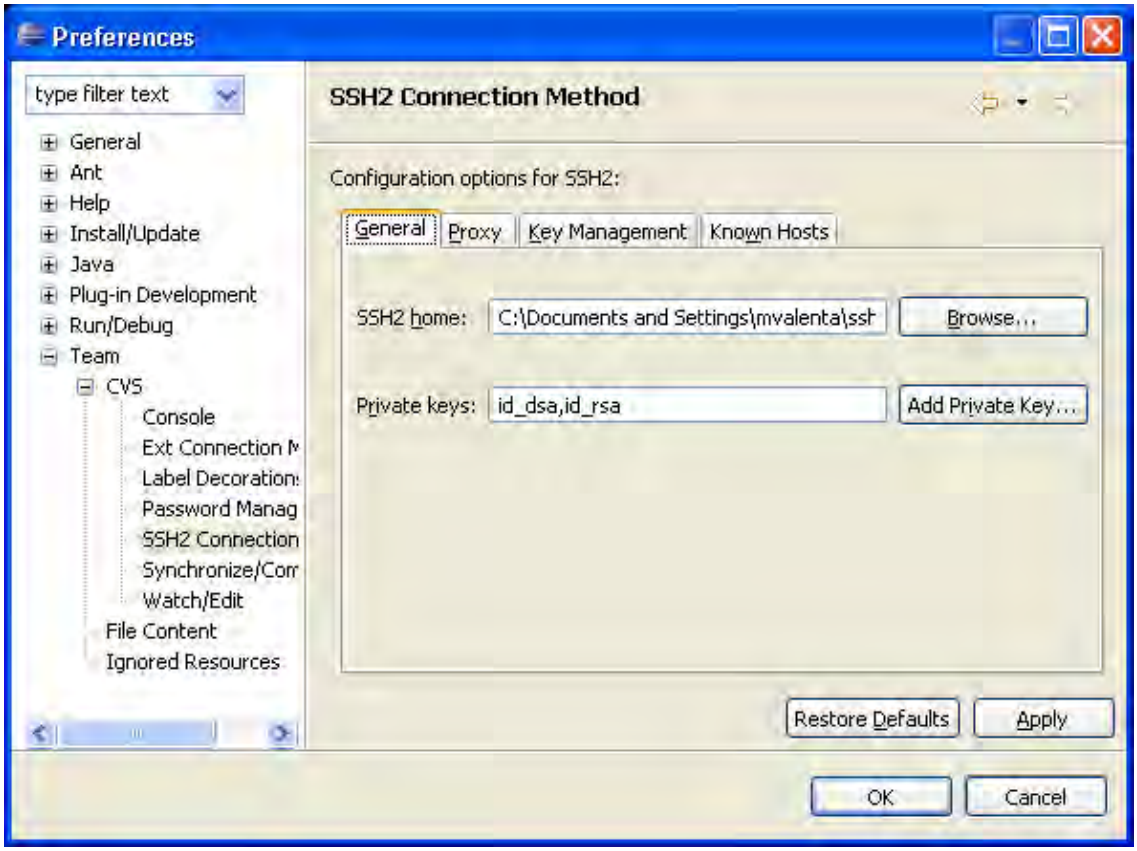


CVS SSH2 Connection Method

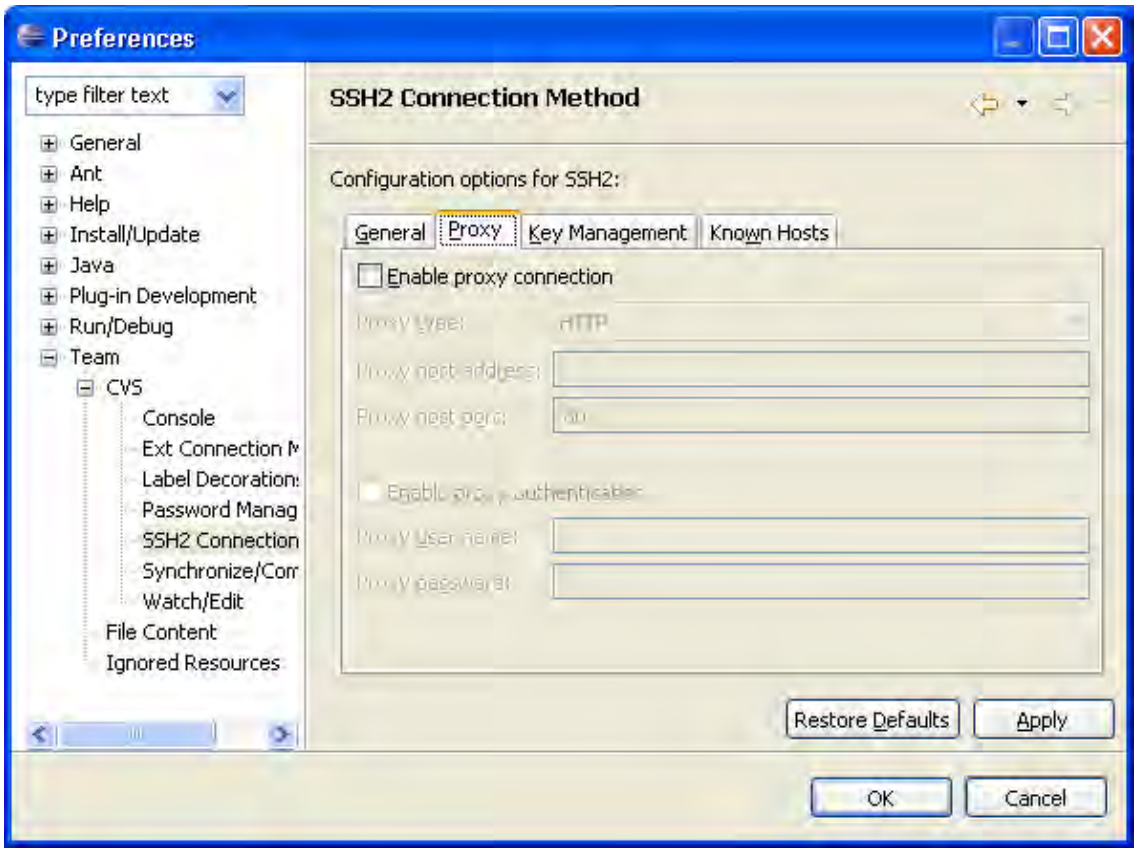
The preferences on the CVS SSH2 Connection Method preference page are divided into 4 groups.

General	Use the options in this tab group to configure where the ssh key directory is and what keys are to be sent to a server when connecting.
Proxy	Use the options in this tab group to configure an HTTP or SOCKS5 proxy.
Key Management	Use the options in this tab group to create, manage and export keys.
Known Hosts	Use the options in this tab group to manage the keys for the known hosts.

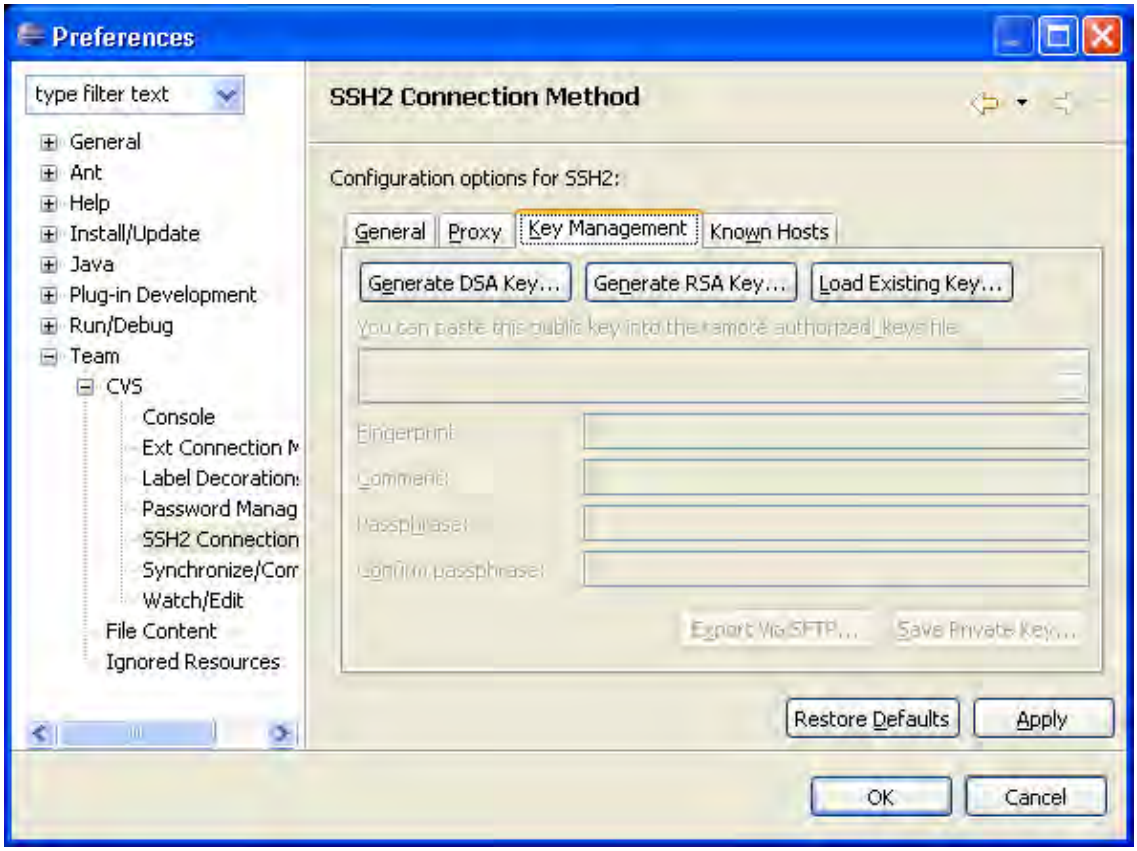
Here's what the General tab of the CVS SSH2 preference page looks like:



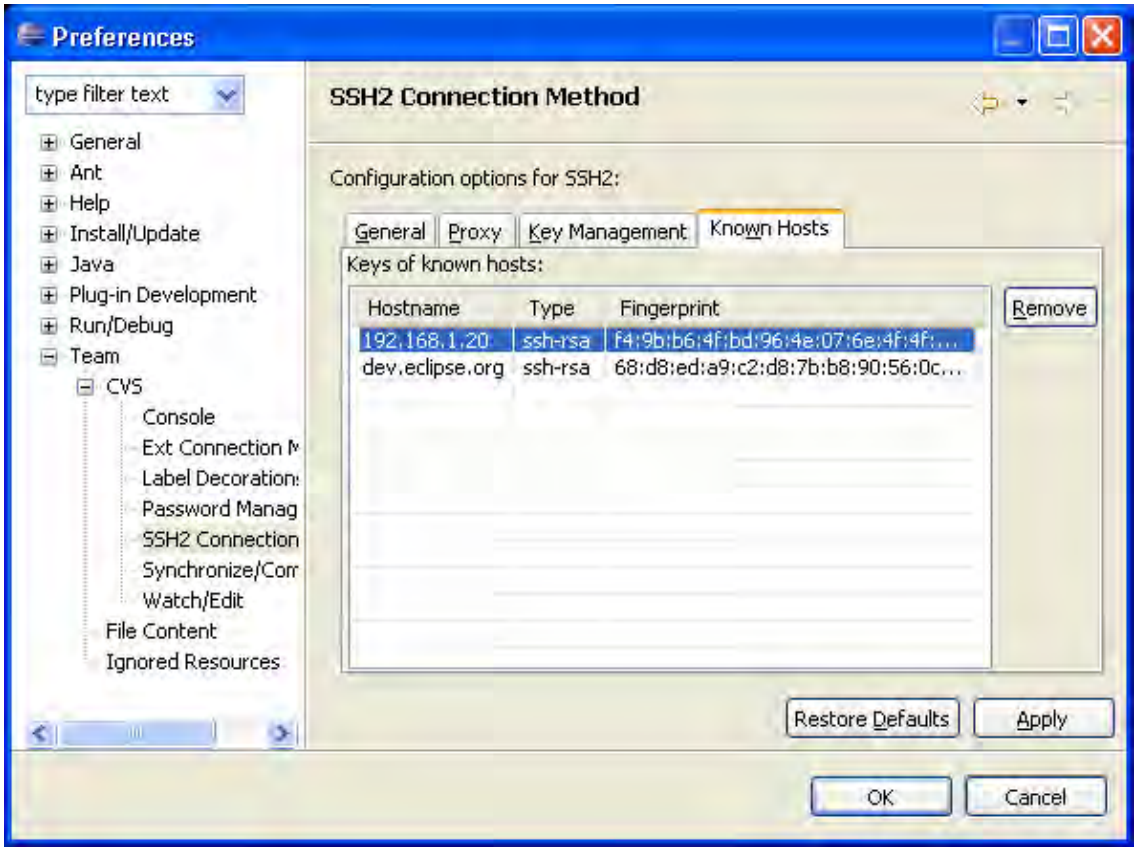
Here's what the Proxy tab of the CVS SSH2 preference page looks like:



Here's what the Key Management tab of the CVS SSH2 preference page looks like:



Here's what the Known Hosts tab of the CVS SSH2 preference page looks like:

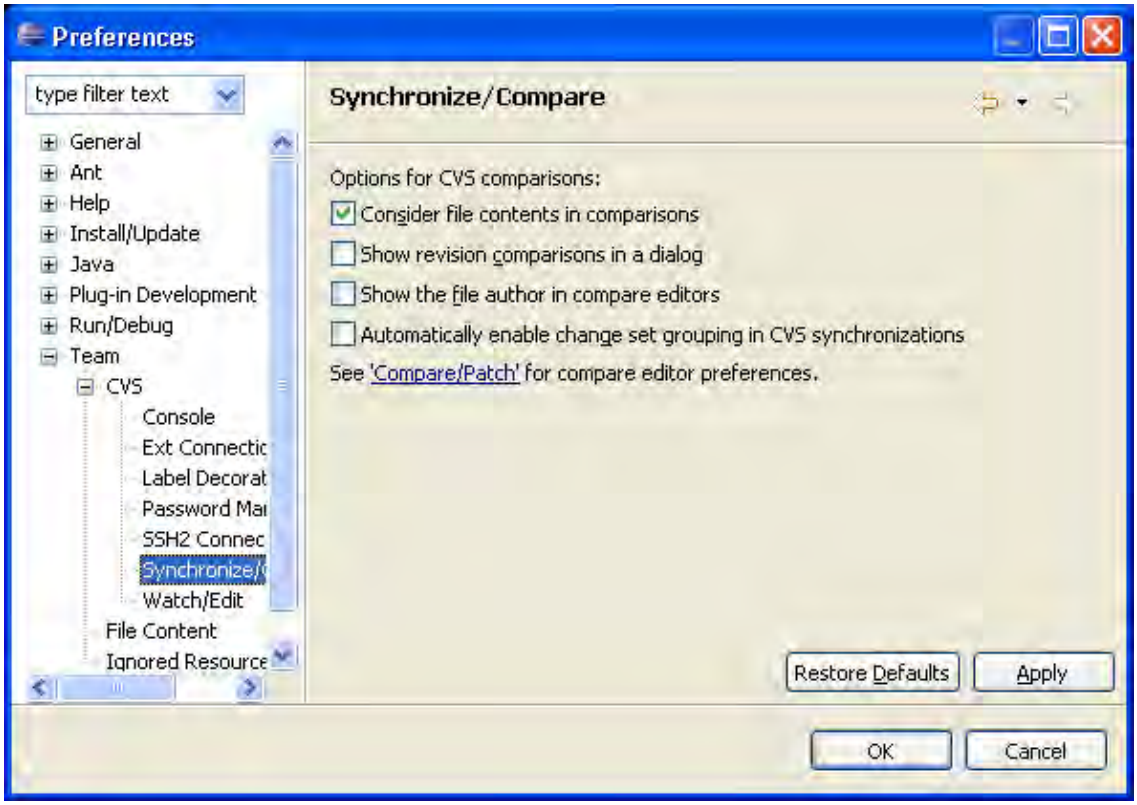


CVS Synchronize/Compare

The following preferences can be changed on the CVS Synchronize/Compare preference page.

Option	Description	Default
Consider file contents in comparisons	Use this option to compare contents for changed files found when comparing CVS resources. Usually, time stamps are used to compare CVS files, and this is by far the fastest method. However, in some cases a more accurate comparison can be achieved by comparing file content. Disabling this option will speed up comparisons but may result in compare entries whose contents are the same. This option only applies to comparisons and merges but not Workspace synchronizations.	Enabled
Show revision comparisons in dialog	Use this this option to show revision comparisons in a dialog instead of a compare editor.	Disabled
Show the file author in compare editors	Use this option to show the author of file revisions when shown in compare editors and dialogs.	Disabled
Enable change set grouping in CVS synchronizations	Use the option to have change sets enabled by default in those CVS synchronizations that support them.	Disabled

Here is what the preference page looks like:



Related reference

Team
CVS Workspace Synchronization

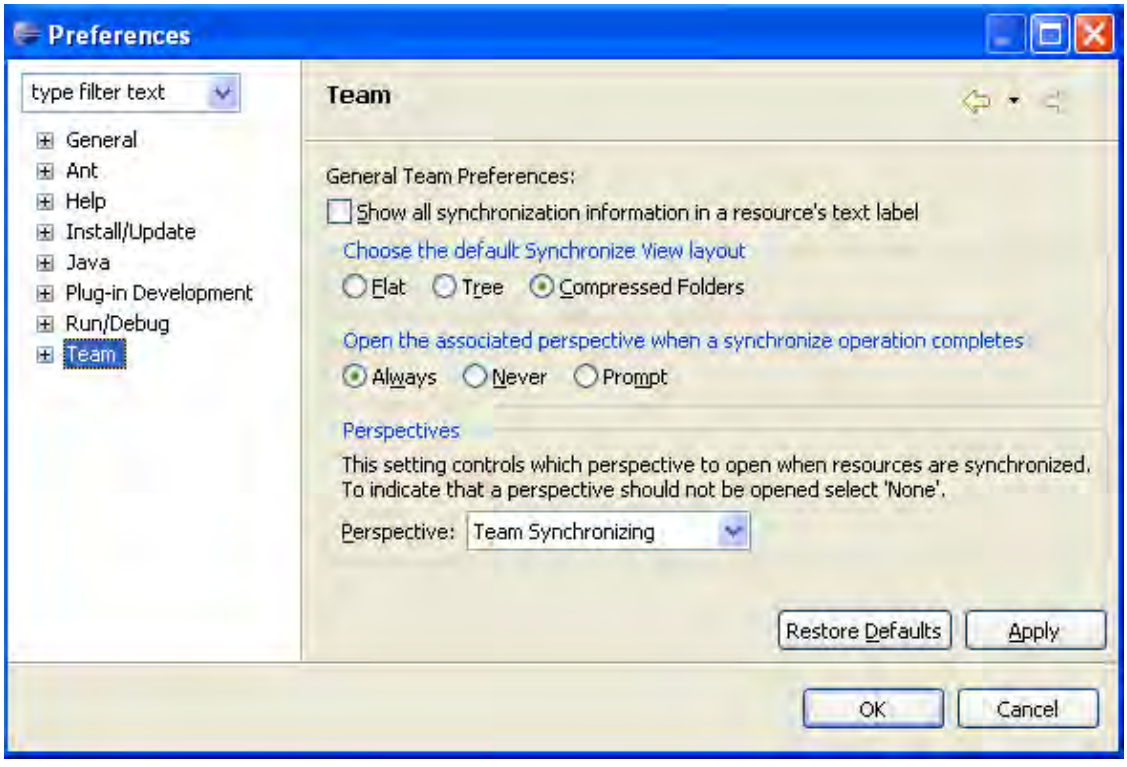
CVS Merge Synchronization

Team

The ***Team*** preferences page contains options which affect the version management Team support.

Option	Description	Default
Show all synchronization information in a resource's text label	Use this option to have the synchronization state of a resource displayed as text in the resource's label. By default, only an icon decorator is used to identify a resource's synchronization state.	Off
Choose the default Synchronize View layout	Use this option to configure the default layout used when a synchronization is first added to the Synchronize view. The layout can be subsequently changed in the view drop down menu.	Compressed Folders
Open the associated perspective when a synchronize operation completes	Use this option to configure what happens when a synchronization operation is run. The options are: • ***Always***: always switch perspectives • ***Never***: never switch perspectives • ***Prompt***: prompt to switch perspectives	Always
Perspectives	Use this option to configure which perspective is to be shown when a synchronize operation is run.	Team Synchronizing perspective

Here is what the Team preference page looks like:



Related reference

CVS Synchronize view

Perspectives

CVS Watch/Edit

The following preferences can be changed on the CVS Watch/Edit preference page.

Option	Description	Default
Configure projects to use Watch/edit on checkout	Use this option to indicate that files checked out out from the repository should be made read−only.	Disabled
When read−only files are modified in an editor	Use this option to configure what occurs when a read−only file is modified in an open editor or by another tool. The options are: • ***Send a cvs edit notification to the server:*** Issues a cvs edit notification to the server before making the file writable. If other editors exist on the file, the user will be prompted and may continue or cancel. • ***Send a cvs edit notification to the server in the background:*** Issues a cvs edit notification to the server in the background after making the file writable. This allows the user to keep typing uninterrupted while the edit notification is sent to the server. • ***Edit the file without informing the server:*** Makes the file read−only without notifying the server.	Send a cvs edit notification to the server in the background
Before a CVS edit notification is sent to the server	Use this option to configure what occurs when a read−only file is modified in an open editor or by another tool and the ***Send a cvs edit notification to the server*** is enabled. The options are: • ***Always Prompt:*** Always prompt the user for confirmation • ***Only prompt if there are other editors:*** Shows the user the list of current editors and allows the user to confirm or cancel the edit. • ***Never Prompt:*** Send the edit notification without prompting	Only prompt if there are editors

Here is what the preference page looks like:

Related concepts

Watch/Edit

Related tasks

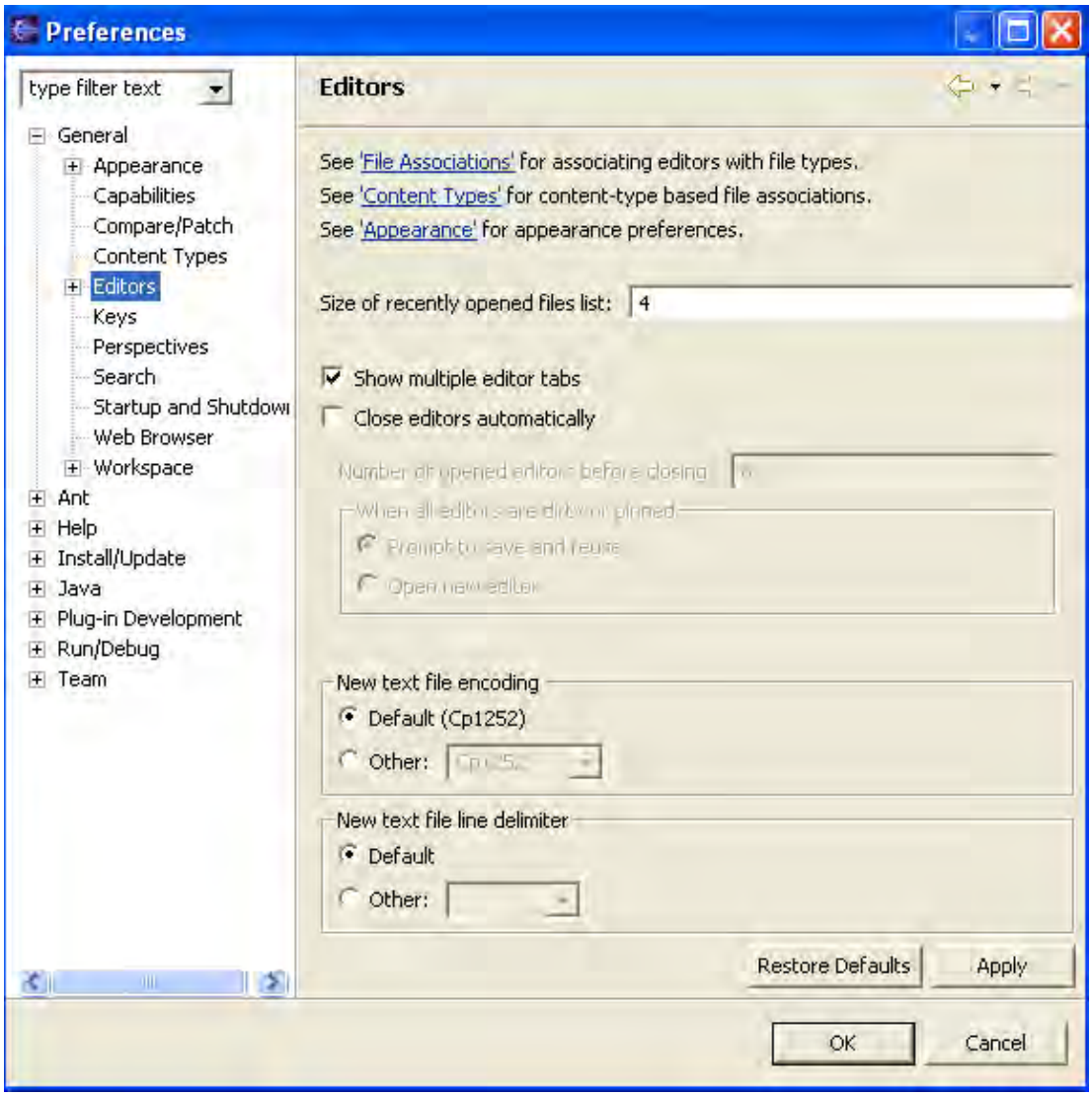
Finding out who's working on what: watch/edit

Editors

The following preferences can be changed on the Editors page.

Option	Description	Default
Size of recently opened files list	For each file that is opened in an editor, it is stored in a list of recently used files. This option controls the number of files which is displayed in this list in the File menu.	4
Show multiple editor tabs	Specifies whether you wish to show multiple editor tabs. If off, then editor workbooks have one large tab and all non−visible editors are accessible only from the chevron.	On
Close editors automatically	This option is used to specify whether or not to re−use editors in the Workbench. If turned on, then you may specify the number of editors to use before they are recycled (Default is 8). You can also specify if a prompt dialog should be opened or if a new editor should be opened when all editors are dirty. Once it is turned on, the Pin Editor action is added to the toolbar and editor tab menu. Pinned editors are not recycled.	Off
Text file encoding	Use this option to specify the encoding to use when saving text files in editors.	Default (CP1252)
Text File line delimiter	Use this option to specify the line delimiter to use for new text files. ***Note:*** This will generally not effect the file line delimiter for existing files.	Default

Here is what the Editors preference page looks like:



Related reference

Editor Area

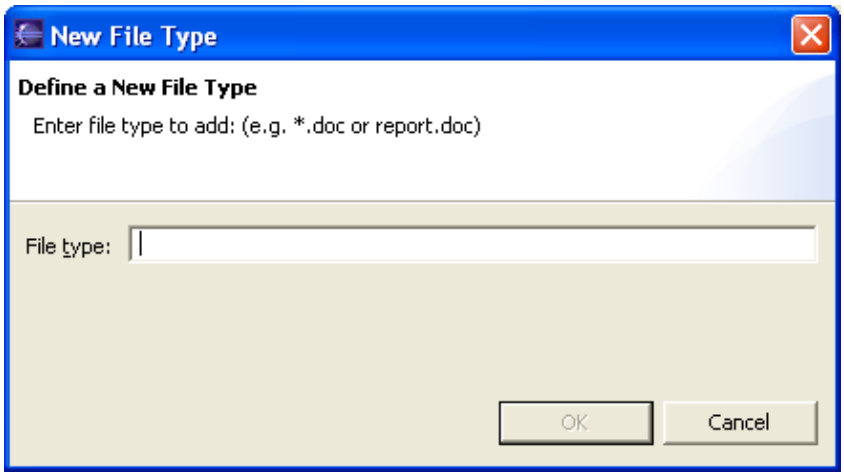
File Associations

On the File Associations preference page, you can add or remove file types recognized by the Workbench. You can also associate editors with file types in the file types list.

