

**Chapter  
2**

**ECLIPSE**

*Get on the  
Fast Track!*

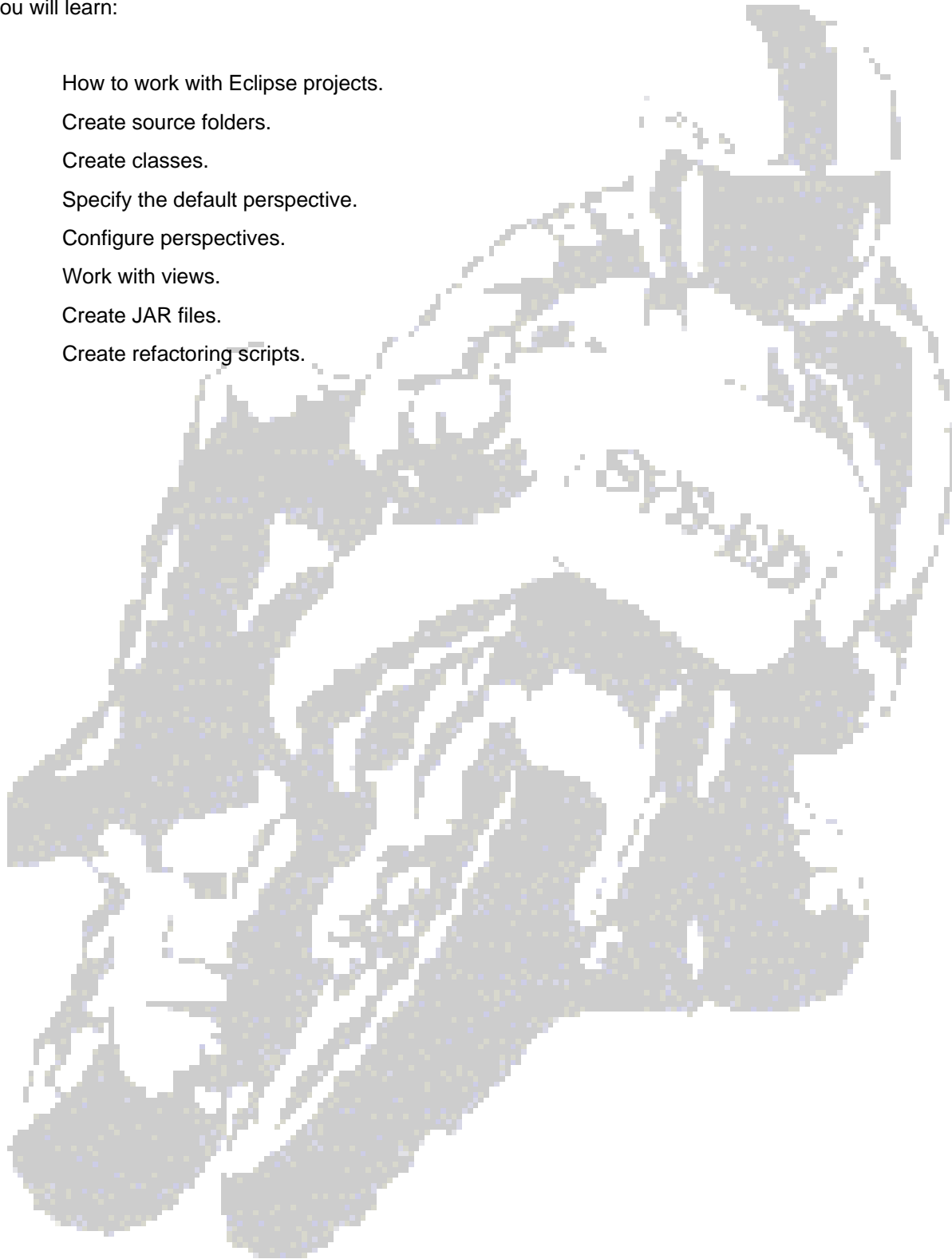


**SYS-ED/  
Computer  
Education  
Techniques, Inc.**

## Objectives

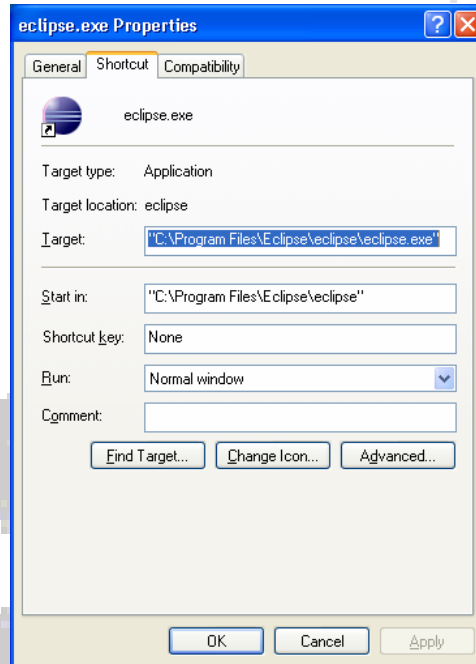
You will learn:

- How to work with Eclipse projects.
- Create source folders.
- Create classes.
- Specify the default perspective.
- Configure perspectives.
- Work with views.
- Create JAR files.
- Create refactoring scripts.

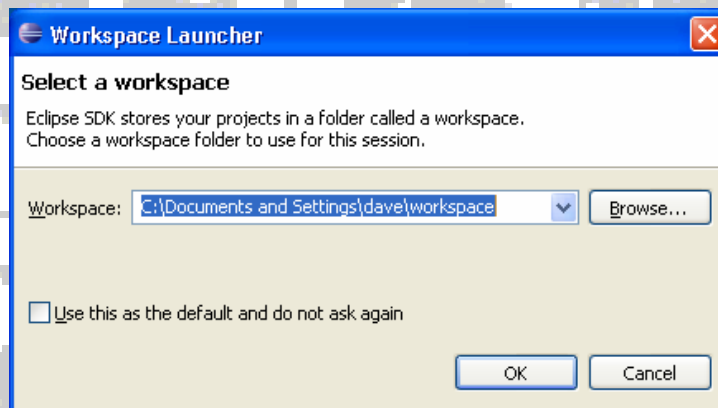


## 1 Starting Eclipse

In order to start Eclipse, the eclipse.exe command will need to be executed in the Eclipse folder. Alternatively, a short cut can be created.