File types list

- ***Add...***: Adds a new file or file type (extension) to the predefined list. In the resulting New File Type dialog, type the name of a file or a file extension. If you are adding a file extension, you must type either a dot or a "*." before the file type (e.g., ".xml" or "*.xml" as opposed to simply "xml").
- ***Remove***: Removes the selected file type from the list

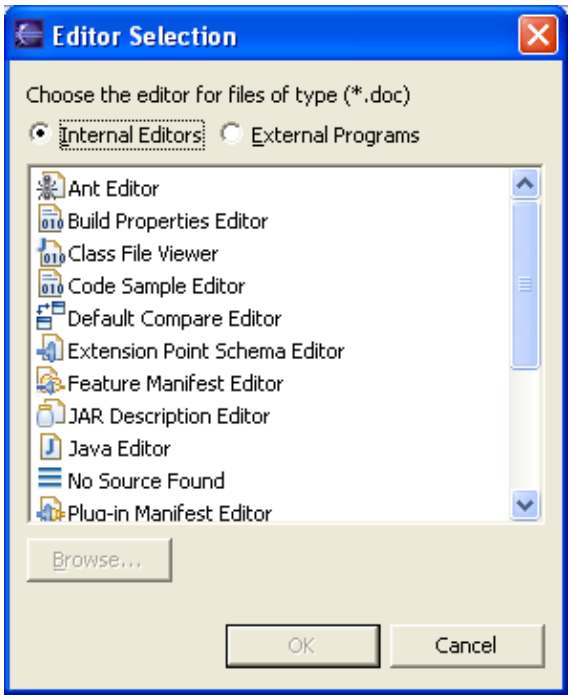
Dialog to create a new file type:



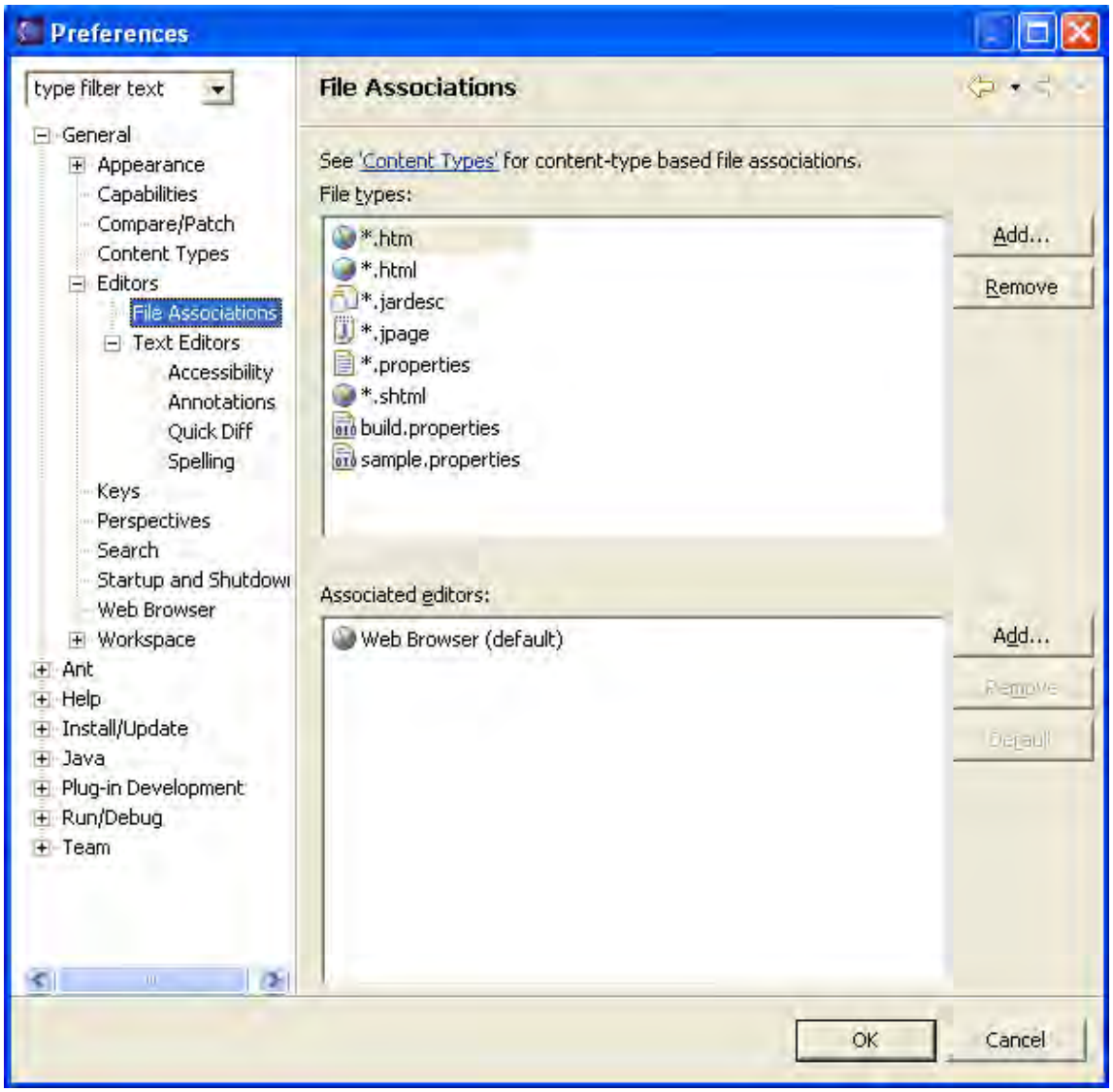
Associated editors list

- ***Add...***: Adds a new editor to the list of editors associated with the file type selected above. In the resulting Editor Selection dialog, you can choose an editor to launch either inside the Workbench (internal) or outside the Workbench (external); click **Browse** to locate an editor yourself if the editor you want is not displayed in the list.
- ***Remove***: Removes the association between an editor and the file type selected above. ***Note:*** Any editor that is bound by content type may not be removed from this list. Currently, there is no mechanism available to remove these editors.
- ***Default***: Sets the selected editor as the default editor for the file type selected above. The editor moves to the top of the Associated Editors list to indicate that it is the default editor for that file type.

Dialog to create a new file association:



Here is what the File Associations preference page looks like:



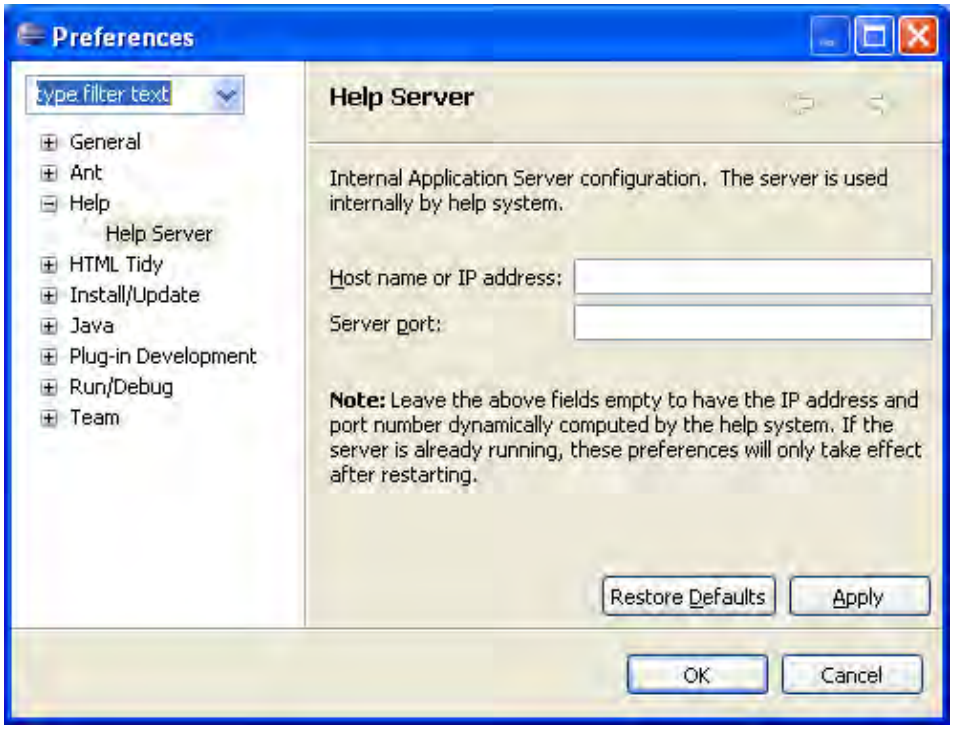
Associated editors list

Help Server

Help system includes an internal server that serves help contents to the browser. Use this preference page to change the interface and port that the servers uses. You should only change these settings if you experience problems and cannot view help with the default preferences.

Option	Description	Default
Host	**Name or address of a local IP interface to be used by the server.**	**blank**
Port	**IP port for server to listen on. If no port is specified for the value, a port will be assigned by the operating system.**	**blank**

Here is what the Help Server preferences page looks like:

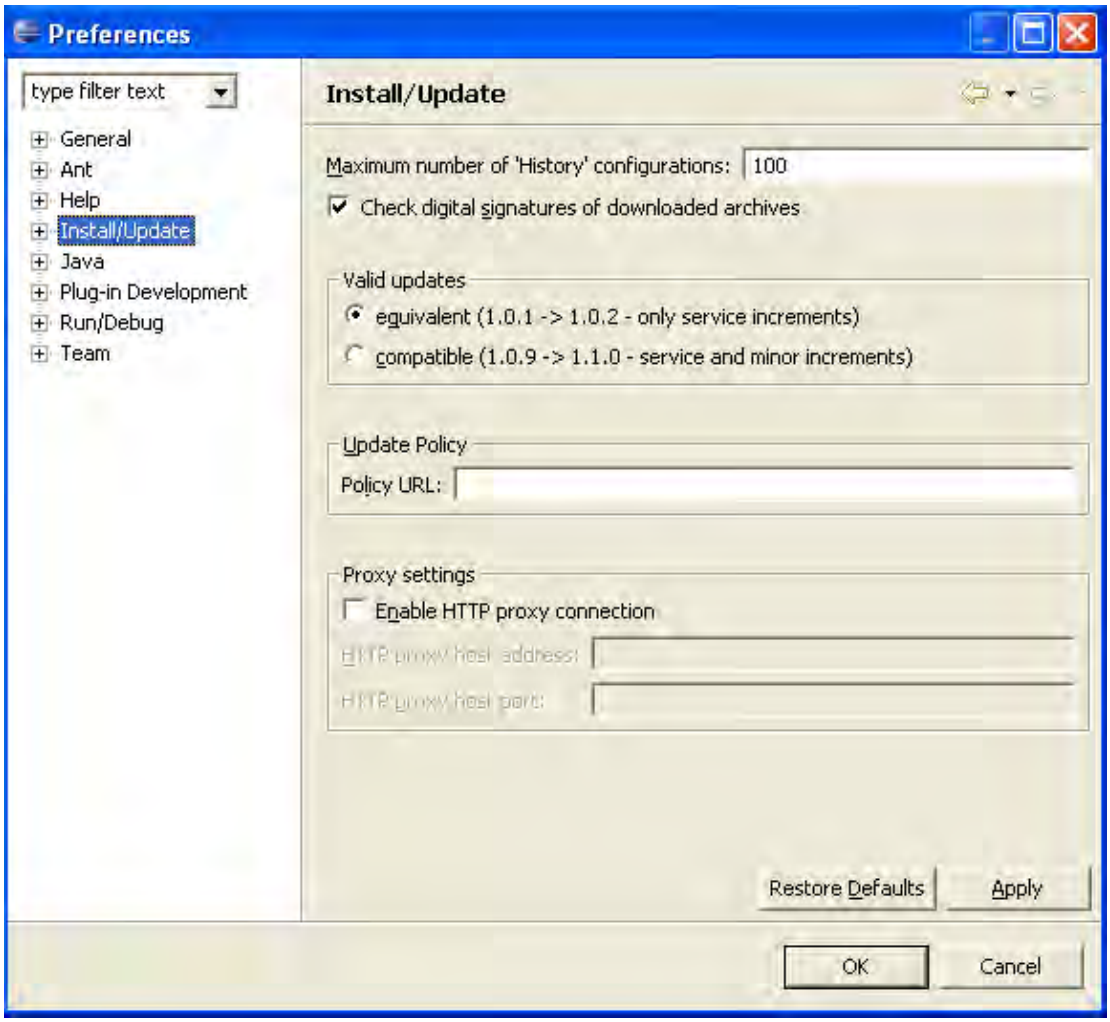


Install/Update

The following preferences can be changed on the Install/Update page:

Option	Description	Default
Maximum number of 'History' configurations	Maximum number of configurations you want maintained in the configuration history. These configurations are maintained to allow you to revert to a previous configuration of installed feature versions.	100
Check digital signatures of downloaded archives	This option will check for digital signatures of downloaded archives.	On
Valid updates	Assuming that feature versions use the form 'major.minor.service', you can select what update level you want to choose from: equivalent Only service updates will be displayed. compatible Service updates and minor updates will be displayed.	equivalent
Update Policy	The update policy URL that controls the redirection of update sites within an organization.	No policy
Proxy settings	Allows connection to the remote server from behind the firewall. When enabled, proxy host address (required) and port number (optional) can be specified.	No proxy

Here is what the Install/Update preference page looks like:



Related tasks

Updating Eclipse with the update manager

Related reference

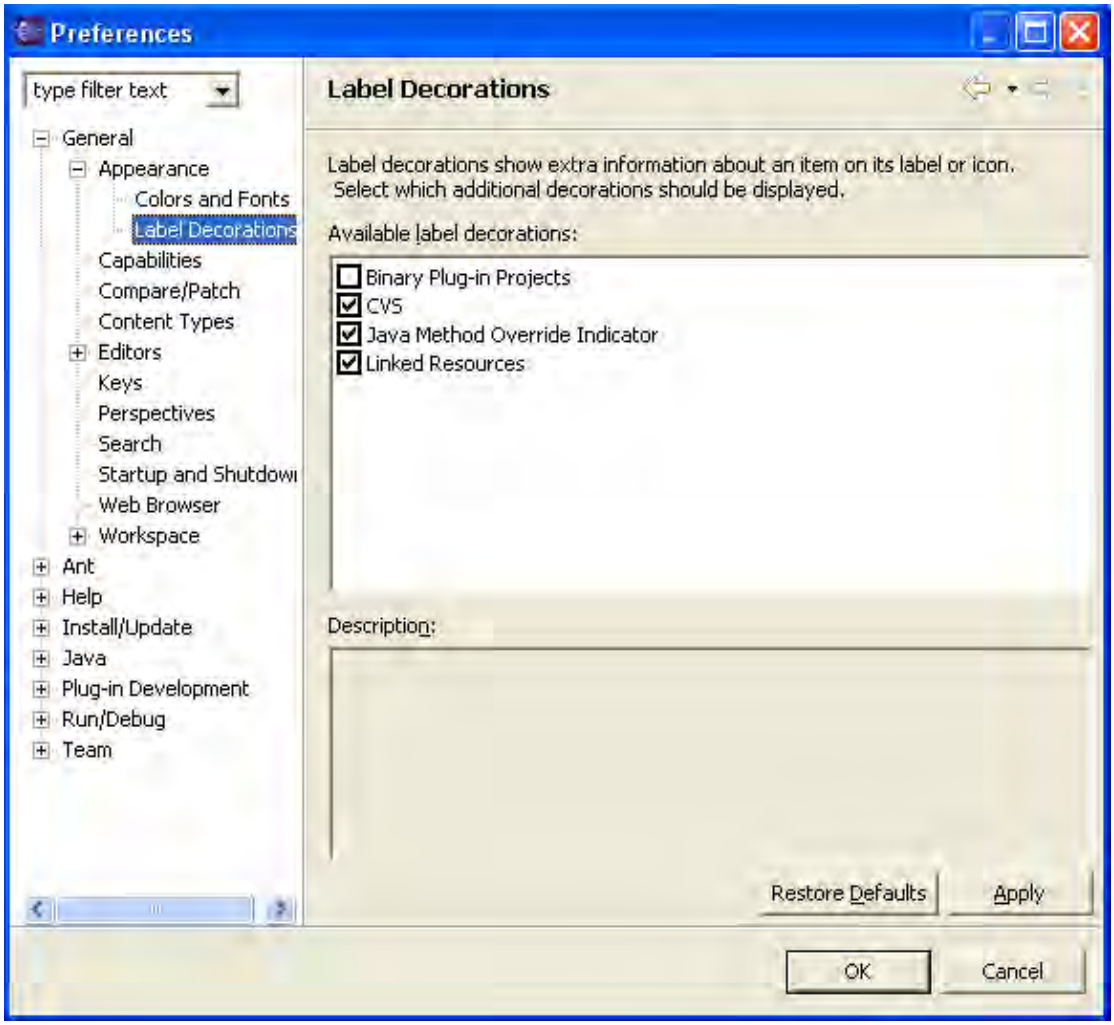
Help Menu

Label Decorations

Label Decorations allow additional information to be displayed in an item's label and icon.

The Label Decorations preference page provides a description of each decoration and allows the selection of which decorations are visible.

Here is what the Label Decorations preference page looks like:



Related concepts

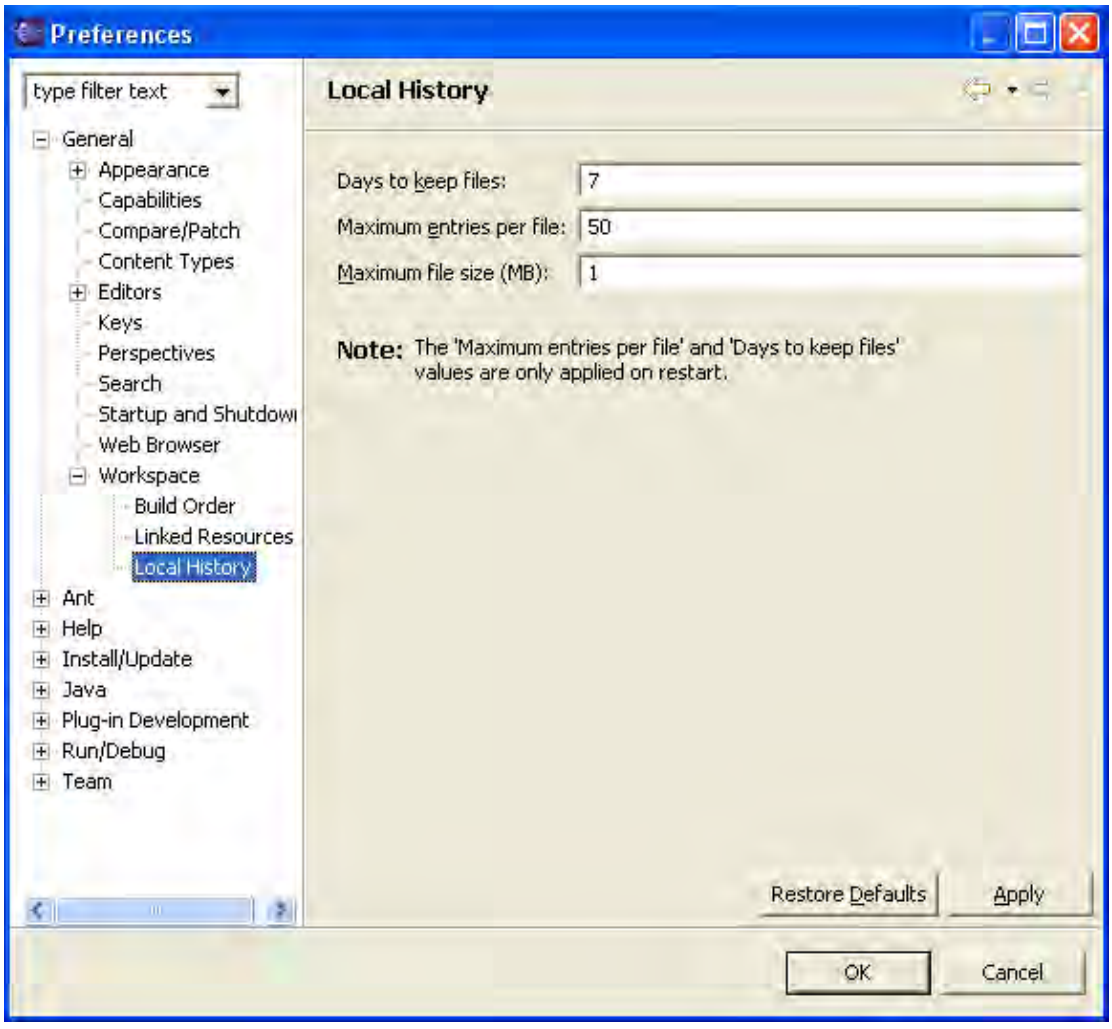
Label Decorations

Local History

The following preferences can be changed on the Local History page.

Option	Description	Default
Days to keep files	Indicates for how many days you want to maintain changes in the local history. History state older than this value will be lost.	7 days
Maximum entries per File	Indicates how many history states per file you want to maintain in the local history. If you exceed this value, you will lose older history to make room for new history.	50 entries
Maximum file size (MB)	Indicates the maximum size of individual states in the history store. If a file is over this size, it will not be stored.	1 MB

Here is what the Local History preference page looks like:



Related reference

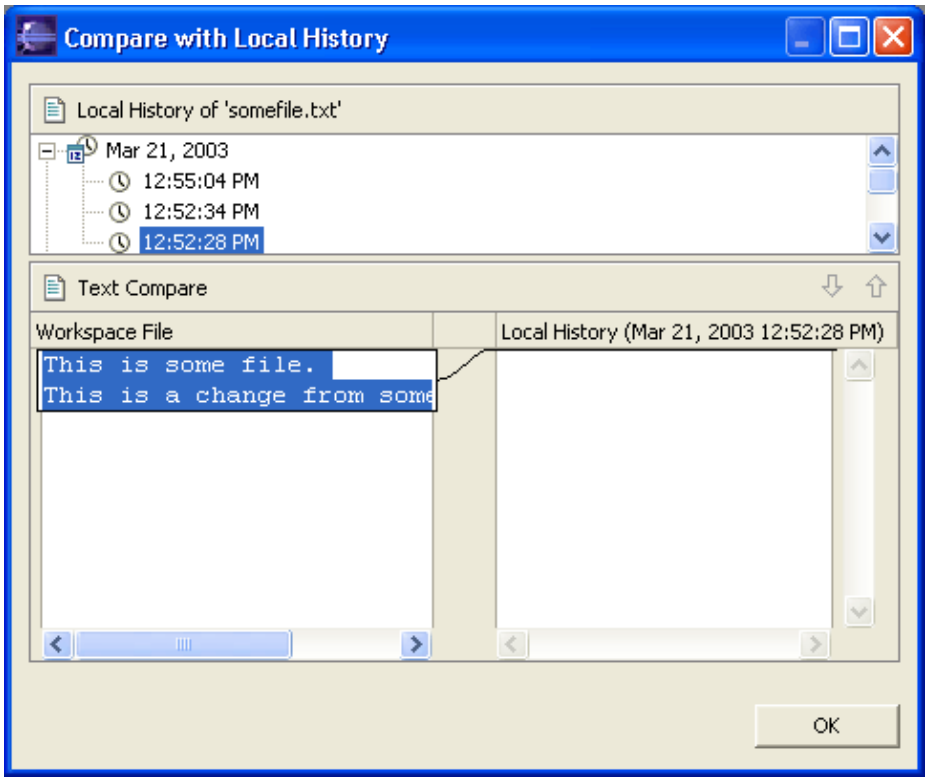
Local History

Local history

Local history of a file is maintained when you create or modify a file. Each time you edit and save a file, a copy of it is saved. This allows you to compare your current file state to a previous state, or replace the file with a previous state. Each state in the local history is identified by the date and time the file was saved.

Neither projects nor folders have local history.

Here is a look at what the local history of a Workbench file might look like:



To view the local history of a file, choose Team > Compare with > Local History... from the pop−up menu. You can select different local states in the list, which are compared against the current file. You can also revert to the local history when you select a file and select the Team > Replace With > Local History... menu item.

Perspectives

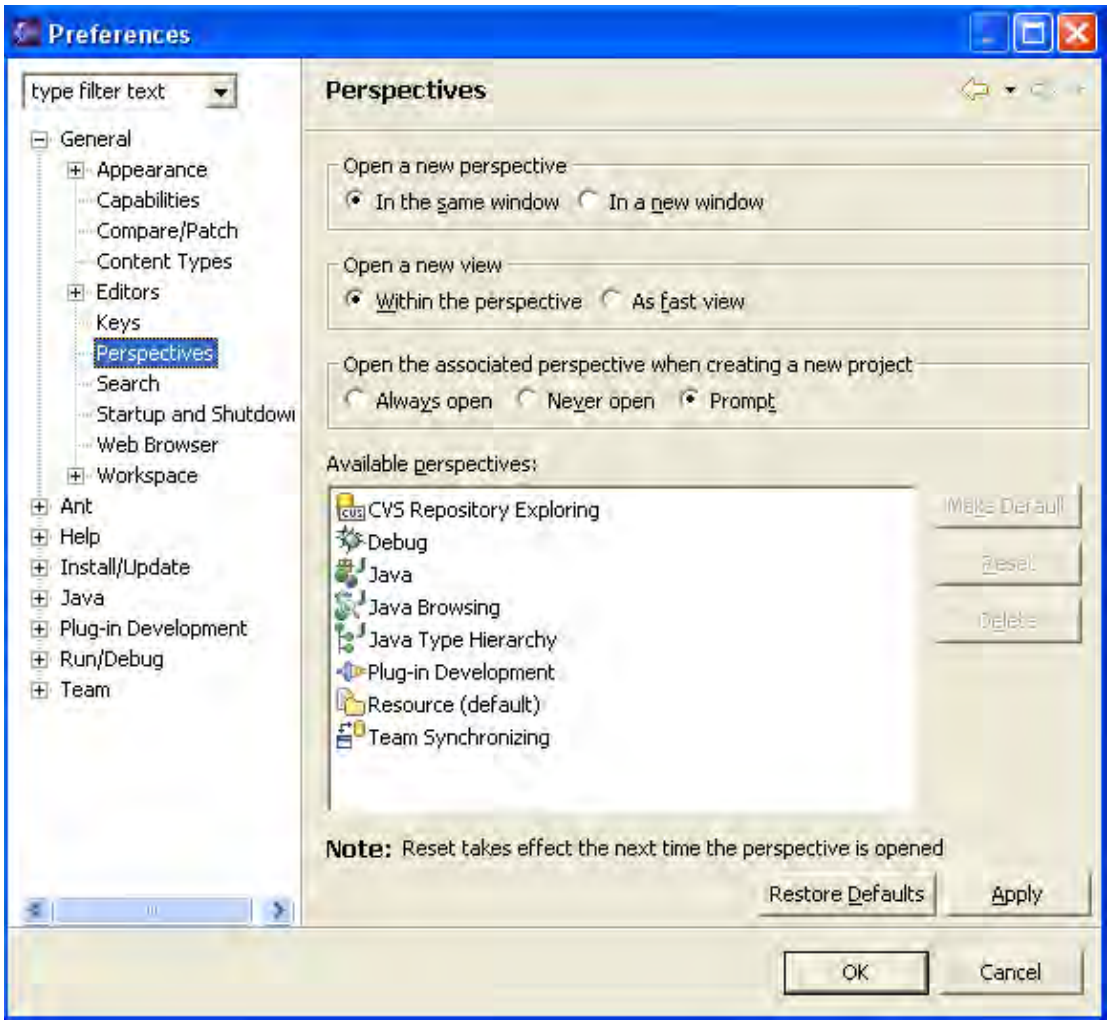
On the Perspectives preference page, you can manage the various perspectives defined in the Workbench.