1. Start Eclipse and supply a path to a new folder which will be serving as the workspace. The workspace is a folder which Eclipse uses to store source code.



- 2. Avoid checking the "Use this as the default..." box, which will always start up Eclipse with the same workspace. It will be useful to have the capability for switching between different workspaces on startup.

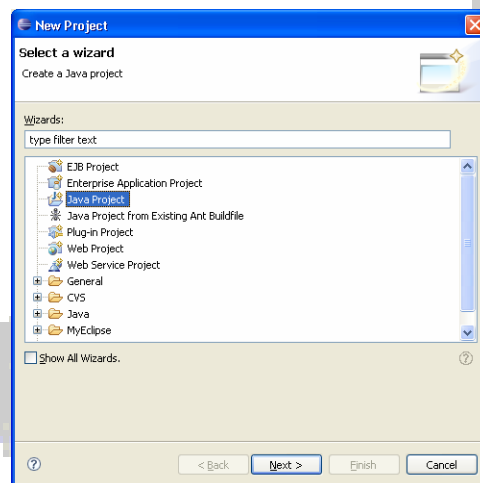
The following Welcome screen will be visible:



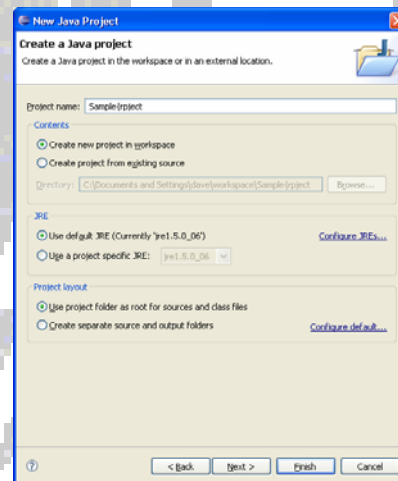
## 2 Project - Creation

Close the Welcome page.