Option	Description	Default
Open a new perspective	Use this option to set what happens when you open a new perspective. Do you want the perspective opened within the current Workbench window or opened in a new window?	In the same window
Open a new view	Use this option to specify what happens when a new view is opened. It is either opened to its default position within the current perspective or it is opened as a fast view and docked to the side of the current perspective.	Within the perspective
New project options	Use this option to specify the perspective behavior when a new project is created. You can set it to switch the current perspective to be the one associated with the project type and open the perspective in the same Workbench window as the current one, switch the perspective and open it in a new Workbench window, or not to switch perspectives at all.	Open perspective in the same window

Available Perspectives Options:

Option	Description	Default
Make Default	Sets the selected perspective as the default perspective.	Resource
Reset	Resets the definition of the selected perspective to the default configuration. This option is only applicable to built−in perspectives that have been overwritten using Window > Save Perspective As...	n/a
Delete	Deletes the selected perspective. This option is only applicable to user−defined perspectives (built−in perspectives can not be deleted).	n/a

Here is what the Perspectives preferences page looks like:



Related reference

Perspectives

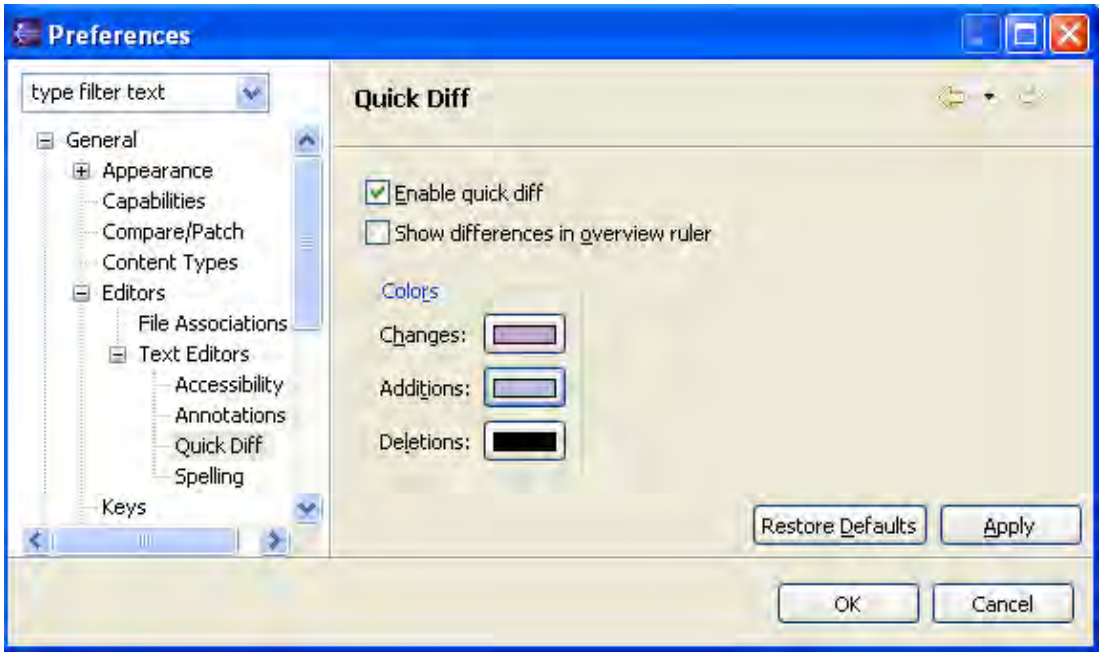
Available Perspectives Options:

Quick Diff Preference Page

The following preferences can be changed on the Quick Diff preference page.

Option	Description	Default
Enable quick diff	This option will enable or disable the quick diff option.	On
Show differences in overview ruler	This option will show differences in the overview ruler.	Off
Colors − Changes	This option controls the color of changes.	
Colors − Additions	This option controls the color of additions.	
Colors − Deletions	This option controls the color of deletions.	

Here is what the Quick Diff preference page looks like:

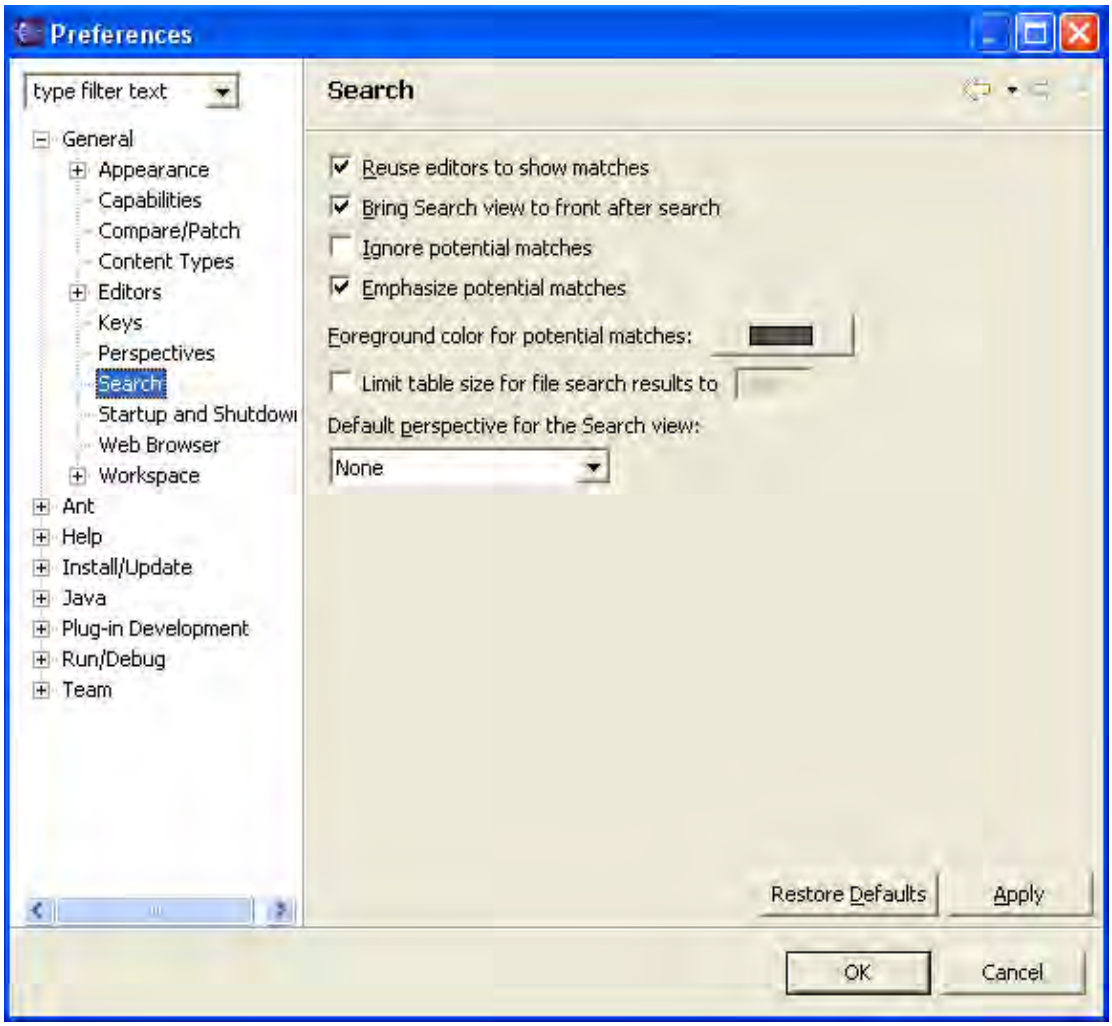


Search

The Search preference page allows the user to set preferences for searches.

Option	Description	Default
Reuse editors to show matches	This option allows the user to keep using the same editor for search results to reduce the number of open editors.	On
Bring Search view to front after search	This option will display the search view at the front after performing a search.	On
Ignore potential matches	Select this option if you only want to see exact matches.	Off
Emphasize potential matches	This option allows you to highlight potential matches in the Search view. If the Search engine isn't 100% sure about the match then it is considered a potential match.	On
Foreground color for potential matches	This option allows you to select the foreground color for potential matches.	
Limit table size for file search results to	This option allows you to limit the table size for file search results to a preset number.	Off
Default perspective for the Search view	This option allows you to define which perspective should be brought to the front when there are new search results.	None

Here is what the Search preference page looks like:

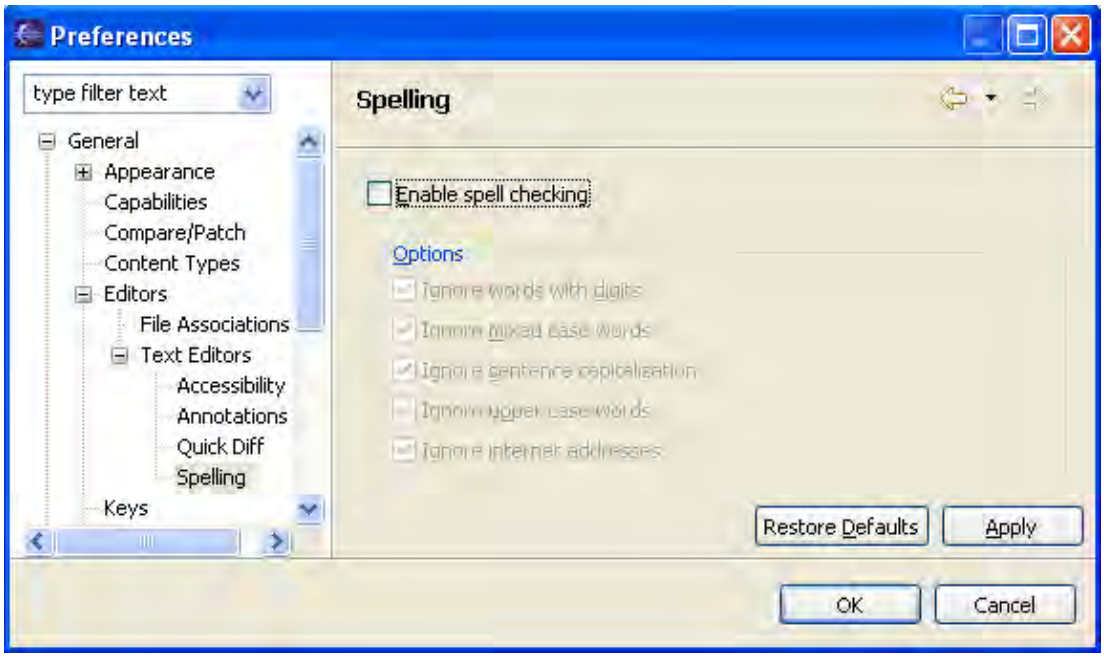

Preferences
type filter text
General
Appearance
Capabilities
Compare/Patch
Content Types
Editors
Keys
Perspectives
Search
Startup and Shutdowi
Web Browser
Workspace
Ant
Help
Install/Update
Java
Plug-in Development
Run/Debug
Team
Search
Reuse editors to show matches
Bring Search view to front after search
Ignore potential matches
Emphasize potential matches
Foreground color for potential matches:
Limit table size for file search results to
Default perspective for the Search view:
None
Restore Defaults
Apply
OK
Cancel

Spelling Preference Page

The following preferences can be changed on the Spelling preference page.

Option	Description	Default
Enable spell checking	This option enables spell checking.	Off
Ignore words with digits	This option ignores words with digits when performing spell checking.	Off
Ignore mixed case words	This option ignores mixed case words when performing spell checking.	Off
Ignore sentence capitalization	This option ignores sentence capitalization when performing spell checking.	Off
Ignore upper case words	This option ignores upper case words when performing spell checking.	Off
Ignore internet addresses	This option ignores internet addresses when performing spell checking.	Off

Here is what the Spelling preference page looks like:



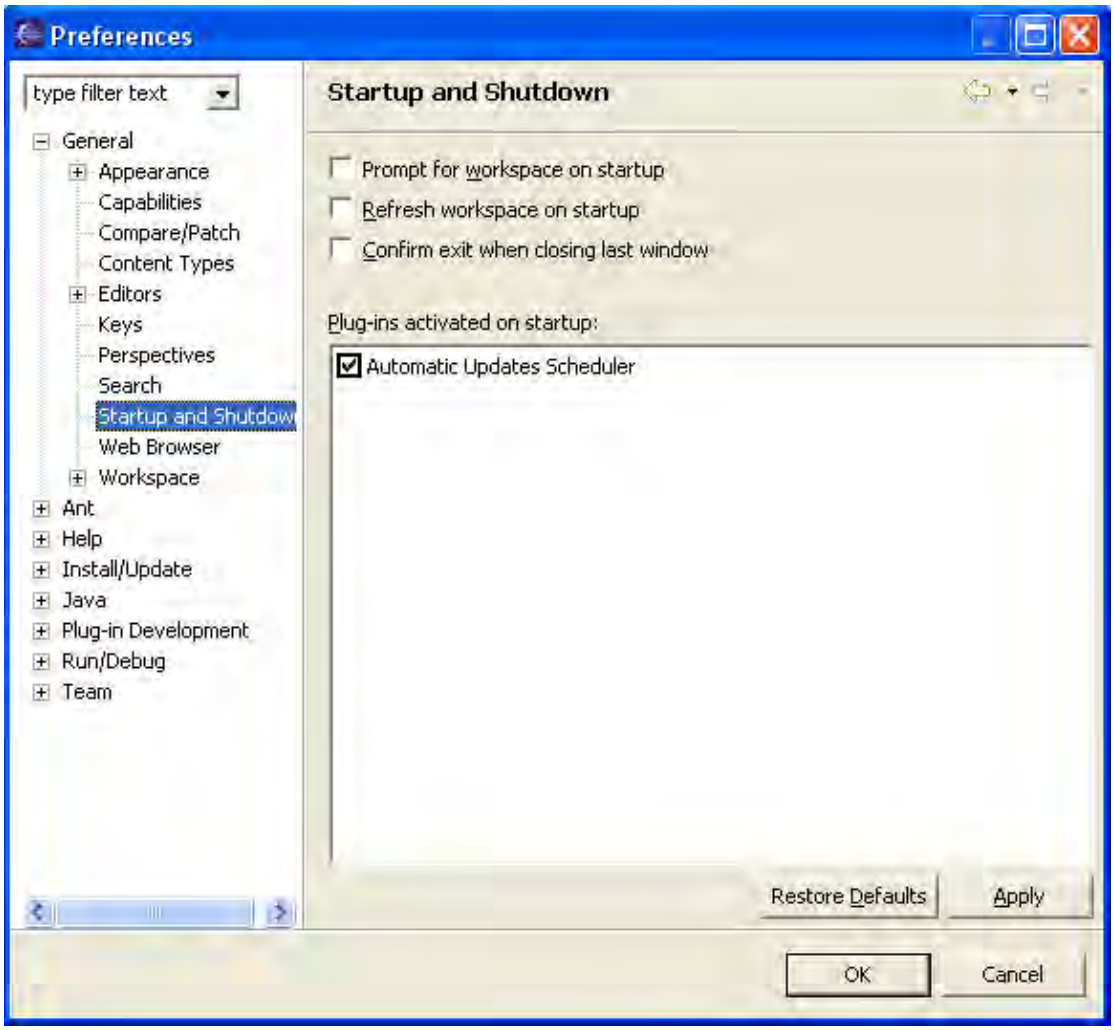
Startup and Shutdown

The Startup and Shutdown preference page allows the selection of plug−ins to be automatically activated during workbench startup.

Normally plug−ins are not activated until they are needed. However some plug−ins may specify that they wish to be activated during startup. This preference page allows the selection of which of these plug−ins will actually be activated during startup.

Option	Description	Default
Prompt for workspace on startup	If this option is turned on then the workbench will prompt the user each time it is started for what workspace to use.	On
Refresh workspace on startup	If this option is turned on then the workbench will synchronize its contents with the file system on startup.	Off
Confirm exit when closing last window	If this option is turned on then the workbench will ask the user if they wish to exit when closing the last window if.	On
Plug−ins activated on startup	This option allows you to select which available plug−ins should be activated on startup.	

Here is what the Startup and Shutdown preference page looks like:


Preferences
type filter text
General
Appearance
Capabilities
Compare/Patch
Content Types
Editors
Keys
Perspectives
Search
Startup and Shutdow
Web Browser
Workspace
Ant
Help
Install/Update
Java
Plug-in Development
Run/Debug
Team
Startup and Shutdown
Prompt for workspace on startup
Refresh workspace on startup
Confirm exit when closing last window
Plug-ins activated on startup:
Automatic Updates Scheduler
Restore Defaults
Apply
OK
Cancel

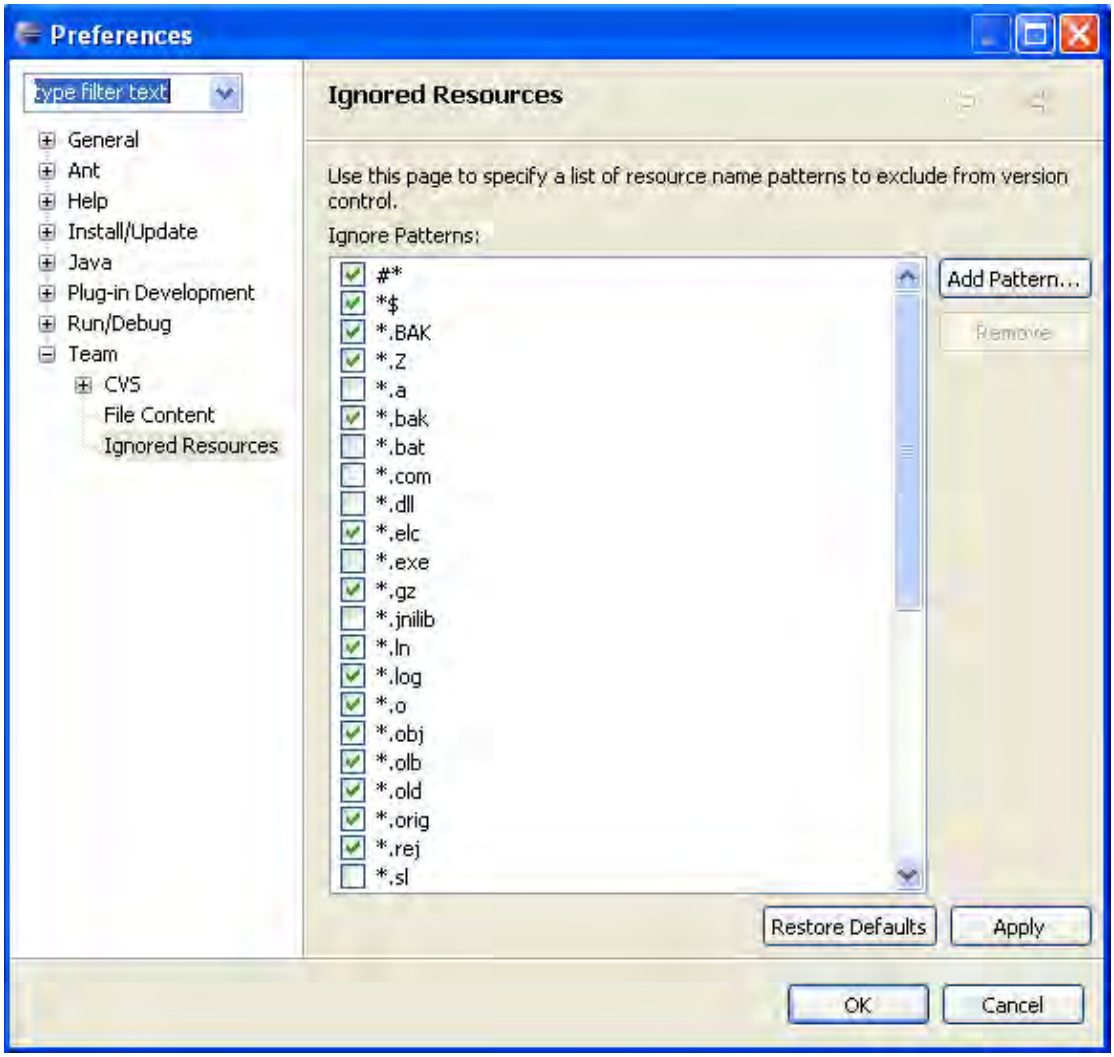
Team Ignored Resources

On the ***Team>Ignored Resources*** preference page, you can specify file name patterns to exclude from the version control management system. There is a list of file patterns against which resources will be matched before they are considered as version control candidates. These patterns may contain the wildcard characters "*" and "?". This pattern "*" represents any sequence of zero or more characters. This pattern "?" represents any one character. For example, you can specify a pattern of "*~", which would match any temporary files that end with "~". Any file or directory that matches any one of the patterns will be ignored during update or commit operations.

To add a file type to the ignore list, simply click ***Add Pattern***. In the dialog, enter a file type (e.g. *.class). To remove a file type from the ignore list, simply select the file type in the ignore list and click ***Remove***.

You can temporarily disable ignoring the file pattern by de−selecting it from the list. You do not have to remove the specified file pattern from the list to temporarily disable it.

Here is what the Ignored Resources preference page looks like:



Related reference

Ignoring Resources from Version Control

Ignoring resources from version control

When synchronizing resources, it is possible that there are some resources that you do not want to commit to the repository. There are two ignore facilities provided, allowing the user to specify which resources should be excluded from update and commit operations.

The first is a global ignore facility, provided by the Workbench. The second is the CVS ignore facility, which reads the contents of a special file *.cvsignore* to determine what to ignore.

Why ignore files when synchronizing?

There are many files that a user may not want to commit to the repository. For example, external editors may create temporary files in your project. Compilation of .java files creates .class files, and likewise many build operations result in binary files. These files, when taken together, may be quite large. In addition, they may be re−generated whenever a build is performed, resulting in many outgoing changes. Typically these are not files that one wants to share with other members of a team or persist in the repository.

Global ignore facility

A global ignore facility is provided by the Workbench via the Team preference page. There is a list of file patterns against which resources will be matched before they are considered as version control candidates. These patterns may contain the wildcard characters "*" and "?". "*" represents any sequence of zero or more characters. "?" represents any one character. For example, you can specify a pattern of "*~", which would match any temporary files that end with "~". Any file or directory that matches any one of the patterns will be ignored during update or commit operations. When you specify a file pattern to ignore, you can temporarily disable ignoring the file pattern by de−selecting it from the list; you do not have to remove the specified file pattern from the list.

The patterns in the global ignore facility are matched against resource names during a synchronize operation. It is important to note that the path leading up to the resource name is not included in the matching. For example, for the file "/path/to/file.txt", only the string "file.txt" is matched against the patterns. This facility is not intended for specifying fully−qualified path names but for specifying globally−applicable patterns.

CVS ignore facility

The Eclipse CVS client recognizes a file named ".cvsignore" in each directory of a project. This is a standard CVS facility and many existing CVS projects may contain this file.

This text file consists of a list of files, directories, or patterns. In a similar way to the global ignore facility, the wildcards "*" and "?" may be present in any entry in the .cvsignore file. Any file or sub−directory **in the current directory** that matches any one of the patterns will be ignored. It is important to note that the semantics of this file differs from that of the global ignore facility in that it applies only to files and directories in the same directory as the .cvsignore file itself. A project may contain one .cvsignore file in each directory. For more information, please visit https://www.cvshome.org.

Resources that have not been added to CVS control can be ignored by selecting **Team > Add to .cvsignore** from the pop−up menu of the resource in one of the navigation views. This menu option is also available in the Synchronize view.

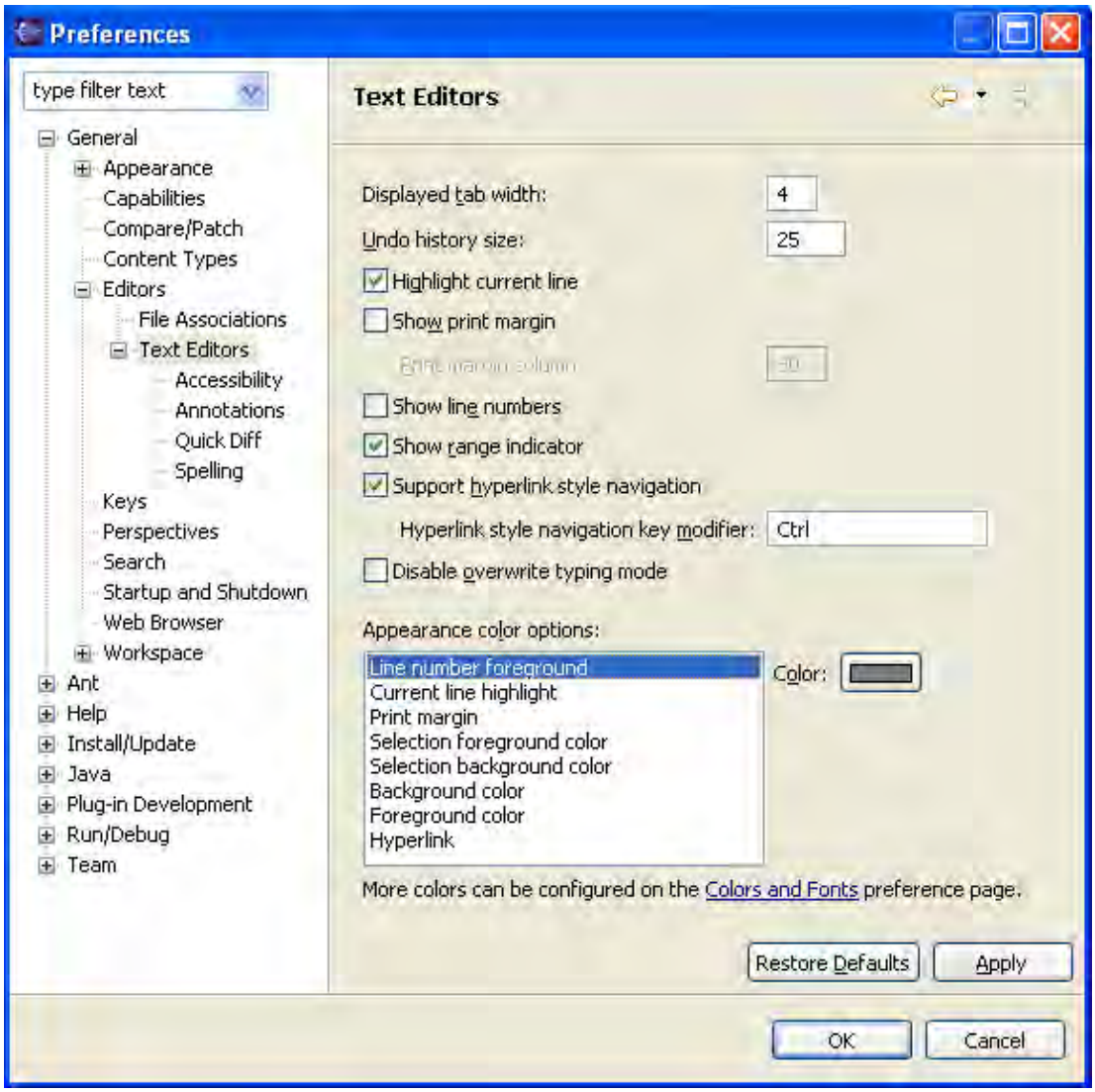
Text Editor Preference Page

The following preferences can be changed on the Text Editor page.

Appearance options

Option	Description	Default
Displayed tab width	This option allows you to set the displayed tab width for the text editor.	4
Undo history size	This option allows you to undo the history size for the text editor.	25
Highlight current line	This option controls whether or the current line is highlighted or not.	On
Show print margin	This option controls whether the print margin is visible or not.	Off
Show line numbers	This option controls whether or not line numbers are shown on the left side of the text editor.	Off
Show range indicators	This option controls whether or not range indicators are shown in the text editor.	On
Support hyperlink style navigation	This option controls whether or not hyperlink style navigation is supported.	On
Hyperlink style navigation key modifier	This option sets the hyperlink style navigation key modifier.	Ctrl
Disable overwrite typing mode	This option controls whether the overwrite typing mode is enabled or disabled.	Off
Appearance color options	This option controls various appearance colors.	

Here is what the Text editor preference page looks like:



Related reference

Editor

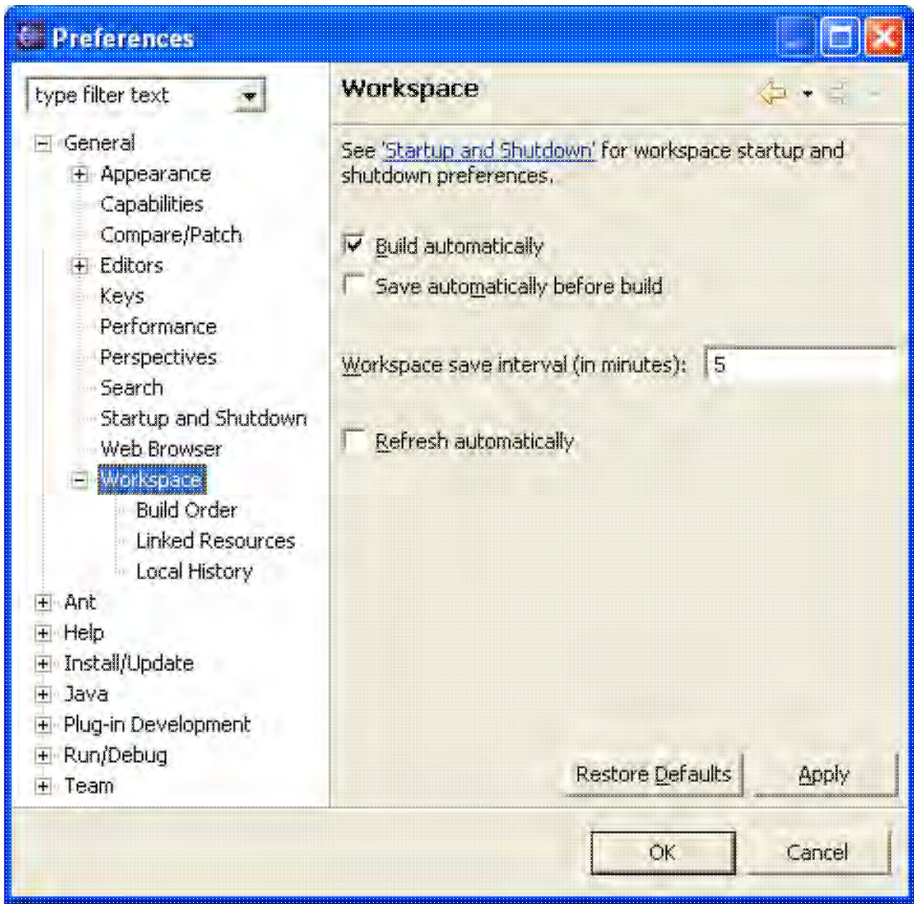
Workspace

On the ***Workspace*** preference page, you can manage various IDE−specific workspace preferences settings in the Workbench.

The following preferences can be accessed on the Workspace page by selecting ***Window*** > ***Preferences*** > ***General*** > ***Workspace***.

Option	***Description***	***Default***
Build automatically	If this option is turned on, then the Workbench will perform an automatic build whenever a modified resource is saved.	On
Save automatically before build	If this option is selected, when a manual build is performed the Workbench will automatically save all resources that have been modified since the last build was performed.	Off
Workspace save interval (in minutes)	This number indicates how often the state of the workspace is automatically saved to disk.	5
Refresh automatically	If this option is turned on then the workspace resources will be synchronized with their corresponding resources in the file system automatically. ***Note***: This can potentially be a lengthy operation depending on the number of resources you have in your workspace.	Off

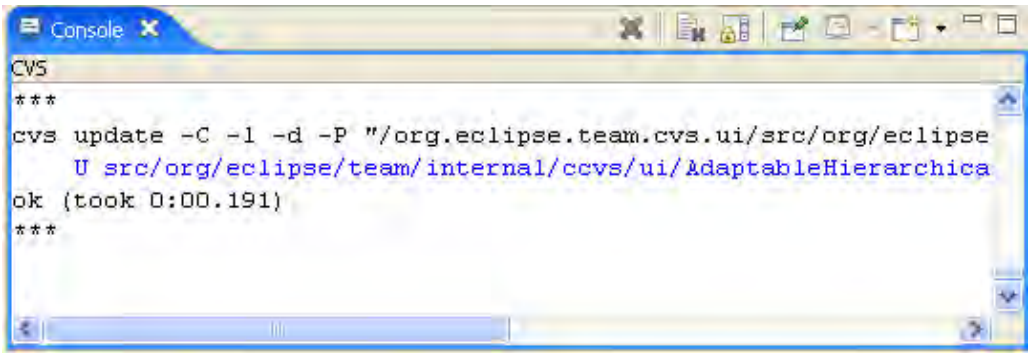
Here is what the ***Workspace*** preferences page looks like:


Preferences
type filter text
General
Appearance
Capabilities
Compare/Patch
Editors
Keys
Performance
Perspectives
Search
Startup and Shutdown
Web Browser
Workspace
Build Order
Linked Resources
Local History
Ant
Help
Install/Update
Java
Plug-in Development
Run/Debug
Team
Workspace
See 'Startup and Shutdown' for workspace startup and shutdown preferences.
Build automatically
Save automatically before build
Workspace save interval (in minutes):
5
Refresh automatically
Restore Defaults
Apply
OK
Cancel

CVS Console

The CVS Console is shown in the Workbench Console view. The CVS entry in this view shows the output from CVS commands. This output is similar to that of the CVS command line client. If an error occurs during a CVS operation opening the console can help troubleshoot the cause of the error.

Here is what the CVS Console in the Console view looks like:



Related reference

CVS Repositories view

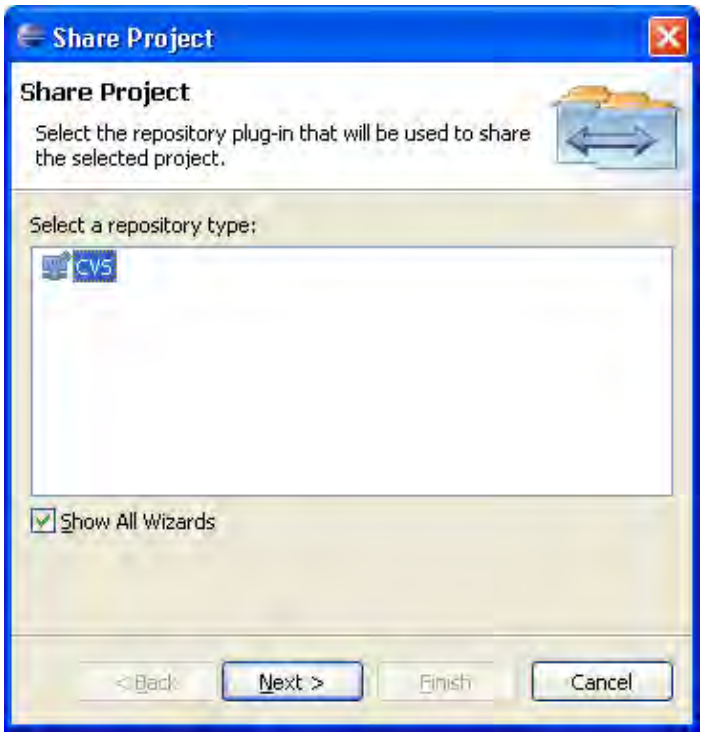
Related concepts

Resources
CVS Repositories
Local history

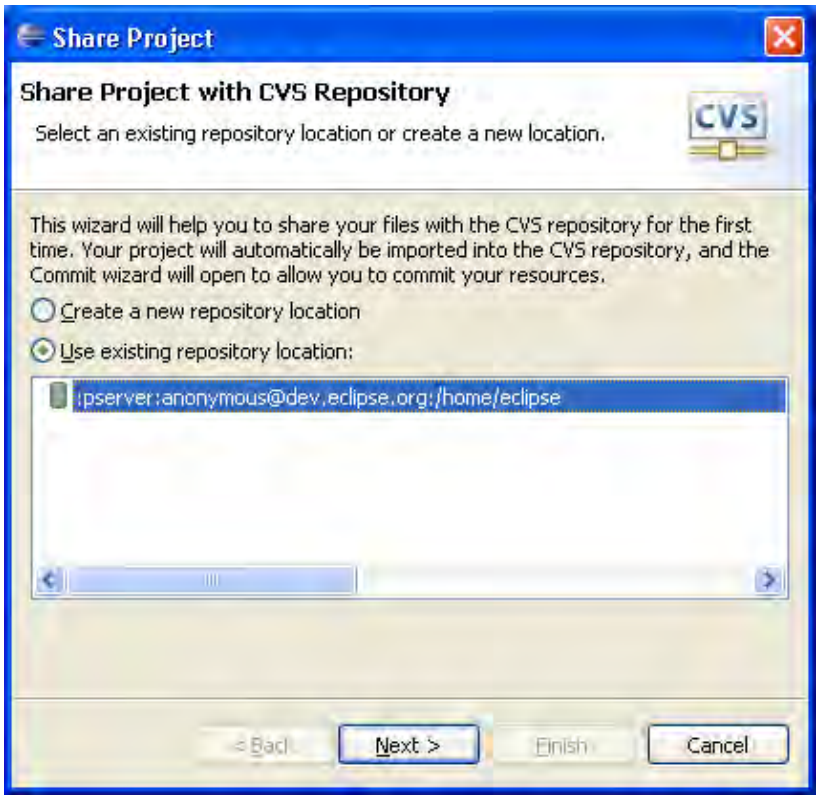
CVS Sharing wizard

This wizard helps you share a project with others using a CVS repository. It is available from the ***Team*** > ***Share Project*** menu command of views that display resources such as one of the navigation views and the Java Packages Explorer.

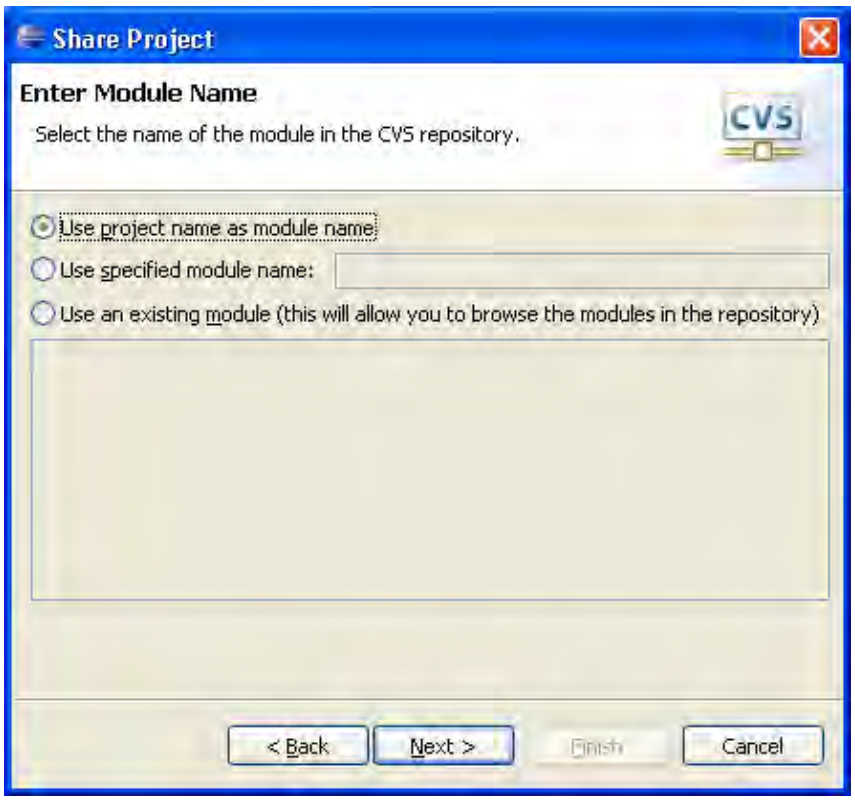
The first page of the Sharing wizard allows you to choose the type of repository you would like to share you project with. If the repository type you wish to share with is not listed, you can click the Show All Wizards checkbox to see repository types that are installed but not enabled.



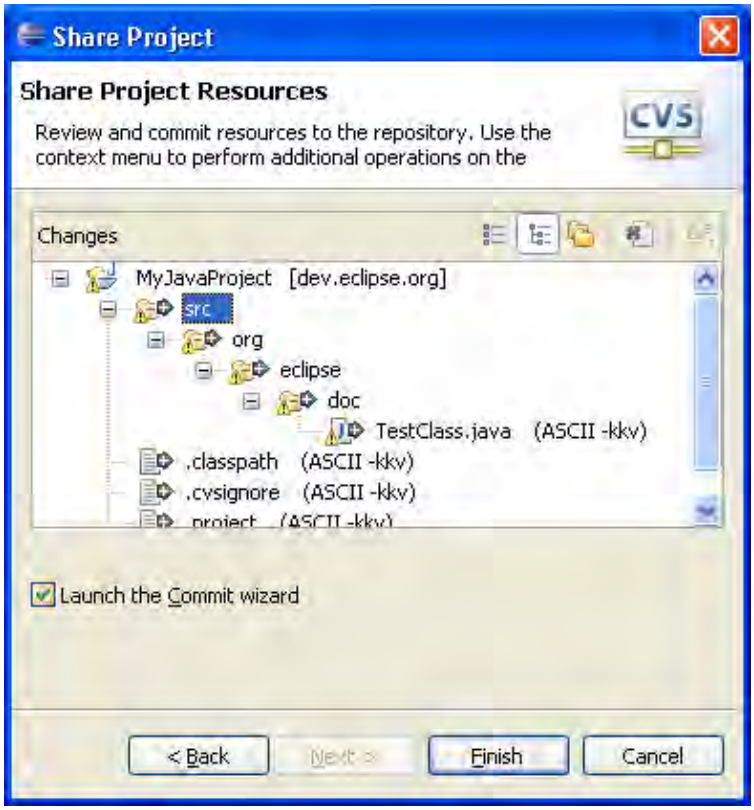
Selecting CVS will bring you to the CVS Sharing wizard. On the first page of the CVS wizard, you can select the repository you wish to share with. If the repository is not in the list, you can select to create a new repository. If you choose to create a new location, the page from the New Repository Location wizard is shown.



After you have selected the repository, you must indicate the name of the module that will be used to share the project. You can choose to use the same name as the local project or you can type in a path containing one or more folder segments indicating the remote module. There is also an option to reconnect the local project to an existing module.



The final page of the CVS Sharing wizard allows you to decide which resources to commit and which to ignore. There are toolbar items for changing the layout of the view and for committing and ignoring resources. If there are resources left in the view when Finish is pressed, you will be prompted to commit the remaining resources in the background (if the *Launch the Commit Wizard* option is enabled) which will allow you to return to work while the commit takes place.



Related reference

Add CVS Repository wizard

Workbench toolbar

The Workbench toolbar is displayed at the top of the Workbench window, directly underneath the menu bar. The contents of the toolbar change based on the active editor. Actions in the toolbar may apply to particular views, so these actions may be enabled or disabled based on the state of the currently active view or editor.

Here is an example of the toolbar in the Resource perspective:

New Wizard

This command brings up a dialog where you can choose the type of resource to create.

Save The Open Editor Contents

This command saves the file currently displayed in the editor area.

Print

This command opens a dialog which allows you to specify where you would like to print the contents of the file currently being displayed in the editor.

Search

This command opens the search dialog, which allows you to search the workspace for specified text.

External Tools

This command presents a drop−down menu which allows you to run or configure external tools.

Related reference

Shortcut Bar
View Toolbars

Perspective Bar

The perspective bar allows quick access to perspectives that are currently open, as well as providing an easy way to open a new perspective. The perspective bar may be docked in three different positions. It may be docked in the upper right corner (the default position), the upper left corner (under the main toolbar) and to the far left.

Here is an example of what the perspective bar looks like:

Open Perspective

This command opens a new perspective which is selected by the user from a drop−down menu. All of the perspectives that are open within a single Workbench window are shown on the shortcut bar.

Perspective Buttons

These buttons provide a quick way to switch to one of the open perspectives in the current Workbench window.

Available Perspectives

There are several available perspectives, while one is set as a default, others can be manually added to the perspective bar. To add new perspectives to your workspace, click ***Open Perspective***  , select ***Other*** and choose from the following available perspectives:

- CVS Repository Exploring;
- Debug;
- Java;
- Java Browsing;
- Java Type Hierarchy;
- Plug−in Development;
- Resource;
- Team Synchronizing.

Related reference

Perspectives
View Toolbars
Workbench Toolbar
Workbench Window Layout

View toolbars

View toolbars contain actions that apply only to the particular view in which they appear. The view toolbar also contains a context menu that contains other actions for that view. This menu is opened by clicking on the downwards pointing triangle. If there is enough space, view toolbars are in the view tab area. Otherwise they appear in the view.

Here is an example of the view toolbar for the Properties view:

Title Bar

View title bars contain the view name, its icon, and the view toolbar.

Related reference

Shortcut Bar
Workbench Toolbar
Workbench Window Layout

Fast View Bar

The fast view bar is the place where fast views are docked. It may be docked in three locations. It may be docked on the left, bottom (default) and right.

Here is an example of what some fast view bars look like:

View Buttons

These buttons provide a quick way to display the fast views in the current perspective. Fast views are essentially minimized views that have been dragged onto the shortcut bar. Fast views pop up when selected, and revert back to their minimized state when the user clicks outside of the view and may be oriented either horizontally or vertically according to their configuration. To convert a fast view back into a normal view, click on the *Fast View* action in the view's menu or drag it back to the workbench.

Related reference

View Toolbars
Workbench Window Layout

List of key bindings

The list of available key bindings in Eclipse depends on many factors, including what view or editor is selected, whether a dialog is open, what plug−ins are installed, and what operating and windowing system is being used. At any time, you can obtain a list of available key bindings using Key Assist (**Help > Key Assist** or Ctrl+Shift+L). The following tables list some popular key bindings available in the Eclipse SDK.

File actions

New	Create a Java element or a new resource. Configure which elements are shown in the submenu in Window > Customize Perspective. In a Java perspective, by default action for creating a project, package, class, interface , source folder, scrapbook, file and folder are available.	Ctrl + N
Close	Close the current editor. If the editor contains unsaved data, a save request dialog will be shown.	Ctrl + F4
Close All	Close all editors. If editor contains unsaved data, a save request dialog will be shown.	Ctrl + Shift + F4
Save	Save the content of the current editor. Disabled if the editor does not contain unsaved changes.	Ctrl + S
Save As	Save the content of the current editor under a new name.	
Save All	Save the content of all editors with unsaved changes. Disabled if no editor contains unsaved changes.	Ctrl + Shift +

		S
Print	Prints the content of the current editor. Enabled when an editor has the focus.	Ctrl + P
Properties	Opens the property pages of the select elements. Opened on Java projects the Java Build Path page and the Javadoc Location page are available. For JAR archives, configure the JAR's Source Attachment and Javadoc Location here.	Alt + Enter

Edit actions

Undo	Revert the last change in the editor	Ctrl + Z
Redo	Revert an undone change	Ctrl + Y
Cut	Copies the currently selected text or element to the clipboard and removes the element. On elements, the remove is not performed before the clipboard is pasted.	Ctrl + X
Copy	Copies the currently selected text or elements to the clipboard	Ctrl + C
Paste	Paste the current content as text to the editor, or as a sibling or child element to the a currently selected element.	Ctrl + V
Delete	Delete the current text or element selection.	Delete
Select All	Select all the editor content..	Ctrl + A
Find / Replace	Open the Find / Replace dialog. Editor only.	Ctrl + F
Find Next	Finds the next occurrence of the currently selected text. Editor only.	Ctrl + K
Find Previous	Finds the previous occurrence of the currently selected text. Editor only.	Ctrl + Shift + K
Incremental Find Next	Starts the incremental find mode. After invocation, enter the search text as instructed in the status bar. Editor only.	Ctrl + J
Incremental Find Previous	Starts the incremental find mode. After invocation, enter the search text as instructed in the status bar. Editor only.	Ctrl + Shift + J
Add Task	Add a user defined task to the current text selection or selected element.	Alt + Enter
Expand Selection to	Enclosing Element: Selects the enclosing expression, block, method in the code. This action is aware of the Java syntax. It may not function properly when the code has syntax errors. (Arrow Up).	Alt + Shift + Arrow Keys
	Next Element: Selects the current and next element. (Arrow Right)	
	Previous Element: Selects the current and the previous element (Arrow Left)	

	Restore Last Selection: After an invocation of Expand Selection to restore the previous selection. (Arrow Down)	
Show Tooltip Description	Shows the value of a hover that would appear at the current cursor location. The dialog shown is scrollable and does not shorten descriptions.	F2
Content Assist	Opens a context assist dialog at the current cursor position to bring up Java code assist proposals and templates. See the Templates preference page for available templates (Window > Preferences > Java > Editor > Templates) and go to the Editor preference page (Window > Preferences > Java > Editor > Code Assist) for configuring the behavior of code assist.	Ctrl + Space
Quick Fix	If the cursor is located at a location with problem indication this opens a context assist dialog at the current cursor to present possible corrections.	Ctrl + 1
Parameter Hints	If the cursor is located at the parameter specification for method reference, this actions shows a hover with parameter types information.The parameter at the current cursor location is shown in bold.	Ctrl + Shift + Space

Navigate actions

Open	Tries to resolve the element referenced at the current code selection and opens the file declaring the reference.	F3
Open Type Hierarchy	Tries to resolve the element referenced at the current code selection and opens the element in the Type Hierarchy view. Invoked on elements, opens the type hierarchy of the element. Supported in the Java editor and views showing Java elements.	F4
Open External Javadoc	Opens the Javadoc documentation of the currently selected element or text selection. The location of the Javadoc of a JAR or a project is specified in the Javadoc Location property page on projects or JARs. Note that this external Javadoc documentation may not be up to date with the Javadoc specified in the current code. You can create Javadoc documentation for source files in a Java project using the Javadoc export wizard.	Shift + F2
Open Type	Brings up the Open Type selection dialog to open a type in the editor.The Open Type selection dialog shows all types existing in the workspace.	Ctrl + Shift + T
Open Type In Hierarchy	Brings up the Open Type selection dialog to open a type in the editor and the Type Hierarchy view. The Open Type selection dialog shows all types that exist in the workspace.	Ctrl + Shift + H
Show Outline	Opens the lightweight outliner for the currently selected type.	Ctrl + O
Go to Next Problem	Selects the next problem. Supported in the Java editor.	Ctrl + .

Go to Previous Problem	Selects the previous problem. Supported in the Java editor.	Ctrl + ,
Go to Last Edit Location	Reveal the location where the last edit occurred.	Ctrl + Q
Go to Line	Opens an a dialog which allows entering the line number to which the editor should jump to. Editor only.	Ctrl + L

Search actions

Search...	Opens the search dialog	Ctrl + H
Occurrences in File	Finds all occurrences of the selected Java element in its file	Ctrl + Shift + U

Project actions

Build All	Builds the all projects in the workspace. This is an incremental build, means that the builder analyzes the changes since the last time of build and minimizes the number of changed files.	Ctrl + B

Source actions

Comment	Comments out all lines containing the current selection.	Ctrl + /
Uncomment	Uncomments all lines containing the current selection.	Ctrl + \
Shift Right	Increments the level of indentation of the currently select lines. Only activated when the selection covers multiple lines or a single whole line.	Tab
Shift Left	Decrements the level of indentation of the currently select lines. Only activated when the selection covers multiple lines or a single whole line.	Shift + Tab
Format	Uses the code formatter to format the current text selection. The formatting options are configured on the Code Formatter preference page (Window > Preferences > Java > Code Formatter)	Ctrl + Shift + F
Organize Imports	Organizes the import declarations in the compilation unit currently open or selected. Unnecessary import declarations are removed, and required import declarations are ordered as specified in the Organize Import preference page (Window > Preferences > Java > Organize Import). Organize import can be executed on incomplete source and will prompt you when a referenced type name can not be mapped uniquely to a type in the current project. You can also organize multiple compilation units by invoking the action on a package or selecting a set of compilation units.	Ctrl + Shift + O

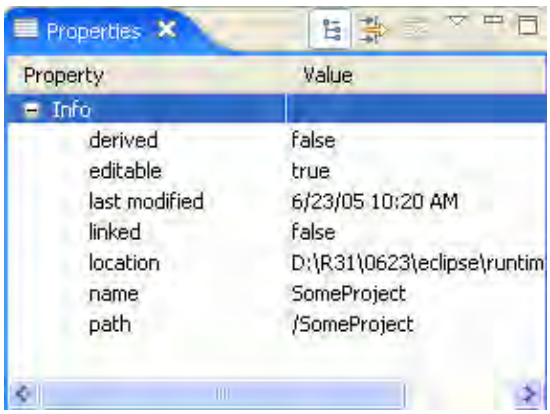
Add Import	Creates an import declaration for a type reference currently selected. If the type reference if qualified, the qualification will be removed if possible. If the referenced type name can not be mapped uniquely to a type of the current project you will be prompted to specify the correct type. Add Import tries to follow the import order as specified in the Organize Import preference page.	Ctrl + Shift + M

Refactor actions

Undo	Does an Undo of the last refactoring. The refactoring undo buffer is only valid as long as no other source changes than refactoring have been performed.	Alt + Shift + Z
Redo	Does a Redo of the last undone refactoring. The refactoring undo/redo buffer is only valid as long as no other source changes than refactoring have been performed.	Alt + Shift + Y
Rename	Starts the Rename refactoring dialog: Renames the selected element and (if enabled) corrects all references to the elements (also in other files). Is available on methods, fields, local variables, method parameters, types, compilation units, packages, source folders, projects and on a text selection resolving to one of these element types.	Alt + Shift + R
Move	Starts the Move refactoring dialog: Moves the selected elements and (if enabled) corrects all references to the elements (also in other files). Can be applied to one instance method (which can be moved to a component), one or more static methods, static fields, types, compilation units, packages, source folders and projects and on a text selection resolving to one of these element types.	Alt + Shift + V
Inline	Starts the Inline refactoring dialog. Inlines local variables, methods or constants. This refactoring is available on methods, static final fields and text selections that resolve to methods, static final fields or local variables.	Alt + Shift + I
Extract Method	Starts the Extract Method refactoring dialog. Creates a new method containing the statements or expression currently selected and replaces the selection with a reference to the new method. You can use Expand Selection from the Edit menu to get a valid selection range. This feature is useful for cleaning up lengthy, cluttered, or overly−complicated methods.	Alt + Shift + M
Extract Local Variable	Starts the Extract Variable refactoring dialog. Creates a new variable assigned to the expression currently selected and replaces the selection with a reference to the new variable. This refactoring is available on text selections that resolve to local variables. You can use Expand Selection from the Edit menu to get a valid selection range.	Alt + Shift + L

Properties view

This view displays property names and basic properties of a selected resource. Here is an example:



Toolbar buttons allow you to toggle whether to display properties by category and whether to filter advanced properties. Another toolbar button allows you to restore the selected property to its default value.

To see more detailed information about a resource than the Properties view gives you, right−click the resource name in one of the navigation views and select Properties from the pop−up menu.

CVS views

There are several CVS specific views, the CVS Repositories view, the CVS Resource History view, the CVS Console view and the Synchronize view.

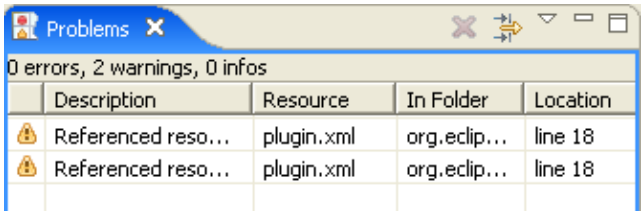
See Team Support in the Reference section for details on these views.

Related reference

CVS Repositories view
CVS Resource History View
CVS Console view
Synchronize view

Problems view

The Problems view displays system−generated errors, warnings, or information associated with a resource. These are typically produced by builders. For example, if you save a Java source file that contains syntax errors, the errors will automatically be logged in this view.



By default, the Problems view is included in the Resources perspective. To add it to the current perspective, click **Window > Show View > Other > Basic > Problems**.

The following icons are used by the Problems view:

Icon	Description
	Information
	Warning
	Error
	Delete
	Filter

The first column indicates whether the line item is a task or a compiler generated error, warning or info.

The Description column contains a description of the line item. You can edit the description of user−defined tasks by selecting **Properties** from the context menu.

The Resource and In Folder columns provide the name and location of the resource associated with each line item.

The Location column indicates the line number of the line item within its associated resource.

Toolbar

The toolbar of the Problems view includes the following buttons.

Delete
: Delete the selected line item.

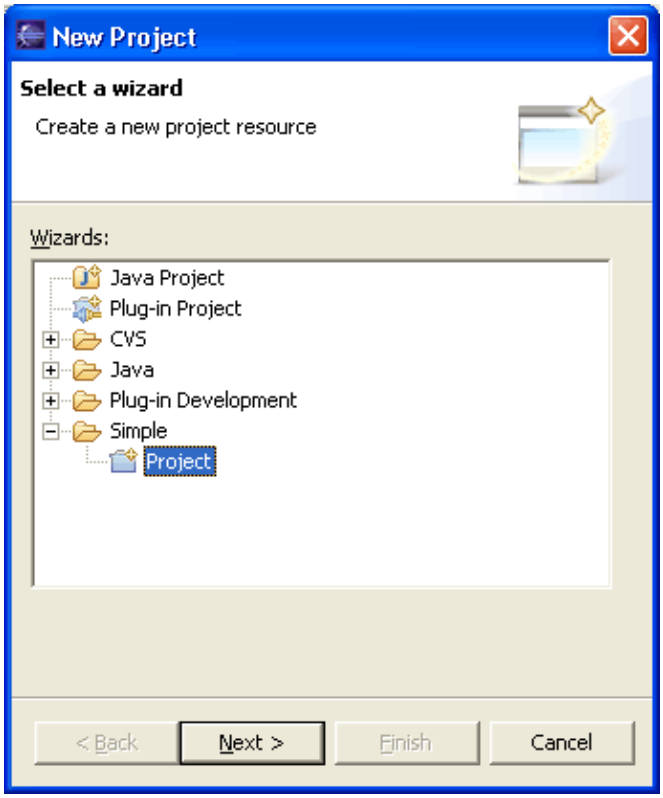
Filter
: Filter the view according to the type of item.

Menus

Click the icon at the left end of the view's title bar to open a menu of items generic to all views. Click the upside−down triangle icon to open a menu of items specific to the Problems view. Right−click inside the view to open a context menu.