1. Right-click in the Navigator panel, and select New...Project. Alternatively, Select File...New...Project.



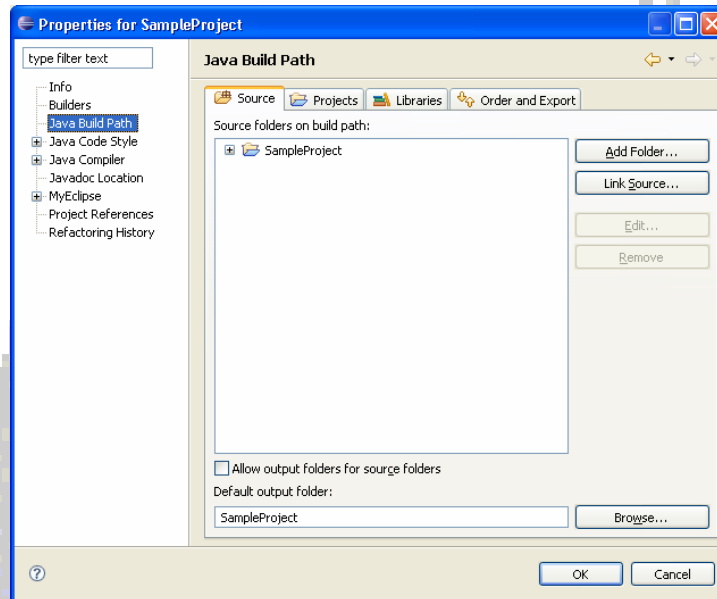
2. Select "Java Project" and then "Next"; enter the name of your project.  
Projects are directories stored under the directory that have been selected as the workspace.



3. Click "Finish".
4. When prompted about switching to the Java perspective, select "Yes".

### 3 Source Folders - Creating

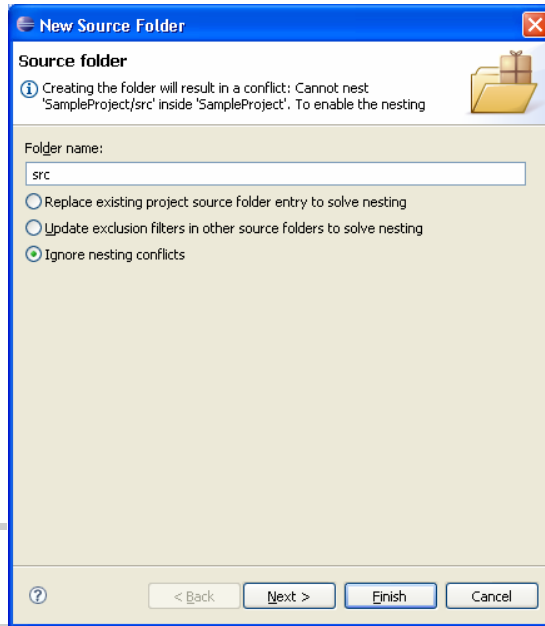
1. Select Project...Properties.
2. Select Java Build Path and click on the Source tab.



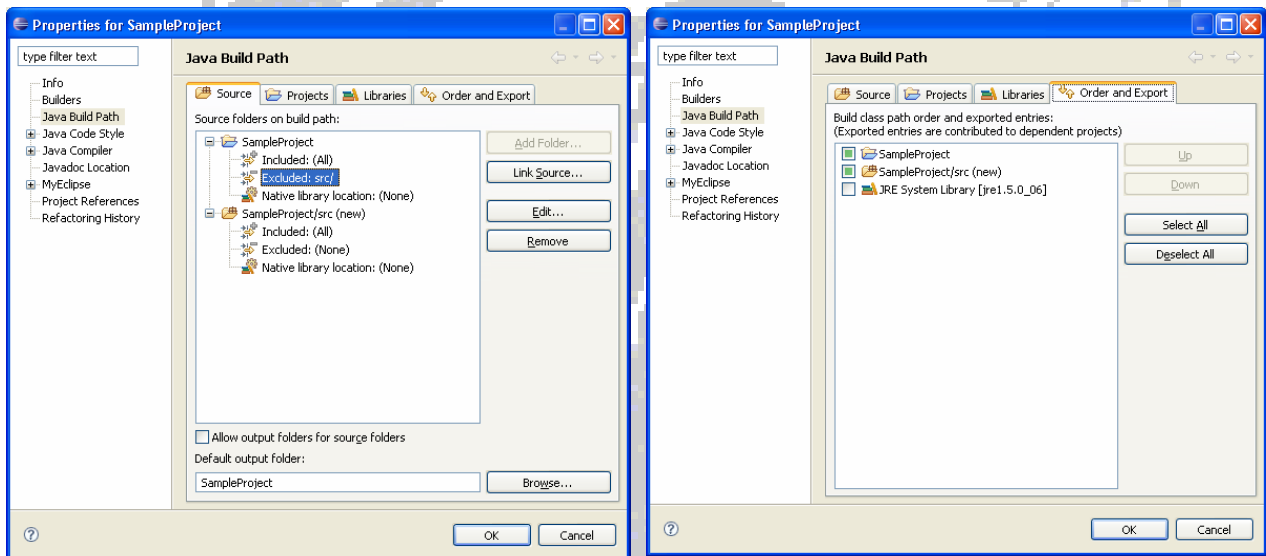
Eclipse projects can contain one or more source folders which contain Java source files. By default, the project directory itself is the only source folder.

Using multiple source folders is a good way to separate different types of code. This can make a project easier to work with and simplifies building Ant scripts. Separate source folders are going to be set up for application code and unit test code.

3. Click on "Add Folder" and type "src":



4. Click on Finish.  
There will now be two source folders



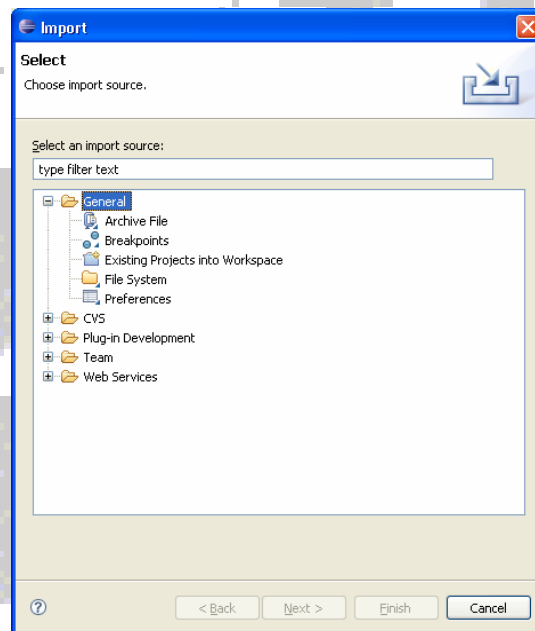
Clicking on "Order and Export" displays the complete classpath used for compiling any Java source files in the project.

## 4 Source Code - Importing

In order to import code:

1. Right click on the project selecting "Import".
2. Select General and Archive for jar/zip files and select File Systems for other files.

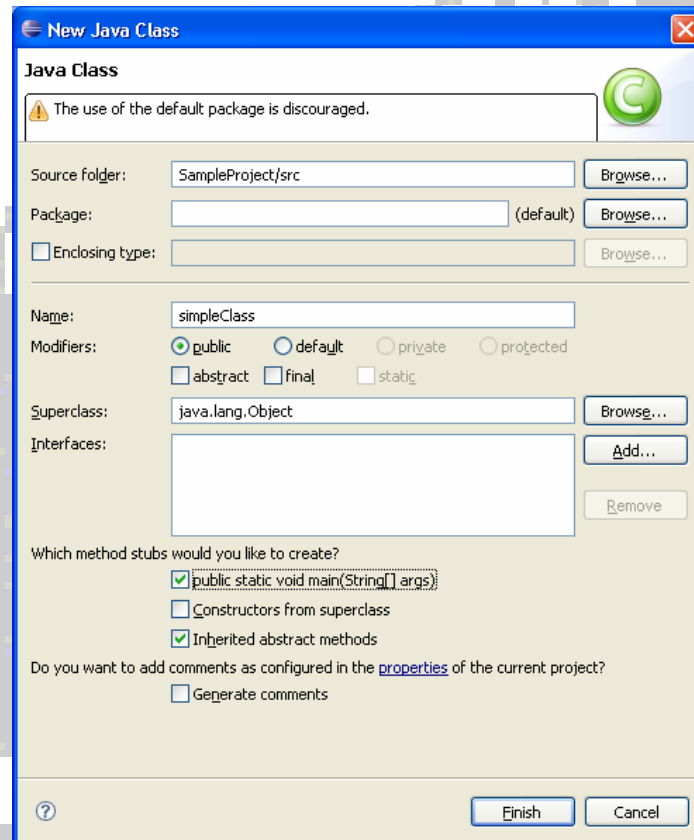
The rest of the process is self explanatory.