New Project wizard

This wizard helps you create a new project in the Workbench.
When you first bring up the New Project wizard, you need to select the type of project you want to create. Select the Simple type if you want to create a generic project.



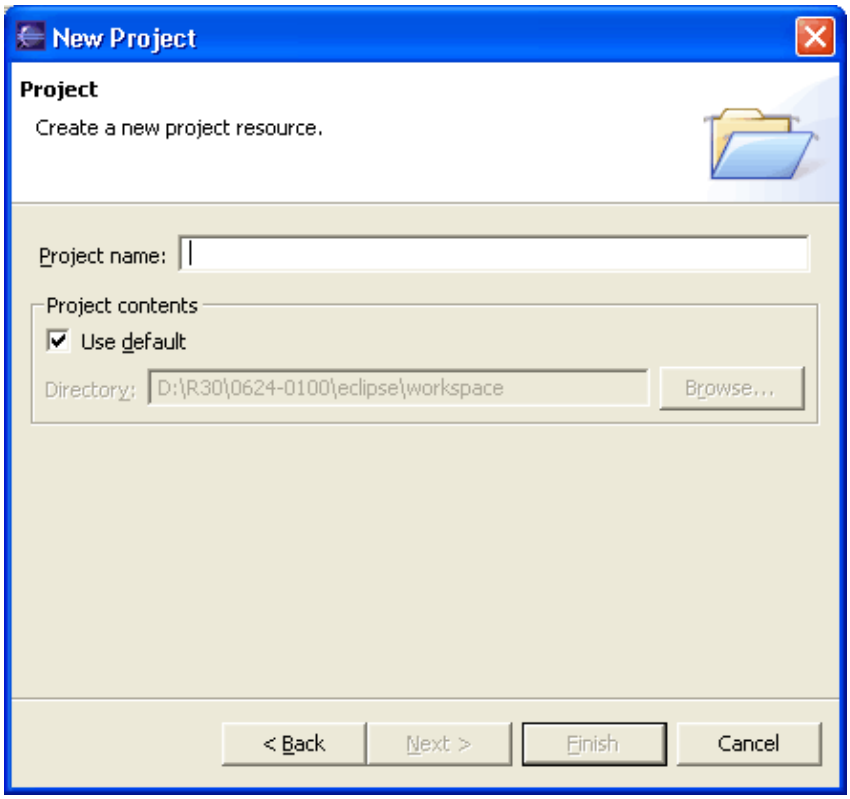
Create a New Project Resource Page

When you select Next, you will be presented with the New Project Resource Page, containing the following edit fields:

Field	Description	Default
Project Name	The name of the new project to be created.	<blank>
Location	The location in the file system where the project will be created. De−select "Use default location" to specify a location other than the default. You can type the new location or browse to select a file system location for the new project.	The workspace root directory

After you indicate a name and location for the project, you can either click ***Finish*** to create the project, or you can click ***Next*** to set up project dependencies on the Select Referenced Projects page.

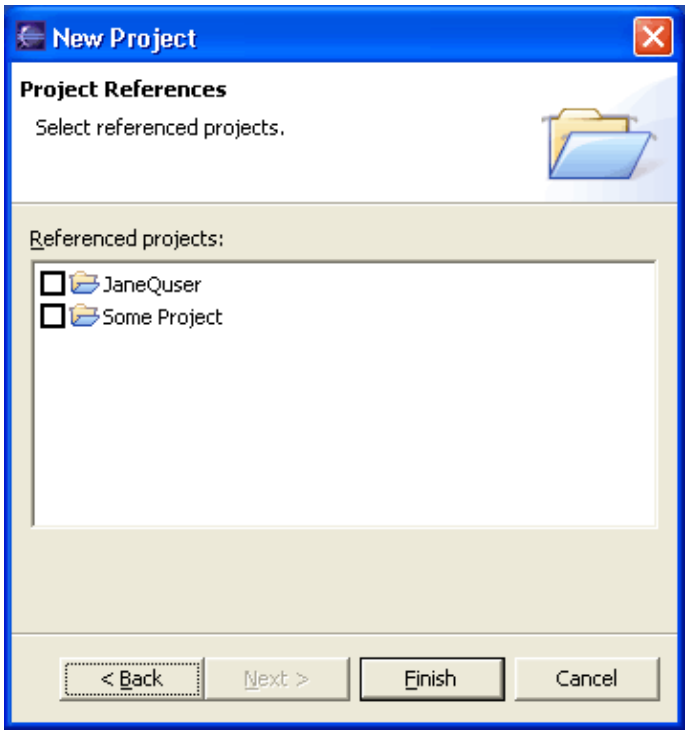
Here is what the New Project Resource page looks like:



Select Referenced Projects page

In the Referenced Projects list, you can set project dependencies for the new project. In the list of other projects in the Workbench, you can select one or more projects on which you want the new project to depend. Initially, no projects will be selected.
Click ***Finish*** when you are done to create the new project in the Workbench.

Here is what the Select Referenced Projects page looks like:



New Project perspective options

On the preferences page (***Window*** > ***Preferences*** > ***General*** > ***Perspectives***), you can change the way that new projects are initially displayed. For details on how to change this option see Workbench.

Related reference

Navigator View

CVS Wizards

There are several CVS specific wizards. These include the Import CVS Project wizard (which is also available from the New Project wizard) and the New CVS Repository Location wizard. There are also wizards for several CVS operations, such as committing and merging.

See Team Support in the Reference section for details on these wizards.

Related reference

CVS Checkout wizard
New CVS Repository Location wizard
Merge wizard

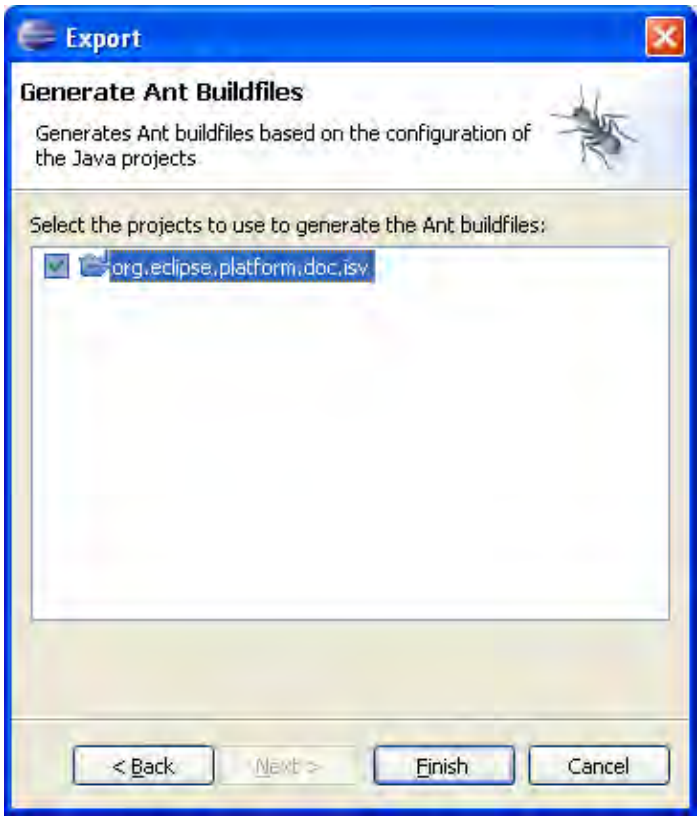
Export wizard

This wizard help you export resources from the Workbench.

When the Export wizard first comes up, you must choose what type of export to do:

Ant Buildfiles

Generates Ant buildfiles based on the configuration of the Java projects.

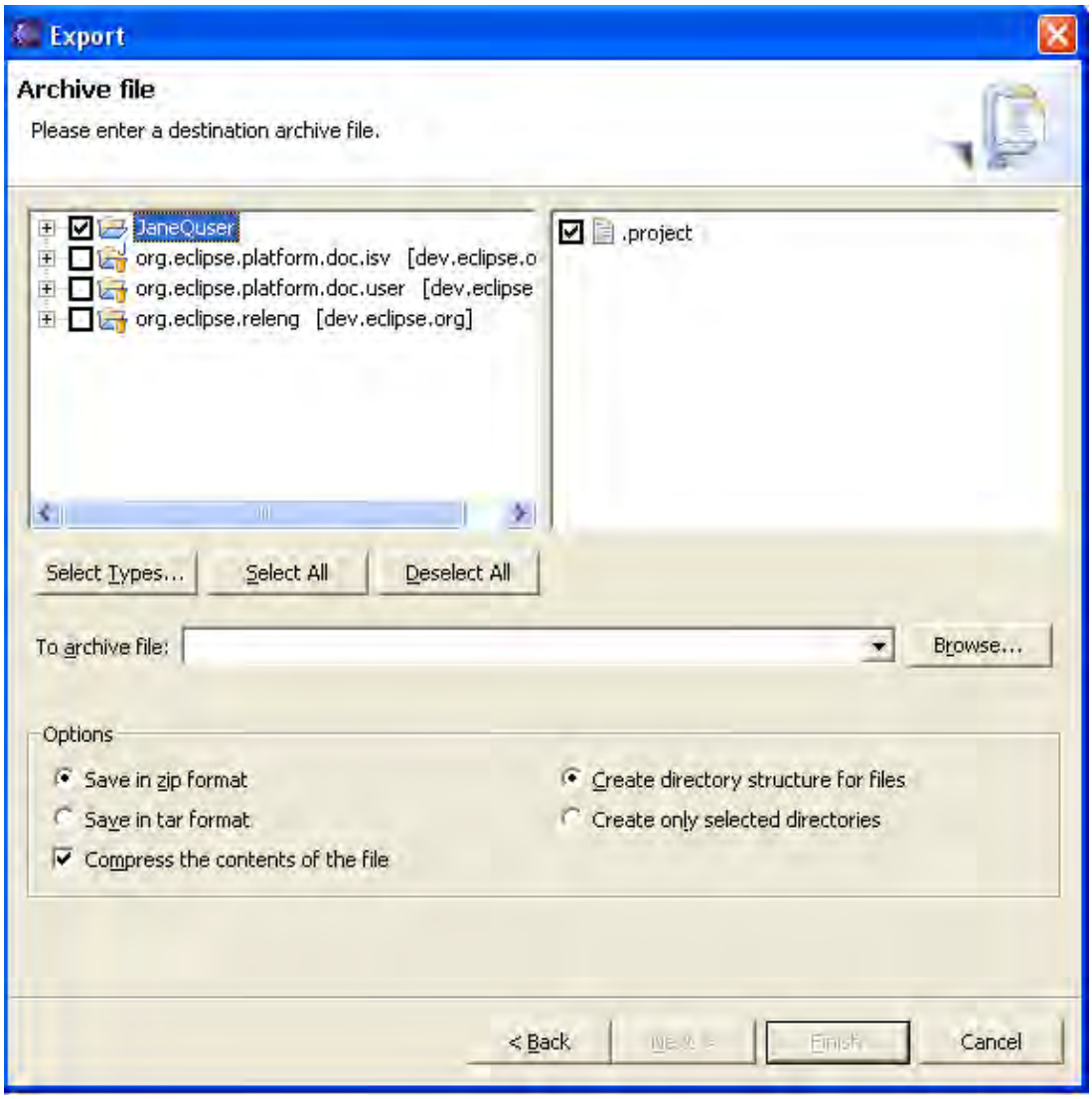


Export − Ant Buildfiles Options

Option	Description	Default
Select the projects to use to generate the buildfiles	The project (and resources within that project) to use to generate the buildfiles.	Java projects selected.

Archive File

If you choose this option, you will export files to an archive file.



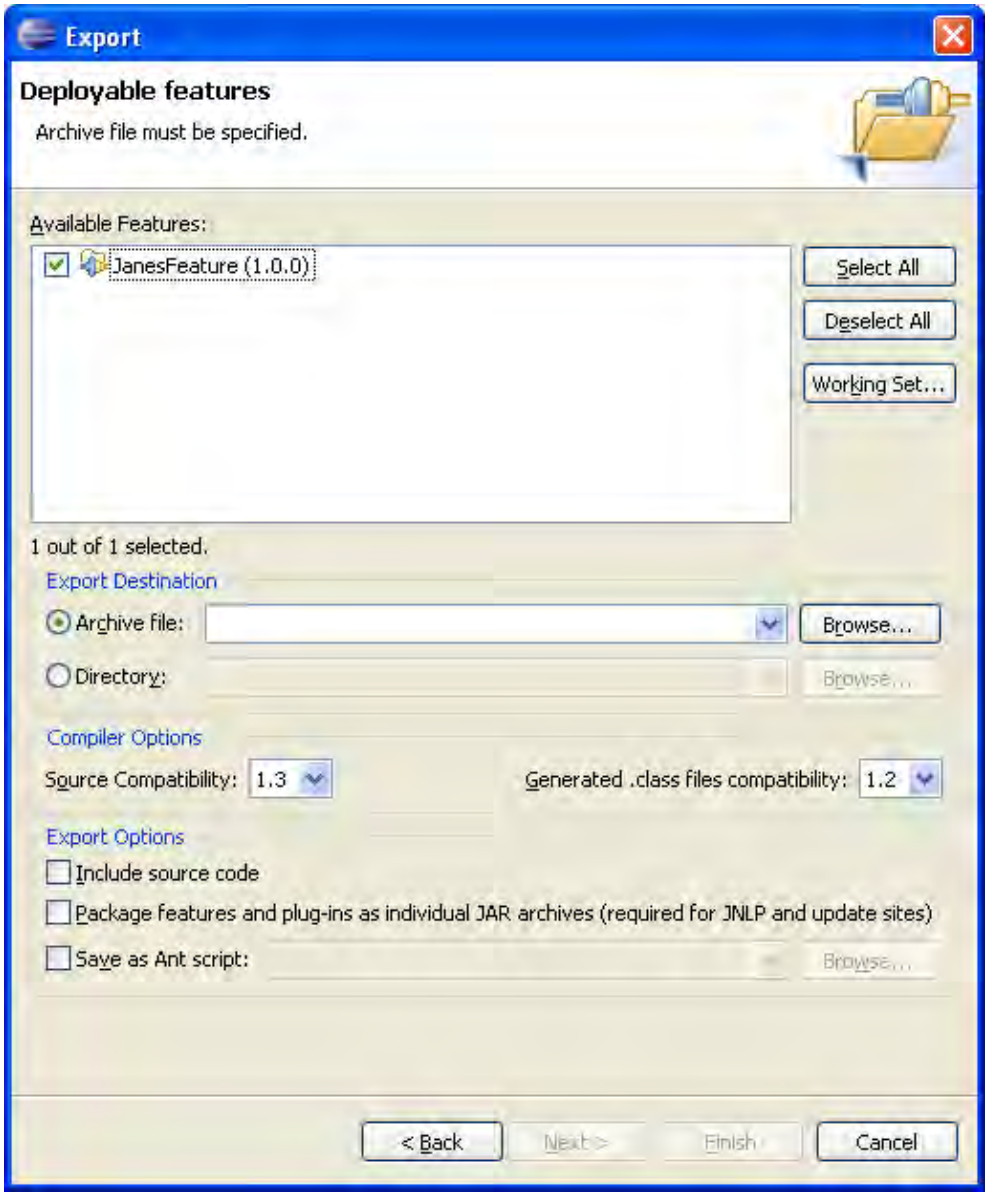
Export − Archive File Options

Option	Description	Default
Select resources to export	The project (and resources within that project) to export to an archive.	The project holding the selected resource
Select Types...	Dialog to select which file types to export. Use this to restrict the export to only certain file types.	*N/A*
Select All	Check off all resources for export.	*N/A*
Deselect All	Uncheck all resources.	*N/A*
Archive File	The path and name of an archive file into which the resources will be exported. Type the path, select a previous path from the drop down list, or Browse to select a path and file name on the file system.	The archive file of the previous export, or <blank>.
Compress the contents of the file	Compresses the contents (resources selected to be exported) in the archive that is created.	On

Overwrite existing file without warning	If the specified archive already exists in the file system, you will be prompted to overwrite the file. If you do not want to be prompted turn this option on.	Off
Create directory structure for files	Create hierarchy (folder) structure in the file system as it exists in the Workbench.	Off
Create only selected directories	Create hierarchy (folder) structure in the file system only for selected folders.	On

Deployable Features

Export the selected features in a form suitable for deploying in an Eclipse product.

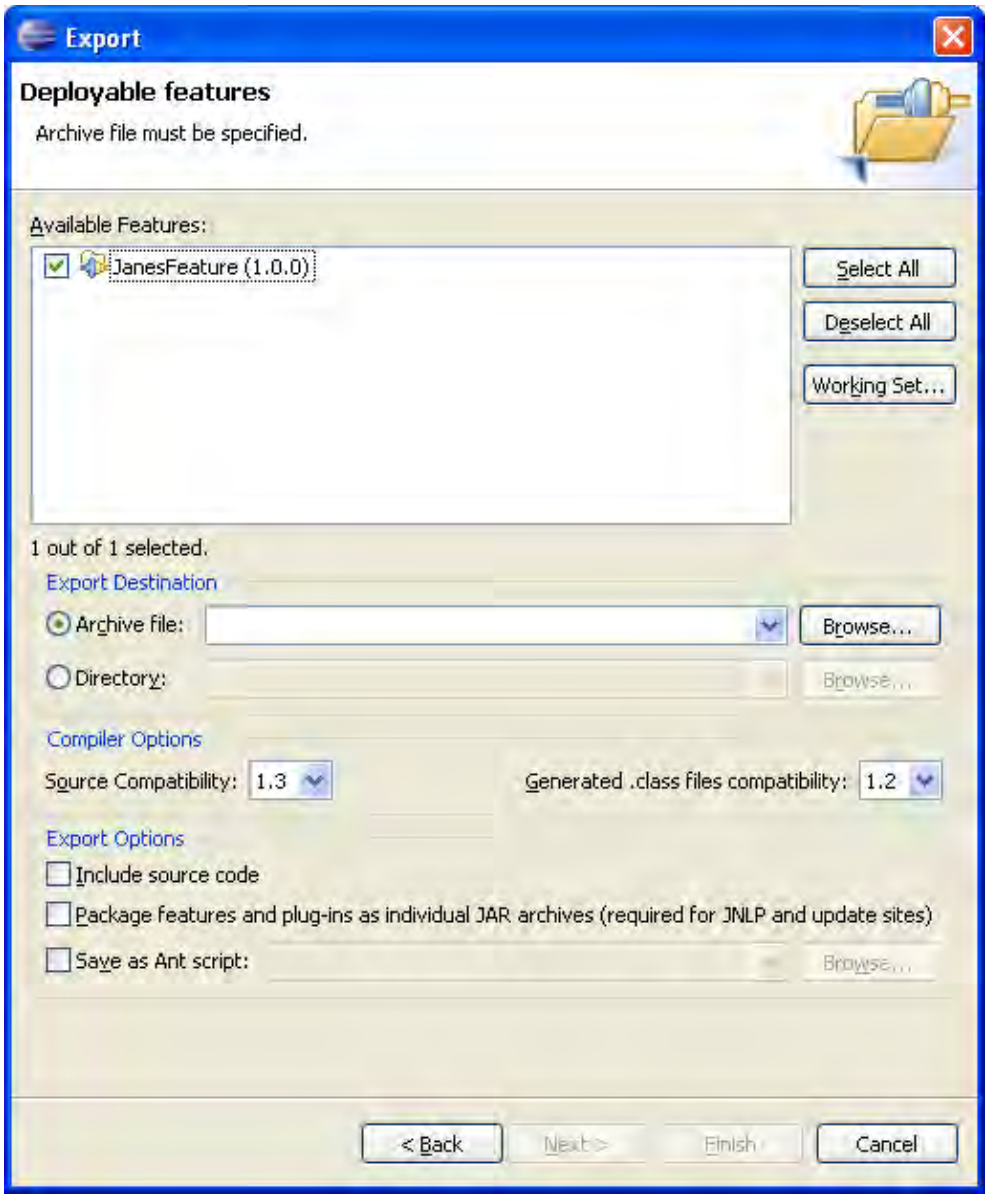


Export − Deployable Feature Options

Option	Description	Default
Available Features	The features in your workspace available for export.	Features selected.
Select All	Select all of the features listed.	
Deselect All	Clear all of the features listed.	
Working Set	Select a defined working set of features.	
Export Destination		
Archive File	Export to this archive file. Type the file, select a previous export file from the drop down list, or Browse to select a file.	<blank>
Directory	Export to this directory on the file system. Type the path, select a previous export path from the drop down list, or Browse to select a path.	<disabled>
Compiler Options		
Source Compatibility	The Java source compatibility level.	<1.3>
Generated .class files compatibility	The Java binary compatibility level.	<1.2>
Export Options		
Include source code	Include the source code in the deployed feature.	<unchecked>
Package features as individual JAR archives	Package features and plug−ins as individual JAR archives. This is required for JNLP and update sites.	<unchecked>
Save as Ant script	Generate an ant script to allow command line builds.	<unchecked>

Deployable Plug−ins and Fragments

Export the selected plug−ins and/or fragments in a form suitable for deploying in an Eclipse product.



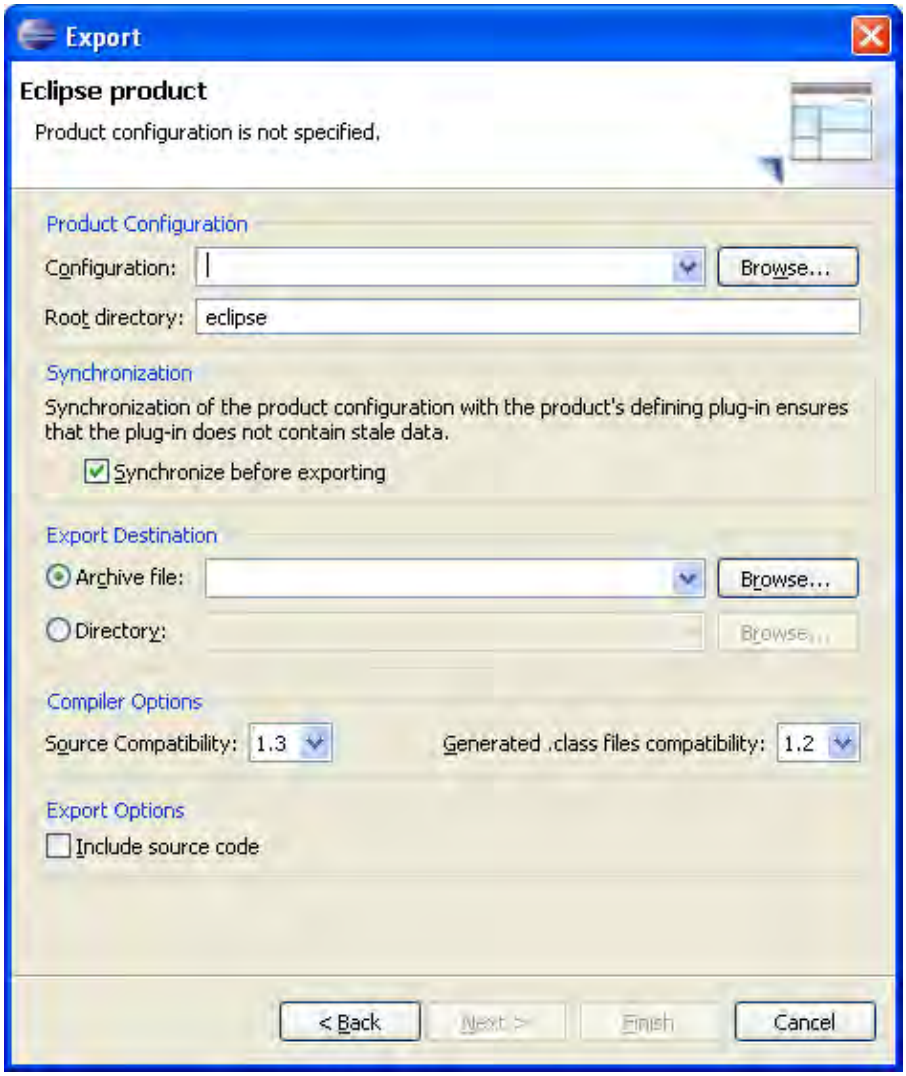
Export − Deployable Plug−ins and Fragments Options

Option	Description	Default
Available Plug−ins and Fragments	The plug−ins and fragments in your workspace available for export.	Plug−ins selected.
Select All	Select all of the plug−ins listed.	
Deselect All	Clear all of the plug−ins listed.	
Working Set	Select a defined working set of plug−ins.	
Export Destination		
Archive File	Export to this archive file. Type the file, select a previous export file from the drop down list, or Browse to select a file.	<blank>
Directory		<disabled>

	Export to this directory on the file system. Type the path, select a previous export path from the drop down list, or Browse to select a path.	
Compiler Options		
Source Compatibility	The Java source compatibility level.	<1.3>
Generated .class files compatibility	The Java binary compatibility level.	<1.2>
Export Options		
Include source code	Include the source code in the deployed feature.	<unchecked>
Package plug−ins as individual JAR archives	Package plug−ins and fragments as individual JAR archives.	<unchecked>
Save as Ant script	Generate an ant script to allow command line builds.	<unchecked>

Eclipse Product

Export an Eclipse product.

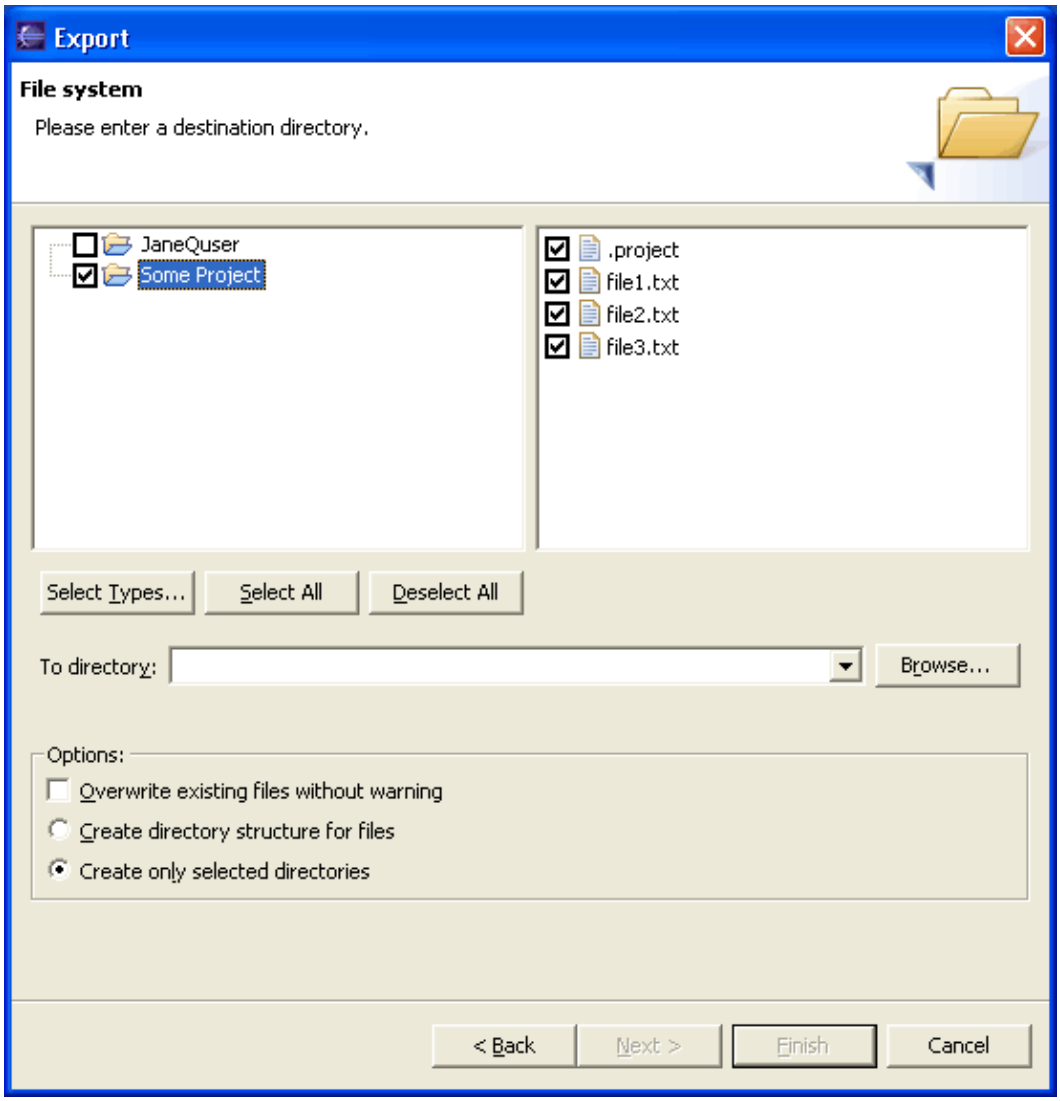


Export − Eclipse Product Options

Option	Description	Default
Product Configuration		
Configuration	The configuration to create the product. Type the configuration, select a previous configuration from the drop down list, or Browse to select a configuration.	<blank>
Root directory		**eclipse**
Synchronization		
Synchronize before exporting	Synchronization of the product configuration with the product's defining plug−in ensures that the plug−in does not contain stale data.	<checked>
Export Destination		
Archive File	Export to this archive file. Type the file, select a previous export file from the drop down list, or Browse to select a file.	<blank>
Directory	Export to this directory on the file system. Type the path, select a previous export path from the drop down list, or Browse to select a path.	<disabled>
Compiler Options		
Source Compatibility	The Java source compatibility level.	<1.3>
Generated .class files compatibility	The Java binary compatibility level.	<1.2>
Export Options		
Include source code	Include the source code.	<unchecked>

File System

If you choose this option, you will export files to the file system.



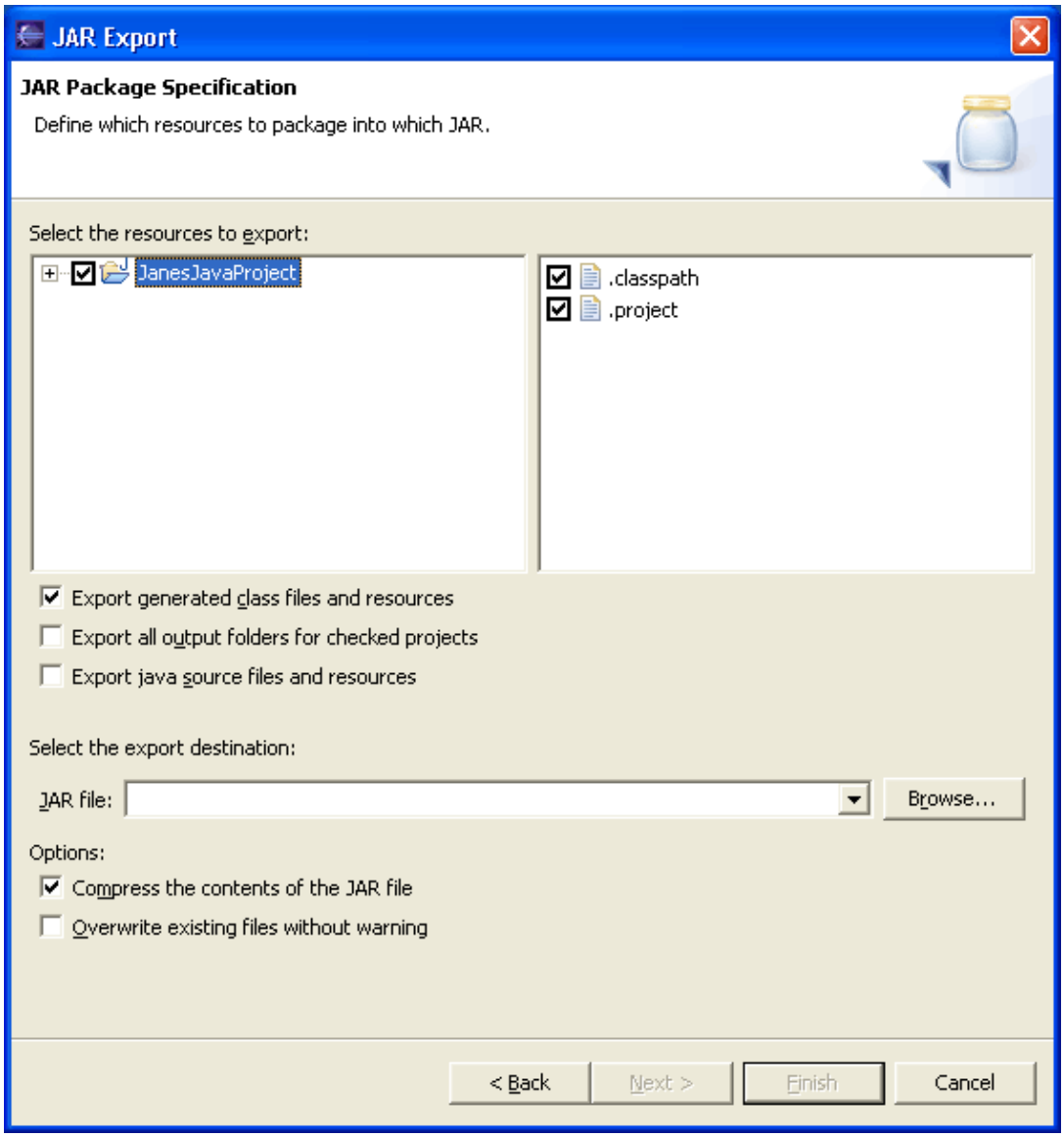
Export − File System Options

Option	Description	Default
Select resources to export	The project (and resources within that project) to export to the file system.	The project holding the selected resource
Select Types...	Dialog to select which file types to export. Use this to restrict the export to only certain file types.	*N/A*
Select All	Checks off all resources for export.	*N/A*
Deselect All	Uncheck all resources.	*N/A*
Directory	The directory on the file system into which the resources will be exported. Type the path, select a previous export path from the drop down list, or Browse to select a path.	The directory of the last export, or

		<blank>
Overwrite existing files without warning	Determines whether exporting a resource should silently overwrite a resource which already exists in the file system. If this option is off, you will be prompted before a given file is overwritten, in which case you can either overwrite the file, skip it, or cancel the export.	Off
Create directory structure for files	Create hierarchy (folder) structure in the file system as it exists in the Workbench.	Off
Create only selected directories	Create hierarchy (folder) structure in the file system only for selected folders.	On

Jar File

If you choose this option, you will export files to an JAR file.

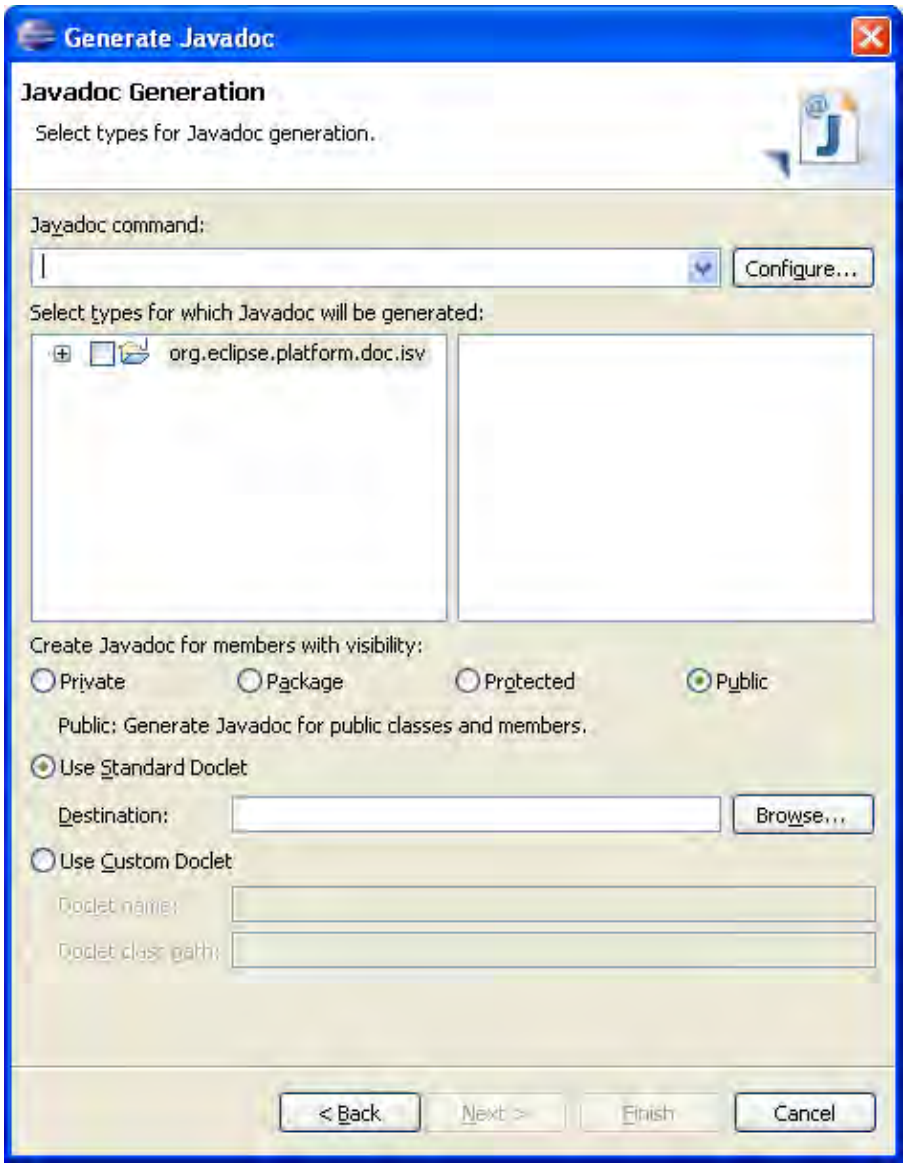


Export − JAR File Options

Option	Description	Default
Select packages to export	The project (and packages within that project) to export to a JAR file.	The project holding the selected resource
Export generated class files and resources	Export the generated bytecodes (.class files) and resource files contained in the selected packages and projects.	On
Export java source files and resources	Export the java source files (.java) and resource files contained in the selected packages and projects.	Off
Jar File	The path and name of a JAR file into which the resources will be exported. Type the path or Browse to select a path and file name on the file system.	The JAR file of the previous export, or <blank>.
Compress the contents of the JAR file	Compresses the contents (resources selected to be exported) in the JAR file that is created.	On
Overwrite existing files without warning	If the specified JAR file already exists on the file system, you will be prompted to overwrite the file. If you do not want to be prompted turn this option on.	Off

Javadoc

Generate Javadocs



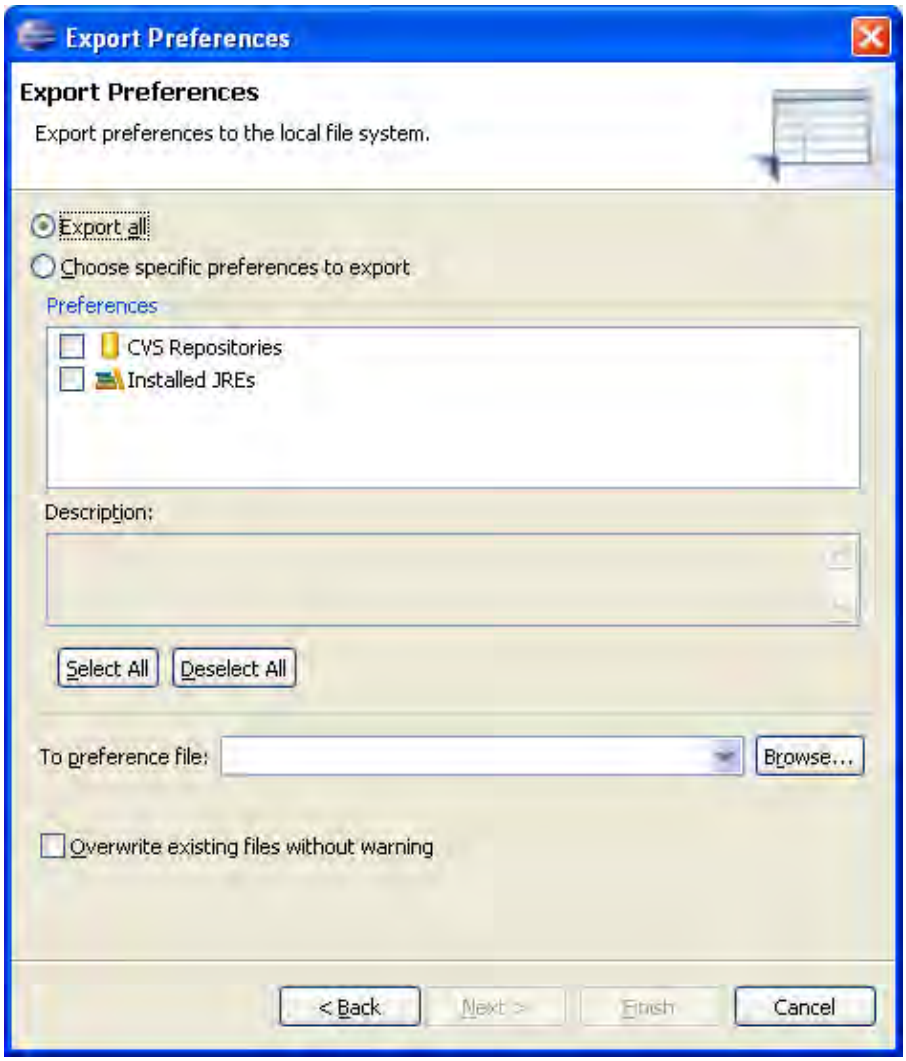
Export − Generate Javadoc Options

Option	Description	Default
Select types for which Javadoc will be generated	In the list, check or clear the boxes to specify exactly the types that you want to export to the JAR file. This list is initialized by the workbench selection. Only one project can be selected at once as only one project's classpath can be used at a time when running the Javadoc tool.	
Create Javadoc for members with visibility	• **Private**: All members will be documented	

	• **Package**: Only members with default, protected or public visibility will be documented • **Protected**: Only members with protected or public visibility will be documented • **Public**: Only members with public visibility will be documented (default)	
Use Standard Doclet	Start the Javadoc command with the standard doclet (default) • **Destination**: select the destination to which the standard doclet will write the generated documentation. The destination is a doclet specific argument, and therefore not enabled when using a custom doclet.	
Use Custom Doclet	Use a custom doclet to generate documentation • **Doclet name**: Qualified type name of the doclet • **Doclet class path**: Classpath needed by the doclet class	

Preferences

Export preferences to the local file system.



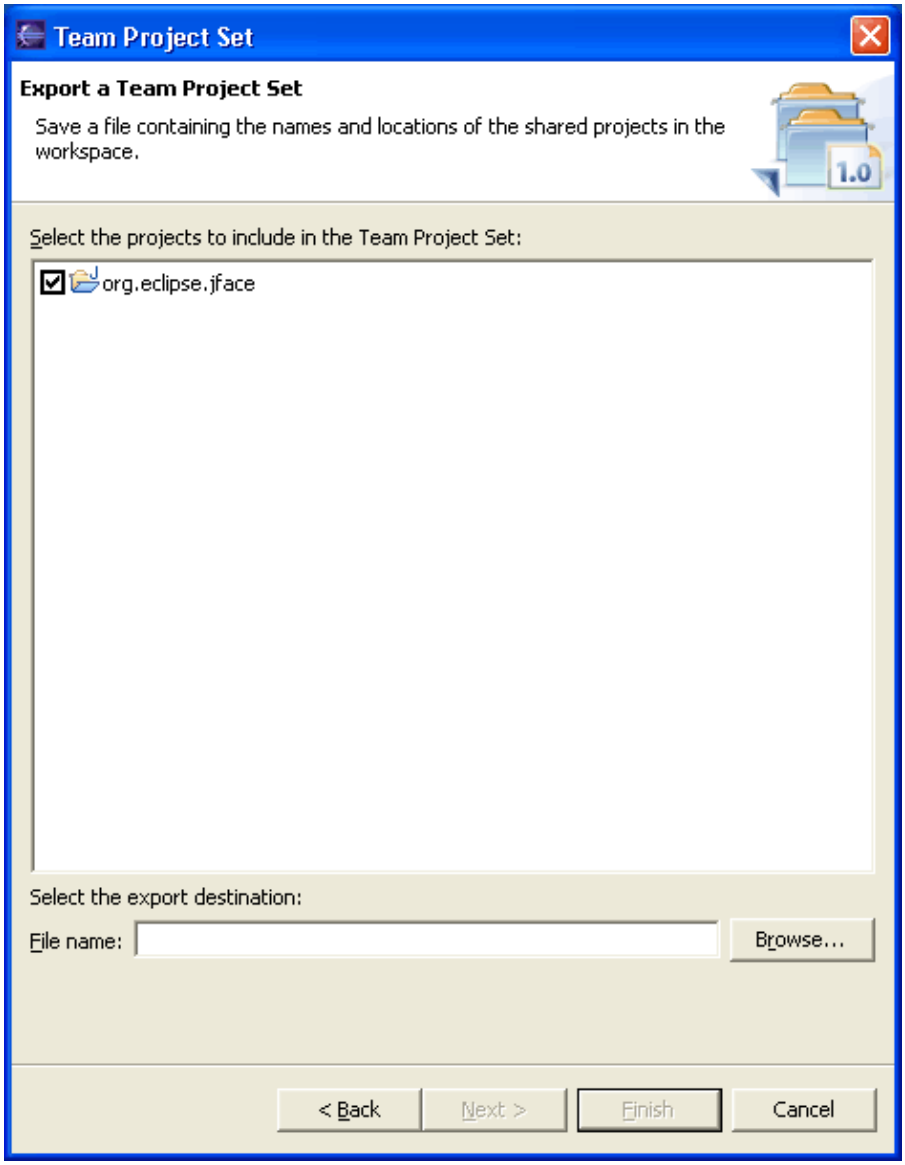
Export − Preference Options

Option	Description	Default
Export All	Export all of the preferences in this session.	<checked>
Choose specific preferences to export	Select preferences from this session to export, like CVS repository preferences.	<disabled>
Select All	Select all of the available preferences.	
Deselect All	Clear all of the available preferences.	
To preference file	A file on the file system to store the preferences. Type the file, select a previous export file from the drop down list, or Browse to select a file.	<blank>

Overwrite existing files without warning	Overwrite a pre−existing file.	<unchecked>

Team Project Set

Exports a description of the repository and version control information for a set of projects. This allows you to synchronize those projects correctly in a different workspace.



Export − Team Project Set Options

Option	Description	Default
Select projects	The project(s) to export to the team project set file.	The selected projects
File name	The path and name of the file to export to.	<blank>

Workbench User Guide

The Help view displays help related to using the Workbench. See Help window on how to navigate through the contents of the Help view.

If you select Workbench User Guide in the list of books displayed in the Help window, you will see help topics related to using the Workbench in the Contents tab. The Workbench User Guide is broken down into four main sections, described below.

Getting started

This section contains tutorials that will help you when you start using the Workbench.

Concepts

Concepts are high level descriptions of the schema and functions of the Workbench. This section helps provide a general understanding of how the Workbench functions. For example, the Concepts section includes a discussion of what perspectives and views are and how they relate to one another.

Tasks

Task descriptions are step by step instructions for performing specific actions and tasks in the Workbench. For example, the Tasks section contains step by step instructions for creating a repository location, and for importing a file from the file system into the Workbench.

Reference

Reference materials are helpful resources that will assist you while you are using the Workbench. This includes descriptions of various wizards, dialogs, and fields as well as Workbench resources such as specific views and perspectives.

The Reference section also includes a glossary of terms that you might find useful while using the Workbench.

Working with cheat sheets

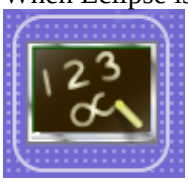
Eclipse provides cheat sheets to guide you through some of its application development processes. Each cheat sheet is designed to help you complete some task, and it lists the sequence of steps required to help you achieve that goal. As you progress from one step to the next, the cheat sheet will automatically launch the required tools for you. If there is a manual step in the process, the step will tell you to perform the task and click a button in the cheat sheet to move on to the next step. Also, relevant help information to guide you through a task is retrieved in a single click so that lengthy documentation searches will no longer be required.

Launching a cheat sheet

There are two ways that cheat sheets can be accessed, on the Welcome page or launched from the Workbench.

Accessing a cheat sheet from the Welcome page

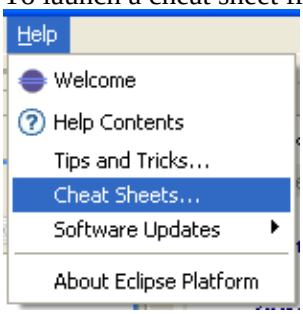
1. When Eclipse is first started, the Welcome page is displayed, on this page select ***Tutorials***.



2. Then, from the Tutorials page select a cheat sheet.

Launching a cheat sheet from the Workbench

1. To launch a cheat sheet from the Workbench, select ***Help*** > ***Cheat Sheets*** from the menu bar.



2. The available cheat sheets are listed, select one and click ***OK***.

The cheat sheet opens as a view. At any time, only one cheat sheet is open and active. When you launch a cheat sheet, any opened cheat sheet is closed before the new one is opened. The completion status of closed cheat sheet is saved.

The cheat sheet has a toolbar at the top right edge. These icons appear in the toolbar:

- Collapses all the expanded steps except the current step or expands steps to the last expanded state. Click to toggle between these two states.
- Allows you to select and open another cheat sheet. The completion status of the active cheat sheet is saved. Then, the active cheat sheet is closed and the selected cheat sheet is opened.
- Hides the cheat sheet.

- Saves the completion status of the active cheat sheet and closes it.

Starting the cheat sheet

Each cheat sheet has a list of steps and it always begins with an Introduction step. When you launch a fresh cheat sheet, the Introduction step is expanded so that you can read a brief description of the cheat sheet. To start working with the cheat sheet, click ***Click to Begin*** in that step. The next step is expanded and highlighted. You should also see one or more actions buttons, such as ***Click to Perform*** in the highlighted step. You can now begin working through the tasks using the cheat sheet. At any time, the only highlighted step in the cheat sheet is the current step.

Restarting the cheat sheet

Any time after starting a cheat sheet, you can restart from the first step by clicking ***Click to Restart*** in the Introduction step. If you have already created some artifacts, you will have to manually clean up the workspace before restarting the cheat sheet.

Progressing through the steps

In the current step, when you click ***Click to Perform***, a tool (which can be a wizard), will be launched and you will be required to work with that tool. When you finish working with that tool, the next step is automatically highlighted and it becomes the current step. When the current step is a manual task, you will need to perform the work and click ***Click to Complete*** to move to the next step. A check mark appears in the left margin of each completed step.

Getting help information for tasks

To get step−by−step instructions for that step, click the help link in the step before you click ***Click to Perform***, and the step−by−step instructions on how to work with that tool will be displayed in the Help window.

Additional help for entry fields in the tool or wizard may be available by focusing on the field (use the ***Tab*** key to position to that entry) and pressing ***F1***.

Skipping a step

If a current step has a ***Click to Skip*** option, then it is an optional step. You must click ***Click to Skip*** to skip the current step, when you do, the step will have the skip mark in the left margin. If the task does not present ***Click to Skip***, you must perform that step and you cannot skip it.

Redoing a step

You can redo any step that you may have completed or skipped in the current cheat sheet. To redo the step, expand the step by clicking its expand icon and then clicking ***Click to Redo***. After redoing a step, the cheat sheet will continue from the redo step.

Closing the cheat sheet

When you finish the last step in a cheat sheet, it automatically restarts. You can also close the active cheat sheet by clicking the close icon  in the cheat sheet's toolbar. The active cheat sheet saves its completion status when it is closed so that you can continue where you left off at a later time.

File menu

This menu allows you to create, save, close, print, import, and export Workbench resources and exit the Workbench itself.

New

This command creates new resources.

Open File

This command allows you to open files that do not reside in the workspace in the text editor.

Close

This command closes the active editor.

Close All

This command closes all open editors.

Save

This command allows you to save the contents of the active editor.

Save As

This command allows you to save the contents of the active editor under another file name or location.

Save All

This command saves the contents of all open editors.

Revert

This command replaces the contents of the active editor with the previously saved contents.

Move

This command moves the currently selected resources to a different location.

Rename

This command changes the name of the currently selected resource.

Refresh

This command refreshes the resource with the contents in the file system.

Convert Line Delimiters To

These commands alter the line delimiters for the selected files in the active part.

Print

This command prints the contents of the active editor.

Switch workspace

This command allows you to switch to a different workspace. This will restart the workbench.

Import

This option launches the import wizard, which allows you to add resources to the Workbench.

Export

This option launches the export wizard, which allows you to export resources from the Workbench.

Properties

This command opens the properties dialog for the currently selected resource.

Recent file list

A list of the most recently accessed files in the Workbench is maintained at the bottom of the File menu. Any of these files can be opened from the File menu by simply selecting the file name.

Exit

This command closes and exits the Workbench.

Edit menu

This menu helps you manipulate resources in the editor area.

Undo

This command reverses your most recent editing action.

Redo

This command re−applies the editing action that has most recently been reversed by the Undo action.

Cut

This command removes the selection and places it on the clipboard.

Copy

This command places a copy of the selection on the clipboard.

Paste

This command places the text or object on the clipboard at the current cursor location in the currently active view or editor.

Delete

This command removes the current selection.

Select All

This command selects all text or objects in the currently active view or editor.

Find/Replace

This command allows you to search for an expression in the active editor, and optionally replace the expression with a new expression.

Find Next

This command allows you to search for the next occurrence of the current selection, or for the next occurrence of the most recent expression found using the Find/Replace action.

Find Previous

This command allows you to search for the previous occurrence of the current selection, or for the previous occurrence of the most recent expression found using the Find/Replace action.

Incremental Find Next

This command allows you to search for expressions in the active editor. As you type the search expression, it will incrementally jump to the next exact match in the active editor. While in this mode, the up and down cursor keys can be used to navigate between matches, and the search can be cancelled by pressing left or right cursor keys, the enter key, or the escape key.

Incremental Find Previous

This command allows you to search for expressions in the active editor. As you type the search expression, it will incrementally jump to the previous exact match in the active editor. While in this mode, the up and down cursor keys can be used to navigate between matches, and the search can be cancelled by pressing left or right cursor keys, the enter key, or the escape key.

Add Bookmark

This command adds a bookmark in the active file on the line where the cursor is currently displayed.

Add Task

This command adds a task in the active file on the line where the cursor is currently displayed.

Word Completion

This action will attempt to complete the word currently being entered in the active editor.

Set Encoding

This action launches a dialog that allows you to change the file encoding used to read and write the file in the active editor.

Navigate menu

This menu allows you to locate and navigate through resources and other artifacts displayed in the Workbench.

Go Into

This command refocuses the active view so that the current selection is at the root. This allows web browser style navigation within hierarchies of artifacts.

Go To

- **Back**: This command displays the hierarchy that was displayed immediately prior to the current display. For example, if you Go Into a resource, then the Back command in the resulting display returns the view to the same hierarchy from which you activated the Go Into command. This command is similar to the Back button in an HTML browser.
- **Forward**: This command displays the hierarchy that was displayed immediately after the current display. For example, if you've just selected the Back command, then selecting the Forward command in the resulting display returns the view to the same hierarchy from which you activated the Back command. This command is similar to the Forward button in an HTML browser.
- **Up one level**: This command displays the hierarchy of the parent of the current highest−level resource.
- **Resource**: This command allows you to navigate quickly to a resource. For more information see the links to related tasks below.

Open Resource

This command displays a dialog that lets you select any resource in the workspace to open it in an editor. For more information see the links to related tasks below.

Show In

This sub−menu is used to find and select the currently selected resource in another view. If an editor is active, these commands are used to select the resource currently being edited in another view.

Next

This command navigates to the next item in a list or table in the active view. For example, when the search results view is active, this navigates to the next search result.

Previous

This command navigates to the previous item in a list or table in the active view. For example, when the search results view is active, this navigates to the previous search result.

Last Edit Position

This command allows you to jump the last edit position.

Go to Line

This command allows you to jump to a specific line in the active editor.

Back

This command navigates to the previous resource that was viewed in an editor. Analogous to the **Back** button on a web browser.

Forward

This command navigates to undo the effect of the previous **Back** command. Analogous to the **Forward** button on a web browser.

Related tasks

Finding a resource quickly

Help menu

This menu provides help on using the Workbench.

Welcome

This command will open the welcome content.

Help Contents

This command displays the help contents in a help window or external browser. The help contents contains help books, topics, and information related to the Workbench and installed features.

Search

This command displays the help view opened on the Search page.

Dynamic Help

This command displays the help view opened to Related Topics page.

Key Assist ...

This command will display a list of key bindings

Tips and Tricks ...

This command will open a list of interesting productivity features that you may not have discovered.

Cheat Sheets ...

This command will open the cheat sheet selection dialog.

Software Updates

This group of commands allows you to update your product and to download and install new features.

About

This command displays information about the product, installed features, and available plug−ins.