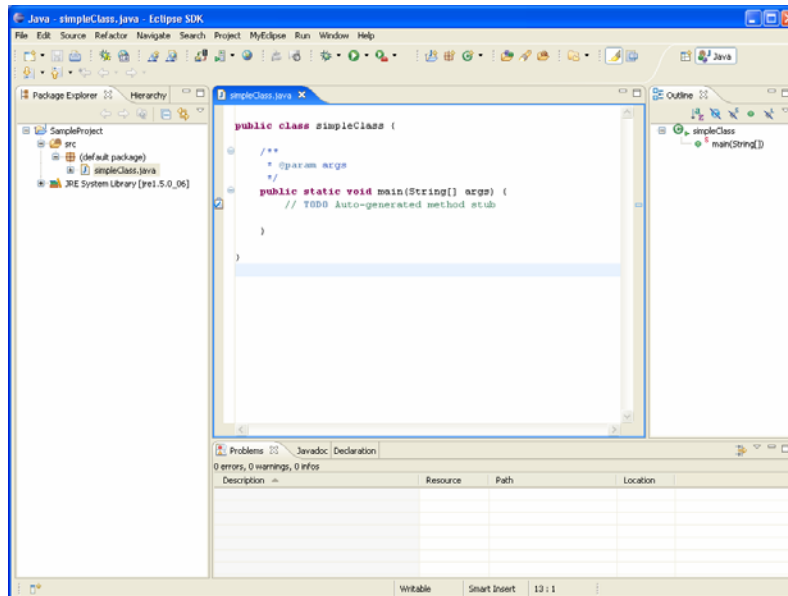
## 5 Class - Creation

In order to create new Java code:

1. Right-click on the "src" source folder and select "New...Class".
2. Supply a class name and select public main.



The program will now be available:



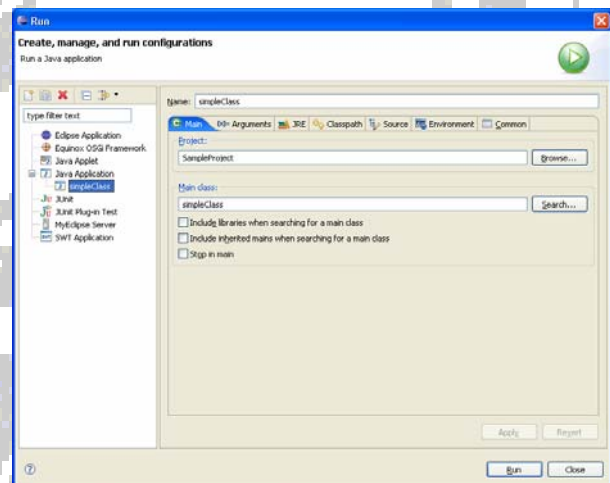
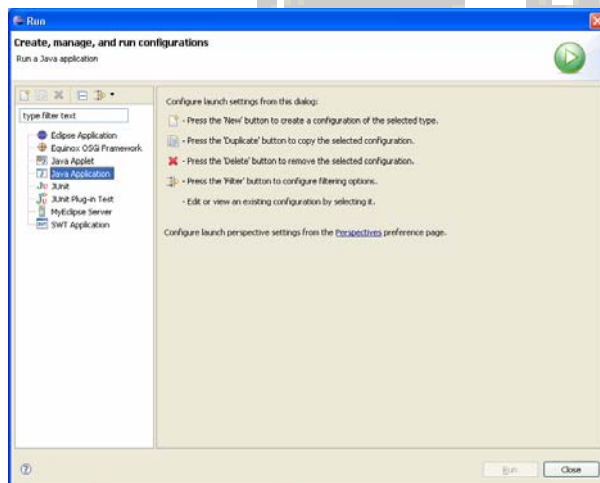
## 6 Execute the Program

Enter some code into the program.