Navigator view icons

The following icons can appear in the navigation views:

Navigator view icons

Icon	Description
	Project (open)
	Folder (open)
	Project (closed)
	Generic File

Editor area marker bar

The following markers can appear in the marker bar (to the left of the editor area):

Marker Bar Markers

Icon	Description
	Bookmark
	Breakpoint
	Task marker
	Search result
	Error marker
	Warning marker
	Information marker

Tasks view

The following markers can appear in the Tasks view:

Tasks View Icons

Icon	Description	Icon	Description
	High priority task		Low priority task
	Completed task		

Tips and Tricks

The following tips and tricks give some helpful ideas for increasing your productivity. They are divided into the following sections:

- Workbench
- Ant
- Help
- Team − CVS

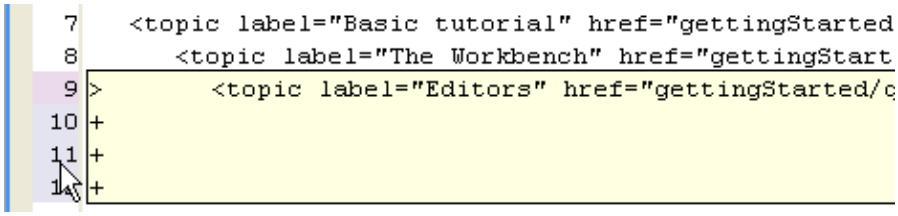
Workbench

Now, where was I?	Workbench editors keep a navigation history. If you open a second editor while you're editing, you can press **Navigate > Backward** (Alt+Left Arrow, or the back arrow on the workbench toolbar) to go back to the last editor. This makes working with several open editors a whole lot easier.
Finding a string incrementally	Use **Edit > Incremental Find Next** (Ctrl+J) or **Edit > Incremental Find Previous** (Ctrl+Shift+J) to enter the incremental find mode, and start typing the string to match. Matches are found incrementally as you type. The search string is shown in the status line. Press Ctrl+J or Ctrl+Shift+J to go to the next or previous match. Press Enter or Esc to exit incremental find mode.
Go to last edit location	**Navigate > Go to Last Edit Location** (Ctrl+Q) takes you back to the place where you last made a change. A corresponding button marked is shown in the toolbar. If this toolbar button does not appear in your perspective, you can add it by selecting **Window > Customize Perspective > Other > Editor Navigation**.
Shortcuts for manipulating lines 3.0	All text editors based on the Eclipse editor framework support editing functions, including moving lines up or down (Alt+Arrow Up and Alt+Arrow Down), copying lines (Ctrl+Alt+Arrow Up and Ctrl+Alt+Arrow Down), inserting a new line above or below the current line (Ctrl+Shift+Enter and Shift+Enter), and converting to lowercase or uppercase (Ctrl+Shift+Y and Ctrl+Shift+X).

Quick Diff: seeing what has changed as you edit

3.0

Quick Diff provides color−coded change indication while you are typing. It can be turned on fo text editors using either the ruler context menu, Ctrl+Shift+Q or for all new editors on the **General > Editors > Text Editors > Quick Diff** preference page. The colors show additions, deletions, and changes to the editor buffer as compared to a reference, for example, the contents of the file on disk or its latest CVS revision.

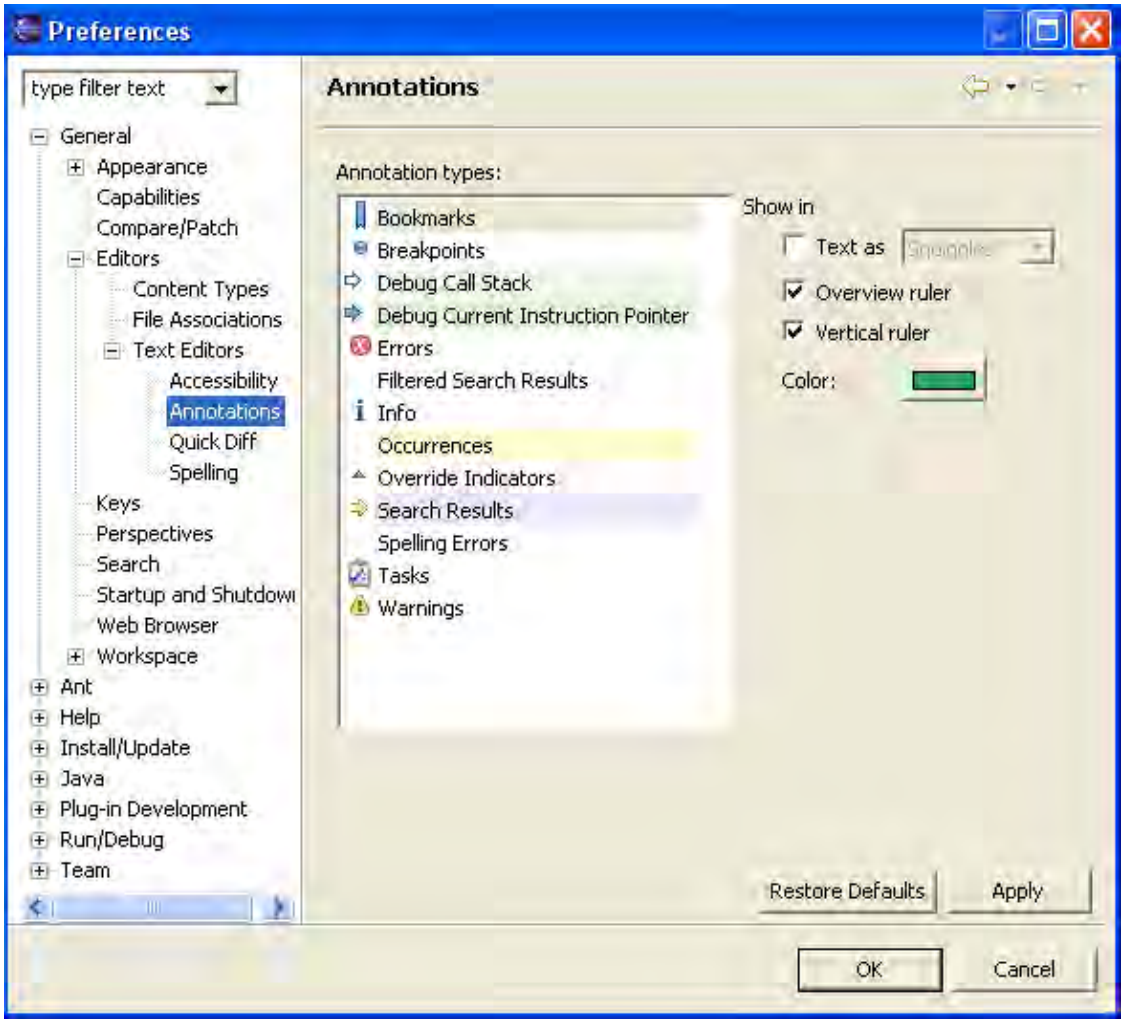


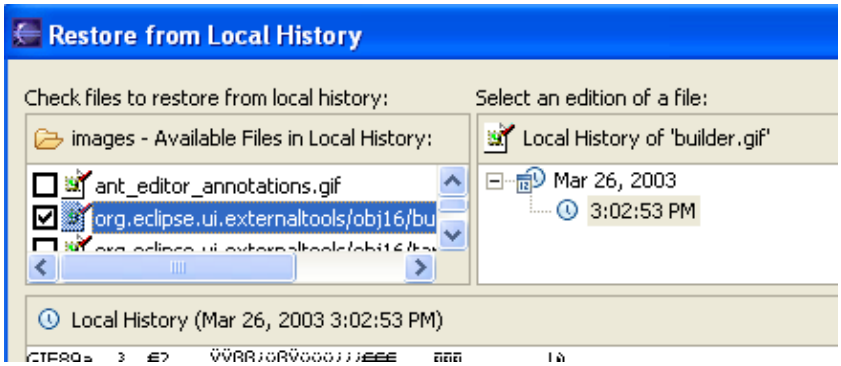
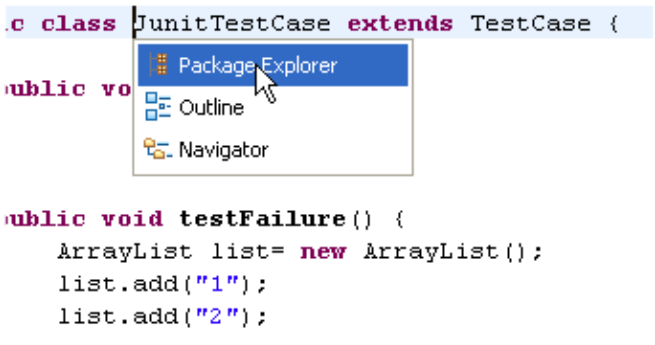
When the mouse cursor is placed over a change in the vertical ruler, a hover displays the original content, which can be restored using the ruler's context menu. The context menu also allows you to switch between the references and enable/disable Quick Diff.

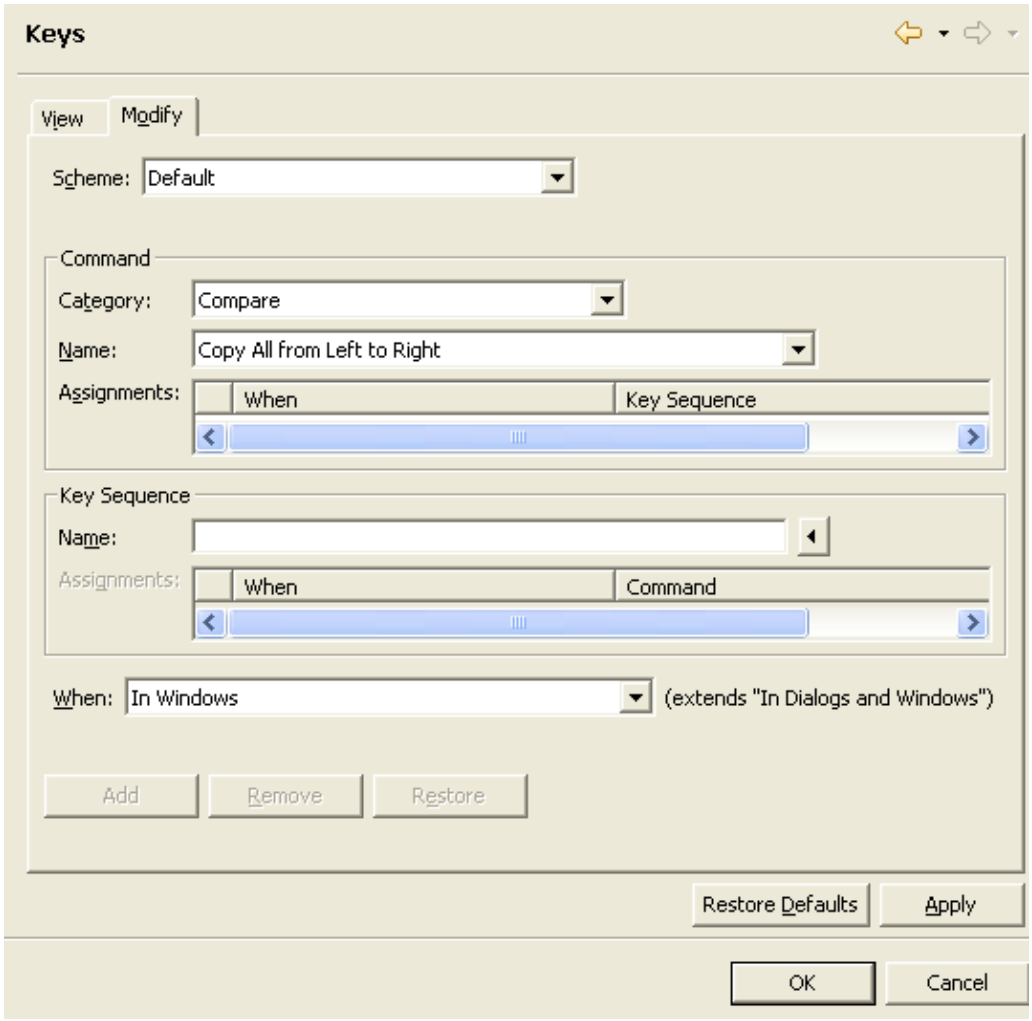
Customizing the presentation of annotations

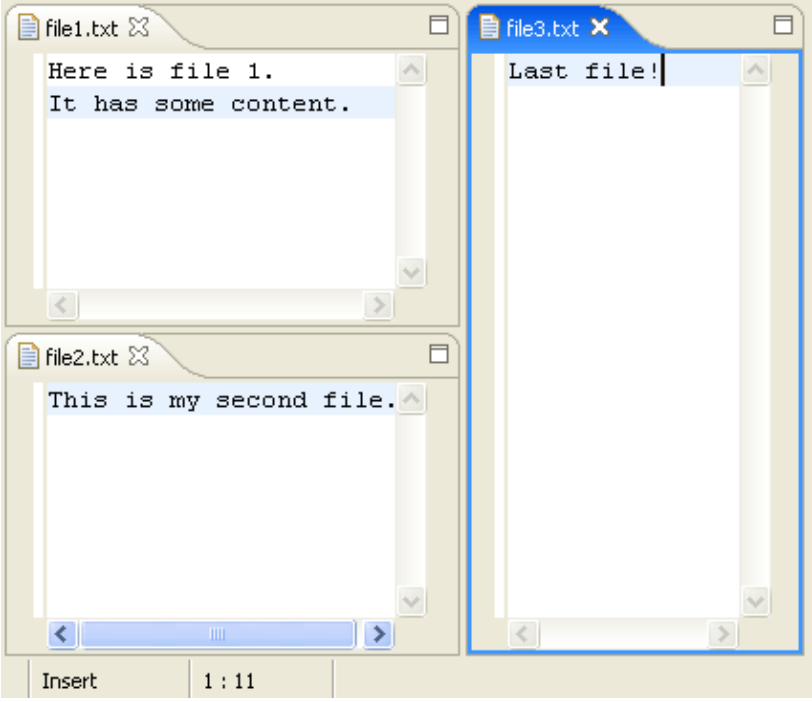
3.0

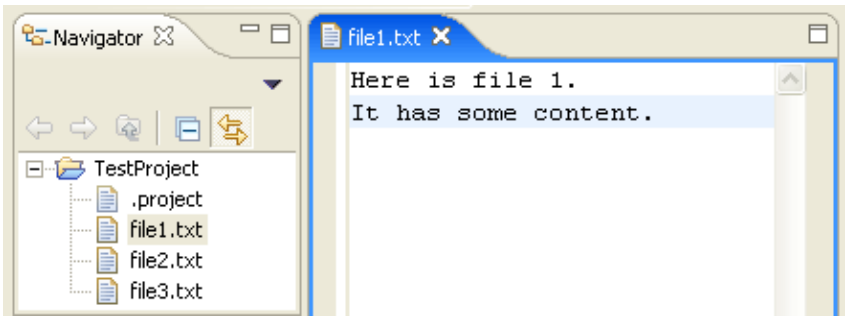
You can customize the presentation of annotations in editors on the **General > Editors > Text Editors > Annotations** preference page:

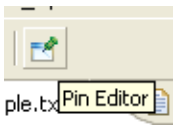


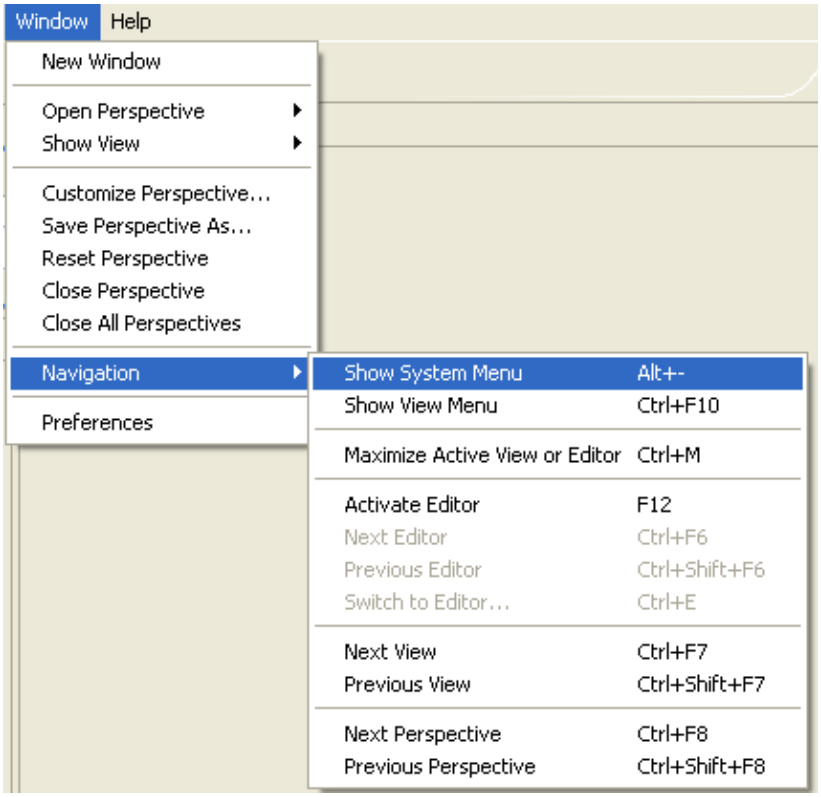
Managing screen real estate with fast views	Use fast views to free up screen real estate while keeping views easily accessible. Clicking on the icon for a fast view temporarily reveals it over top of the other views. The fast view retracts as soon you click outside of it. The **Fast View** command in the view's system menu toggles whether it is a fast view. You can also create a fast view by dragging a view onto the shortcut bar at the left.
Opening editors using drag and drop	You can open an editor on an item by dragging the item from a view like the Navigator or Package Explorer and dropping it over the editor area.
Restoring deleted resources	Select a container resource and use **Restore from Local History** to restore deleted files. You can restore more than one file at one time. 
Like to start afresh each session?	A setting on the **General > Editors** preference page closes all open editors automatically whenever you exit. This makes start−up cleaner and a bit faster.
Better UI for editor / view synchronization	The **Navigate > Show In** command provides a uniform way to navigate from an open editor to a view showing the corresponding file (e.g., in the resource Navigator view), or from a file selected in one view to the same file in a different view (e.g., from the resource Navigator view to the Packages Explorer view). Typing Alt+Shift+W opens a shortcut menu with the available view targets. 

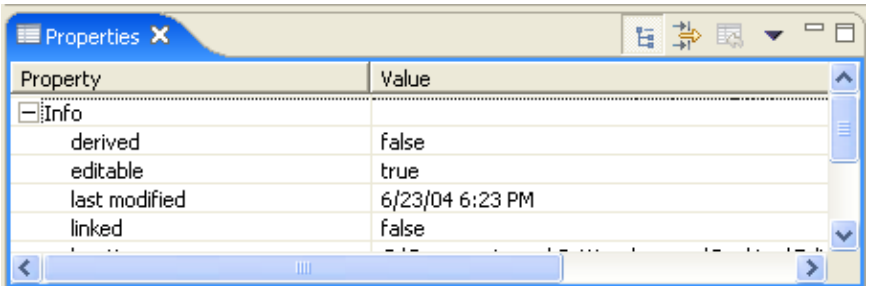
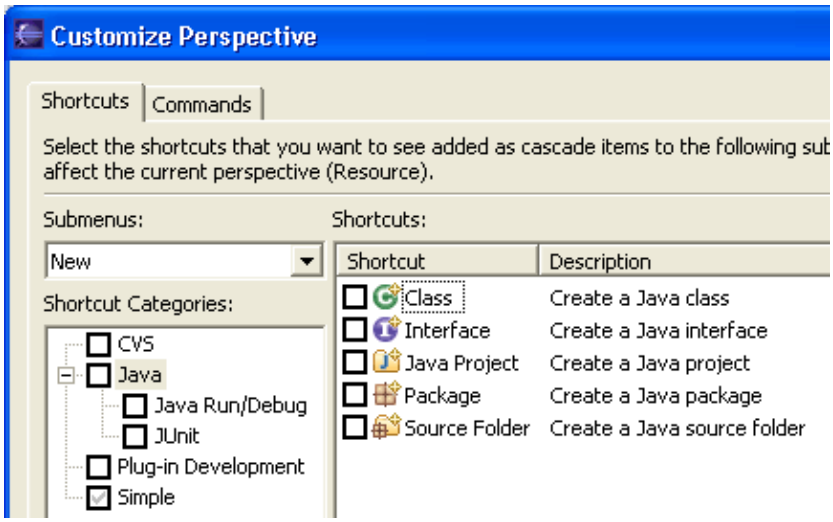
User customizable key bindings	If you find yourself repeatedly doing some command, you might be able to streamline things by assigning a key sequence to trigger that command. Assigning new key bindings, and viewing existing bindings, is done from the **General** > **Keys** preference page. 
Faster workspace navigation	**Navigate** > **Open Resource** (Ctrl+Shift+R) brings up a dialog that allows you to quickly locate and open an editor on any file in the workspace. In the same vein, **Navigate** > **Go To** > **Resource** expands and selects the resource in the Navigator view itself, if it has focus.
Tiling the editor work area	You can use drag and drop to modify the layout of your editor work area. Grab an editor tab and drag it to the edge of the editor work area. The arrow dock icons (e.g., ⬅) indicate which way the editor work area will split.

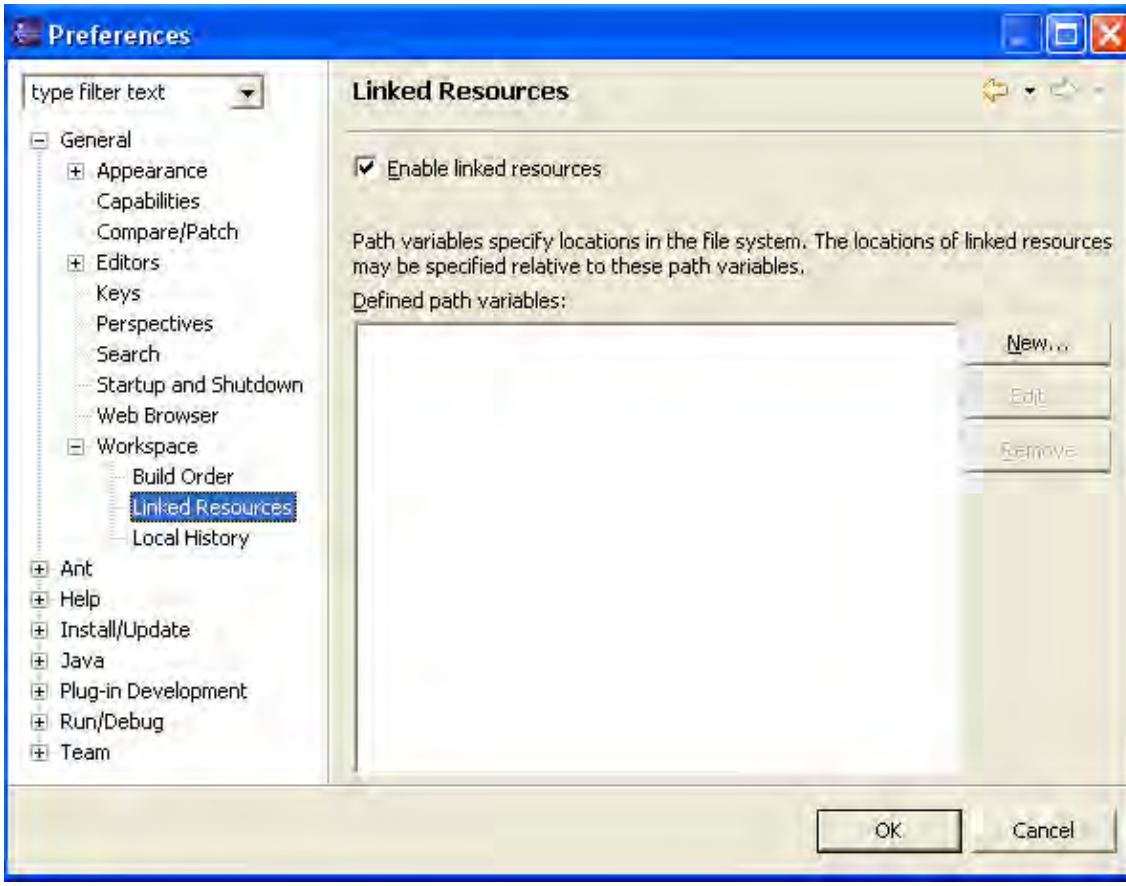


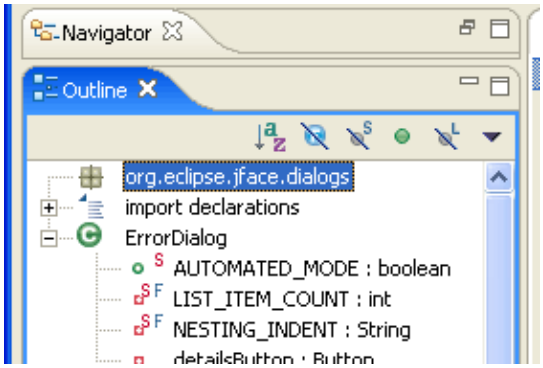
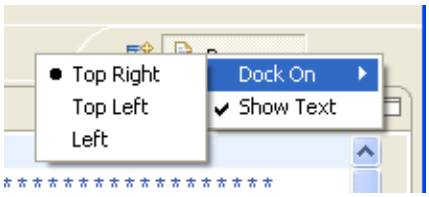
Linking view to current open editor	The resource Navigator view (and similar views) is not tightly linked to the currently open editor by default. This means that closing or switching editors does not change the selection in the Navigator view. Toggling the **Link with Editor** button in the Navigator view toolbar ties the view to always show the current file being edited. 
Copying and moving resources	You can drag and drop files and folders within the Navigator view to move them around. Hold down the Ctrl key to make copies.
Importing files	You can quickly import files and folders into your workspace by dragging them from the file system (e.g., from a Windows Explorer window) and dropping them into the Navigator view. The files and folder are always copied into the project; the originals are not affected. Copy and paste also work.
Exporting files	Dragging files and folder from the Navigator view to the file system (e.g., to a Windows Explorer window) exports the files and folders. The files and folder are always copied; workspace resources are not affected. Copy and paste also work.

Global find/replace	Use ***Search*** > ***File*** from the main menu to specify the text that you want to replace and the scope in which you want to replace it. Then press **Replace...**.
Replace from Search view	You can replace the matches in the files by using **Replace...** or **Replace Selected...** from the context menu in the Search view.
Collapsing all open items	Use the **Collapse All** button on the toolbar of the Navigator view (and similar views) to collaps all expanded project and folder items. 
Open editors with a single click	Use the Open mode setting on the **General** preference page to activate single click opening for editors. In single click mode, a single click on a file in the Navigator view (and similar views) selects and immediately opens it.
Next / previous navigation	You can use Ctrl+. and Ctrl+, to navigate to the next or previous search match, editor error, or compare difference. These are the shortcut keys for **Navigate** > **Next** and **Navigate** > **Previous**
Describing your configuration	When reporting a problem, it's often important to be able to capture details about your particula setup. The **Configuration Details** button on the **Help** > **About *Product*** dialog opens a file containing various pieces of information about your setup, including plug−in versions, preference settings, and the contents of the internal log file. You can save this, and attach the file to your problem report.
Workspace project management	Use the **Project** > **Close Project** command to manage projects within your workspace. When a project is closed, its resources are temporarily "offline" and no longer appear in the Workbench (they are still sitting in the local file system). Closed projects require less memory. Also, since they are not examined during builds, closing a project can improve build times.
Restoring a perspective's layout	Rearranging and closing the views in a perspective can sometimes render it unrecognizable and hard to work with. To return it to a familiar state, use **Window** > **Reset Perspective**.
Pinning editors	When the **Close editors automatically** preference is active (found on the **General** > **Editors** preference page), you can stop an editor from being closed by using the **Pin Editor** button which appears in the workbench toolbar. 
Importing an existing project	If you import an existing project, the resources files for the project are *not* copied. If you check the properties of the project, you'll see that the project's location in the file system is the locatio you specified.

Reordering editor tabs	You can rearrange the order of open editors by using drag and drop. Grab the editor tab and drag it to the position you want the editor to appear. When positioning editors, the stack icon indicates a valid spot to drop.
Deleting completed tasks	Use the **Delete Completed Tasks** command in the Task view context menu to remove all completed tasks from the Tasks view. This is more convenient than individually selecting and deleting completed tasks.
Quick navigation between views, editors and perspectives	A look at the **Window > Navigation** menu reveals a number of ways to quickly navigate between the various views, editors, perspectives, and menus in the workbench. These commands have keyword accelerators such as **Ctrl+F6** for switching between editors, **Ctrl+F7** for switching between views, **Ctrl+F8** for switching between perspectives, and **F12** for activating the editor.  To directly navigate to a particular view you can define a keyboard shortcut to a view via the **General > Keys** preference page.
Maximizing a view or editor	You can maximize a view or editor by double−clicking on the view's title bar or the editor's tab. Double−click again to restore it to its usual size.

Viewing resource properties	Use the Properties view (**Window > Show View > Properties**) when viewing the properties for many resources. Using this view is faster than opening the Properties dialog for each resource. 
Quickly find a resource	Use the **Navigate > Go To > Resource** command to quickly find a resource. If the **Go To > Resource** command does not appear in your perspective, you can add it by selecting **Window > Customize Perspective > Other > Resource Navigation**.
Extra resource information	Label decorations are a general mechanism for showing extra information about a resource. Use the **General > Label Decorations** preference page to select which of the available kinds of decorations you want to see.
Filtering resources	The Navigator and Tasks views both support filtering of their items. You control which items are visible by applying filters or working sets. The **Filters** commands are found on the view menu. The working set is selected using the **Select Working Set** command in the Navigator view menu. In the Tasks view, a working set can be selected from within the **Filters** dialog.
Customizing toolbar and menu bar	You can customize which items appear on the main toolbar and menu bar using the **Window > Customize Perspective** command. 
Quick fix in Tasks view	You can use the **Quick Fix** command in the Tasks view to suggest an automatic fix for the selected item. The **Quick Fix** command is only enabled when there is a suggested fix.

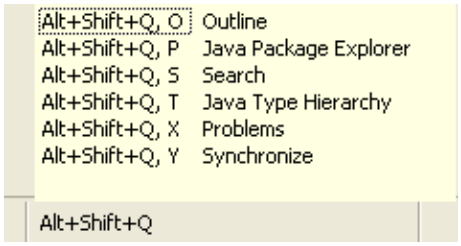
Creating path variables	When creating a linked folder or file, you can specify the target location relative to a path variable. By using path variables, you can share projects containing linked resources without requiring team members to have exactly the same path in the file system. You can define a path variable at the time you create a linked resource, or via the **General > Workspace > Linked Resources** preference page. 
Comparing zip archives with each other or with a folder	Select two zip archives or one archive and a folder in the resource Navigator view and choose ***Compare With > Each Other*** from the view's popup menu. Any differences between the two inputs are opened in a Compare editor. The top pane shows all the archive entries that differ. Double clicking on an item performs a content compare in the bottom pane. This works in any context where a file comparison is involved. So if a CVS Synchronize operation lists an archive in the resource tree, you can double click on it in order to drill down into changes within the archive.
Switch workspace 3.0	Instead of shutting down eclipse and restarting with a different workspace you can instead use the ***File > Switch Workspace***. This trick is also useful when you change certain preferences that require a restart to take effect (such as the ***General > Appearance*** Presentation preference). To restart quickly simply switch workspaces to your current workspace.

Ctrl+E Editor List 3.0	You can quickly switch editors using the Ctrl+E keybinding which opens a list of all open editors. The list supports type−ahead to find the editor as well as allows you to close editors using a popup menu or the Delete key.
View Minimizing 3.0	Running out of space? Try minimizing your unused views to reclaim screen real−estate. Each view stack contains a minimize icon along side the maximize icon. 
Detached Views 3.0	In 3.0 it's possible to dock a view in its own window, separate from the workbench window. To do so, simply drag a view outside of the workbench window. To return it to the workbench window, drag it back.
Fast Views and the Perspective Bar 3.0	The fast view and perspective bars are independent entities in 3.0 and they may be docked independent of one another. By default the Perspective Bar is located in the upper right hand corner of the screen. It may also be docked on the top left, under the main toolbar or to the far left. It may be moved via the perspective bar context menu or via the ***General*** **>** ***Appearance*** preference page.  By default the Fast View Bar is located in the bottom left hand corner of the screen. Like the Perspective Bar, it may be docked elsewhere. This may be done by dragging the area to either the left or right side of the screen (or back to the bottom if it is already in one of these positions). 

Key Binding Assistance

3.0

Eclipse supports key bindings that contain more than one key stroke. Examples of such key bindings are "Ctrl+X S" ("Save" in the Emacs key configuration) or "Alt+Shift+Q Y" ("Open Synchronize View" in the Default key configuration). It is hard to learn these keys, and it can also be hard to remember them if you don't use them very often. It is now possible to get a little pop−up showing you the possible completions for the keys you have pressed already.

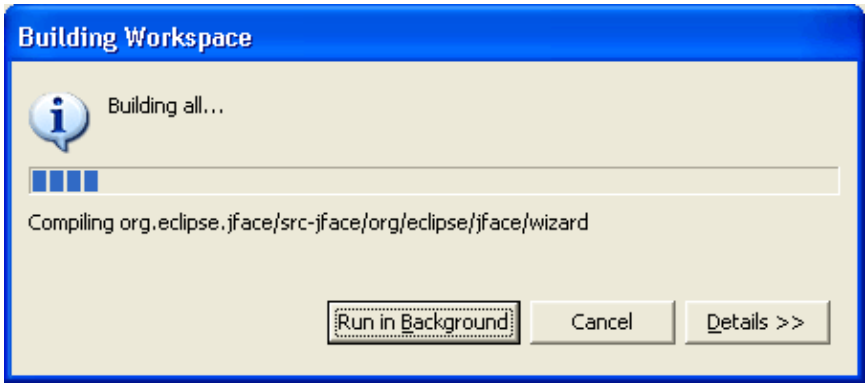


In the preferences, under ***General*** > ***Keys***, there is an "Advanced" tab. Go to this tab, and check "Help Me With Multi−Stroke Keyboard Shortcuts".

Always Run in Background

3.0

In Eclipse 3.0 many operations can be optionally run in the background so that you can continu working while they complete.

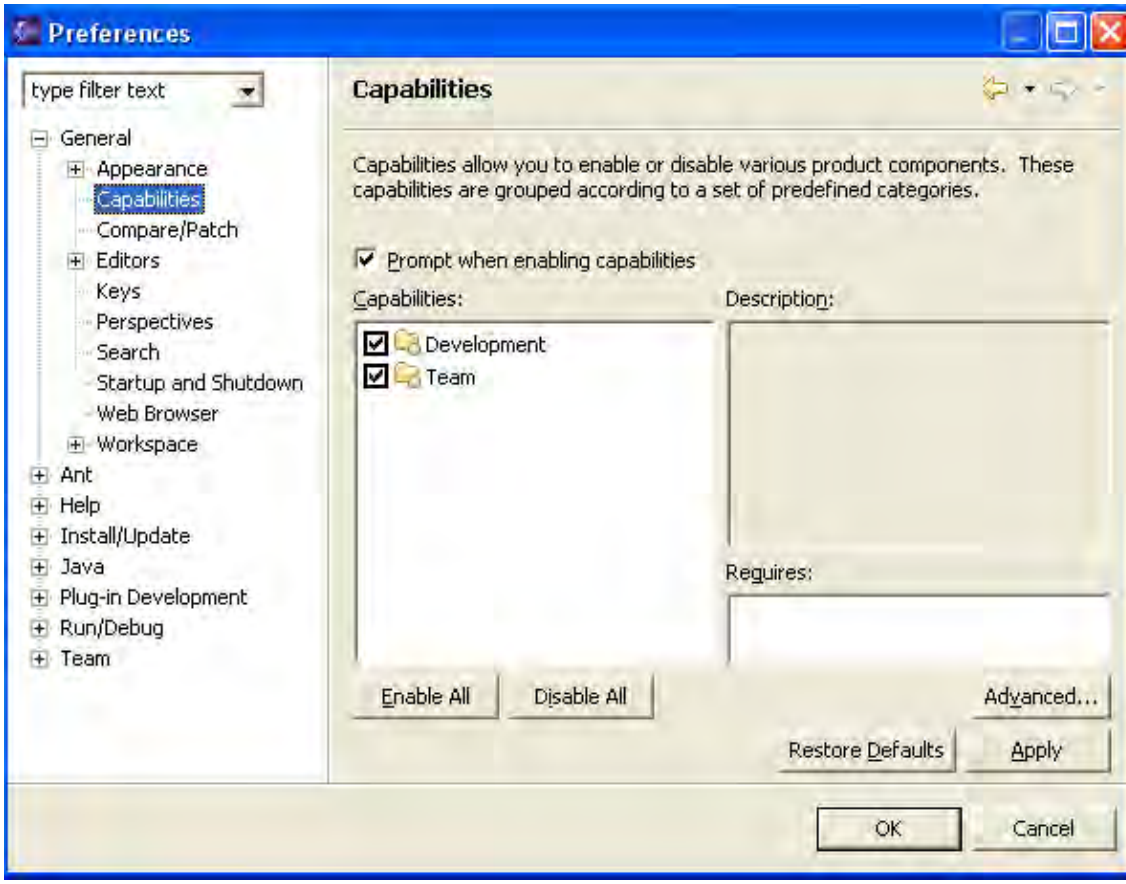


In the ***General*** preference page you can choose to always run in background so that you never get the initial dialog for these operations.

Disabling Unused Capabilities

3.0

If there are parts of the Eclipse Platform that you never use (for instance, you don't use CVS repositories or you don't develop Plug−ins) it's possible that you can disable them from the UI entirely. Segments of the Workbench that may be filtered can be found in the ***General*** > ***Capabilities*** preference page. By disabling capabilities you are able to hide views, perspectives, preference pages and other assorted contributions.



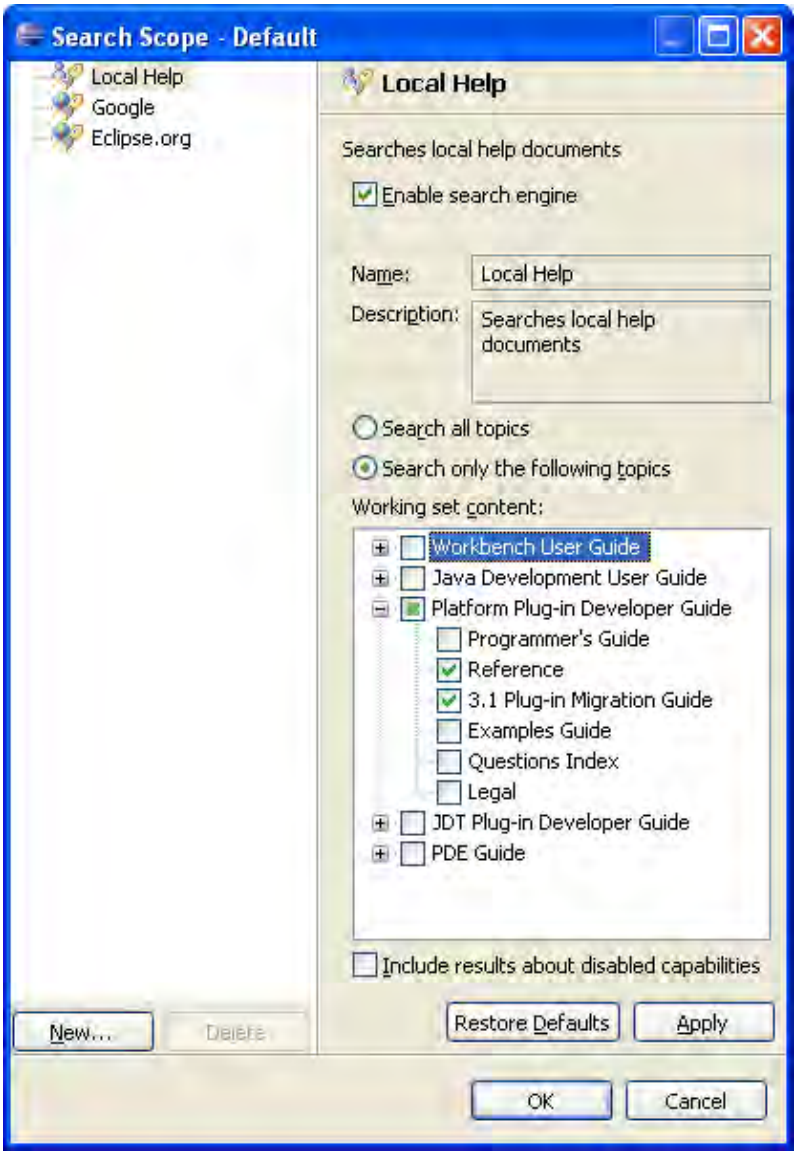
Ant

Launching from the Context menu 3.0	You can launch an Ant build from the context menu. Select an Ant buildfile and then choose **Run > Ant Build** from the context menu. To configure options before running the build, use **Run > Ant Build...**, which will open the launch configuration dialog. A build can also be started from the Ant editor outline context menu.
Specification of JRE 3.0	You can specify the JRE that an Ant build occurs in using the **JRE** tab of the launch configuration dialog for an Ant launch configuration. The build can be set to run in a separate JRE (the default setting) or the same JRE as the Eclipse workspace. Note that some Eclipse specific tasks require that the build occurs in the same JRE as Eclipse.
Running Ant targets in the Ant view	You can double click on a target in the Ant view to run it (equivalent to selecting the target and choosing the **Run** command from the context menu).
Terminating Ant builds	The **Terminate** command in the console (or Debug view) can be used to terminate an Ant build running in the background.
Ant output	The output from Ant builds is written to the **Console** view in the same

and hyperlinks	hierarchical format seen when running Ant from the command line. Ant tasks (for example "[mkdir]") are hyperlinked to the associated Ant buildfile, and javac error reports are hyperlinked to the associated Java source file and line number. The Console supports hyperlinks for javac and jikes as well as the Eclipse Java compiler. All such error reports are hyperlinked to the associated Java source file and line number.
Ant can find it	When the **Run > External Tools > Run As > Ant Build** launch shortcut is used, it searches for the buildfile to execute starting in the folder of the selected resource and working its way upwards (some will recognize this as Ant's "−find" feature). The names of buildfiles to search for are specified in the Ant preference page.

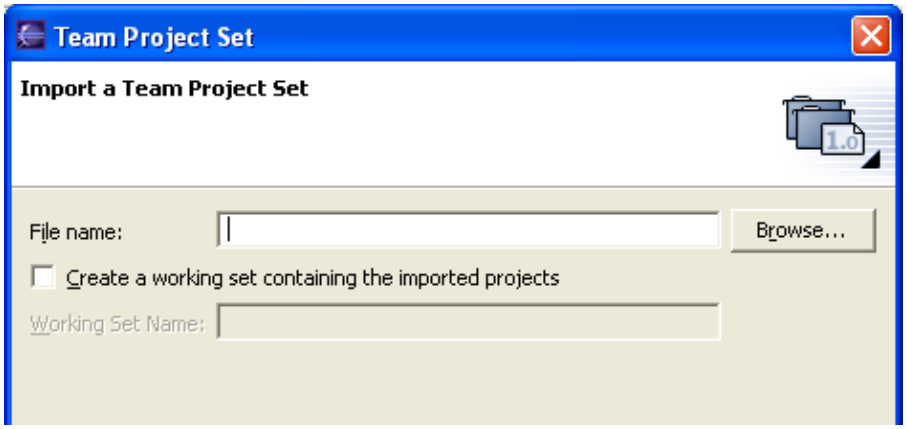
Help

Help bookmarks	You can now keep your own list of bookmarks to pages in help books. Create a bookmark with the  **Bookmark Document** button on the toolbar of the Help browser. The bookmarks show up in the  **Bookmarks** tab.
Help search scope	Help search scope allows narrowing searches down as far as a section of a book, or expanding searches to remote search engines in addition to local documentation. Working sets are persisted from one session to the next, and can be used in workbench help searches. Search scope of local documentation also applies when searching from within the Help browser.

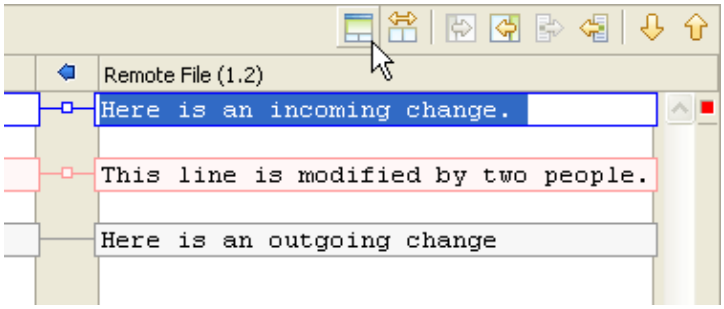
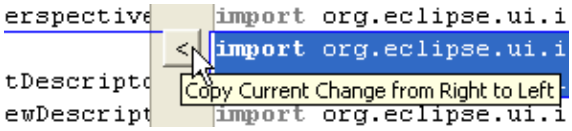
	
Context−sensitive infopops	If you are familiar with infopops used in previous releases, you can configure Help to use infopops instead of help view for context sensitive help, via a setting on the **Help** preference page.

Team − CVS

CVS Watch/Edit	The "edit" portion of CVS Watch/Edit is now supported. Through settings on the **Team > CVS > Watch/Edit** preference page (which must be set before the projects are added to your workspace), you can choose to automatically notify the CVS server whenever you start to edit a file. In turn, the CVS server will notify others on the watch list for that file. When you go to edit a file, you are warned if there are others editing the same file. **Team > Show Editors** on a file's context menu lists everyone currently working on the file. There are also **Team > Edit** and **Unedit** commands.

Working set for imported team projects	There is an option to create a working set for projects imported into the workspace via **Import** > **Team Project Set**. This works for all types of repositories. 
CVS now supports working sets	Users can now define working sets which will limit the number of projects shown in the CVS Repositories view.
Comparing different versions	Select any folder or file in the CVS Repositories view and choose **Compare With** from context menu to compare it against another version, branch, or date.
Restoring deleted files from CVS	Deleted files can now be queried and restored from the CVS repository using the **Team** > **Restore from Repository** command, which is available on CVS projects and folders.
Pin a Synchronization 3.0	You can now have multiple synchronizations defined and available in the Synchronize View. Use the pin toolbar button in the Synchronize View to pin a synchronization. The next time you synchronize a new synchronization will be created. This way you can synchronize different sets of resources.
Checkout Wizard 3.0	You can now checkout projects in one easy step via the **File** > **Import** > **Checkout projects from CVS wizard**. This also allows checking out projects from a CVS server that doesn't support browsing of its contents.
Browsing changes by CVS change set 3.0	You can browse a set of changes shown in the Synchronize View grouped logically by author, comment, and date. Enable the layout by clicking on the Change Set toolbar button. This layout can be used in the Incoming mode when synchronizing and when comparing.
Group outgoing changes	You can group outgoing changes into change sets in the Synchronize View. To enable this, switch to Outgoing mode and enable the Change Set toolbar button. You can then create outgoing change sets and assign

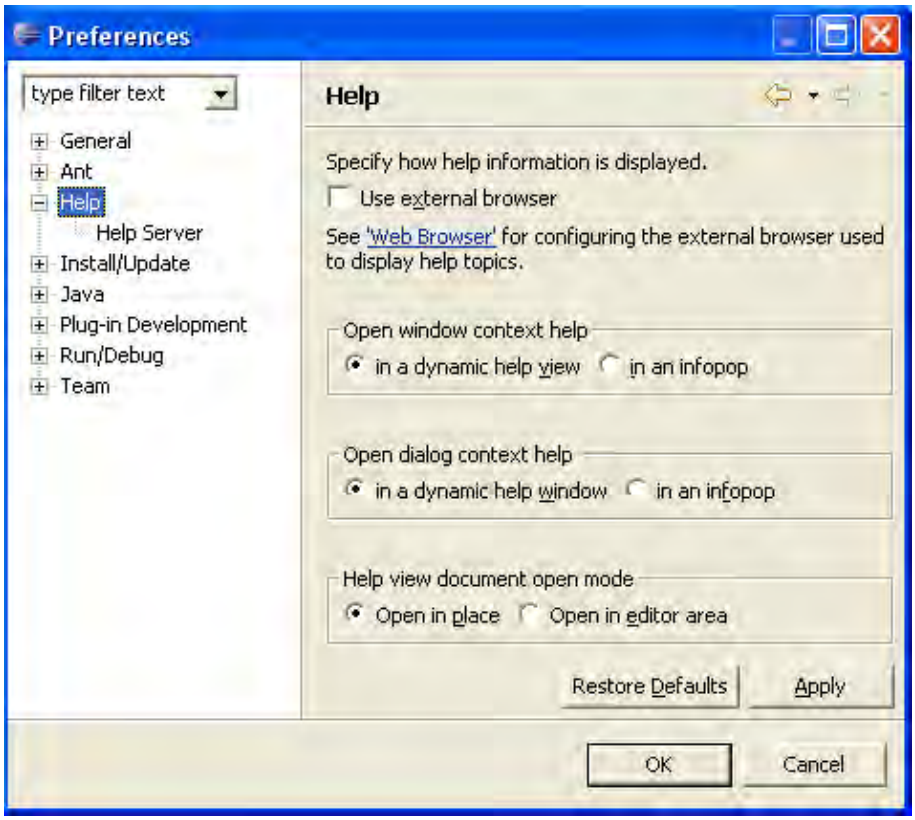
3.1	changes to them.
Schedule a synchronize 3.0	You can schedule that a certain synchronization run periodically. You can schedule any CVS synchronization from within the Synchronize View via the **Schedule...** action in the view's dropdown menu.
Want to release changes to an existing branch 3.0	If you have changes in your workspace that you would like to commit to another branch than the one currently connected to, you can run the **Team > Switch to Another Branch or Version** command and switch to another branch. This operation won't modify the changed files and you can then commit them to the other branch.
Sharing your CVS lineup with others	You can save the list of projects shared with CVS into a team project set. This provides an easy way of re−creating your workspace with shared CVS projects. 1. Once you have checked out the set of projects from the CVS repository, select ***File > Export*** from the main menu. 2. Select ***Team Project Set*** from the list and then select the projects to be exported. The generated file can be shared with your team to allow quick setups of your development environment. 3. To import the project set select ***File > Import*** and select ***Team Project Set***. The projects will be checked out of CVS and a repository location will automatically be created.
Reverting a managed CVS file that was edited, but not committed	There are a two ways of doing this: 1. Select the file and from the context menu select **Replace With > Latest from HEAD**. or 1. Select the file or a parent folder and from the context menu select ***Team > Synchronize with Repository***. 2. Next switch to incoming/outgoing mode using the toolbar button in the view. 3. Select the file and from the context menu select ***Override and Update***.
Show ancestor pane in 3−way compares	Whenever a CVS synchronization results in a conflict, it is helpful to view the common ancestor on which the two conflicting versions are based. You can view the common ancestor by toggling the **Show Ancestor Pane** button in the compare viewer's local toolbar.

	 If you always want to have the ancestor pane open automatically for conflicts, you can check the option **Initially show ancestor pane** on the Text Compare tab of the Compare/Patch preference page.
Merge in Compare editor 3.0	You can merge incoming changes in the compare editor with one click. Hover over the small square in the middle of the line connecting two ranges of an incoming or conflicting change. A button appears that allows you to accept the change.  Note that for this the option **Connect ranges with single line** on the **General > Compare/Patch > Text Compare** preference page has to be enabled.
Content assist for branching and merging 3.1	When branching and merging with CVS, you can use content assist in the tag fields to help select an appropriate tag. For instance, when branching, you can use content assist to pick a tag from the list of branch tags that exist on the other projects in your workspace. When merging, you can use content assist to pick the branch that contains the changes you are merging. The merge wizard will also try to pick the proper start tag for you so you do not have to pick it manually.
Filtering in tag selection dialogs 3.1	There are several CVS operations that allow the user to specify a tag (e.g. Replace With Branch or Version, Compare With Branch or Version, Checkout, etc.). These dialogs now allow you to type in part of the tag name (or simple name filters using the * and ? wildcard characters) and display all the tags that match what you have typed so far. This greatly simplifies finding the desired tag when performing these operations.

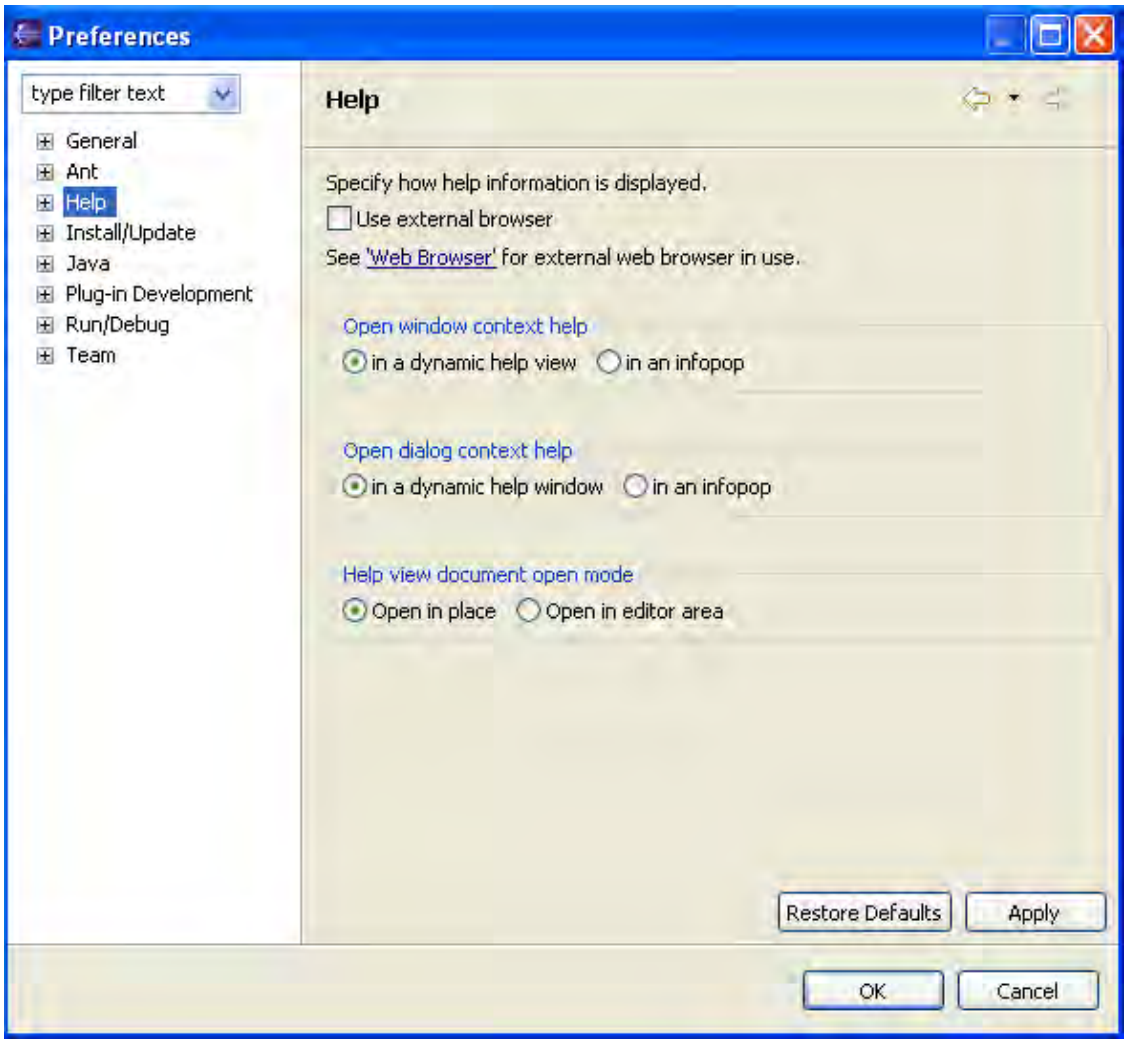
Documentation Images

Many of the images found in the Eclipse documentation have been taken using the Windows XP Manifest, giving them the default Windows XP look. The Windows XP Manifest affects the look and feel of the Eclipse platform, rounding out otherwise blunt edges of boxes and buttons, as can be seen in the following examples.

Screenshot of the Help Preferences page without the Windows XP Manifest.



Screenshot of Help Preferences page with the Windows XP Manifest.



If you would like to use the Windows XP Manifest on your installation simply place the javaw.exe.manifest file in the same directory as your java.exe executable. The javaw.exe.manifest file can be found in org.eclipse.swt.win32.win32.x86_3.1.0.jar.

Notices

About This Content

February 24, 2005

License

EPL, "Program" will mean the Content.

If you did not receive this Content directly from the Eclipse Foundation, the Content is being redistributed by another party ("Redistributor") and different terms and conditions may apply to your use of any object code in the Content. Check the Redistributor's license that was provided with the Content. If no such license exists, contact the Redistributor. Unless otherwise indicated below, the terms and conditions of the EPL still apply to any source code in the Content.