### Example:

```
System.out.print("Hello World");
```

1. Click on the save button.
2. Select the menu Run...Run...
3. Double click on Java Application and select the class to run.



4. Click on Run button and the output will appear in the output window.

---

## 7 Default Perspective - Specifying

The default perspective is indicated in the Select Perspective dialog which is accessible via the Window...Open Perspective...Other... menu. The pre-defined default perspective is indicated by the word default in brackets following the perspective name.

In order to change the default perspective:

1. Open the General...Perspectives preference page.
2. Select the perspective that is to be defined as the default from the list of available perspectives, and click Make Default.  
The default indicator moves to the perspective that has been selected.
3. Click OK.

---

## 8 Perspectives - Configuring

In addition to configuring the layout of a perspective, it is also possible to control several other key aspects of a perspective.

- 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.







In order to configure a perspective:

1. Switch to the perspective that is to be configured.
2. Select Window...Customize Perspective....
3. Expand the item that is to be customized.
4. Use the checkboxes to select which elements are to be seen on drop-down menus in the selected perspective. Items which have not been selected will be accessible by clicking the Other menu option.
5. Click OK.

## 9 Views - Moving and Docking

In order to change the location of a view in the current perspective:

6. Drag the view by its title bar. Do not release the left mouse button yet.
7. While moving the view around the Workbench, the mouse pointer changes to a drop cursor.  
The drop cursor indicates where the view will be docked if you release the left mouse button. IN order to see the drop cursor change, drag the view over the left, right, top, or bottom border of another view or editor.  
  
The view can also be dragged outside of the Workbench area to turn it into a "Detached" view.
8. When the view is in the specified location, relative to the view or editor area underneath the drop cursor, release the left mouse button.
9. Optionally, in order to save the changes, select Window...Save Perspective As... from the main menu 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	The view cannot be docked in this area.

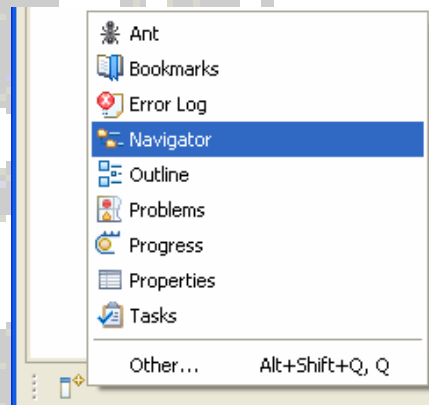
## 10 Fast Views - Creation

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 the Workbench window.

In order to create a fast view:

1. Click the title bar of the specified view. Hold the mouse button down.
2. Drag the view to the Fast View bar and release the mouse button. By default the shortcut bar is located in the lower left corner of the window.

Alternatively, the button located on the left side of the Fast View bar can be clicked which will present a selection of views. Choosing one of these views will add it to the Fast View bar immediately.



---

## 11 Bookmark - Creating within a File

The Workbench provides the capability for creating bookmarks in files that are being edited in order that these files can be quickly reopened from the Bookmarks view (Window...Show View...Other.....General...Bookmarks).

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 is to be bookmarked.
2. Select Add Bookmark from the pop-up menu.

An icon for the bookmark which now appears in the left border of the editor area. A line is also added to the Bookmarks view.

The file can be reopened for editing at any time by double-clicking the bookmark in the Bookmarks view.

---

## 12 Associating a Task with a Resource

Tasks can be associated with an editable resource.

1. In one of the navigation views, double-click the resource with which new tasks is to be associated. 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 the new task is to be logged.
3. On the pop-up menu, select Add Task.
4. When prompted, enter a brief description of the task.

---

## 13 Local History

A local edit history of a file is maintained when a file is being created or edited. Each time a file is edited and saved, a copy is saved which will replace the current file with a previous edit or even restore a deleted file.

It is also possible to 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.

In order to compare an unmanaged Workbench resource with a state in the local history:

1. In one of the navigation views, select the resource that is to be compared 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 finished.

In order to compare a Workbench resource that is managed by CVS with a state in the local history:

1. In one of the navigation views, select the resource that is to be compared with a local history state.
2. From the resource's pop-up menu, select Compare With...History.... The History View opens.
3. Put the History View into Local Revisions mode by clicking on the Local Revisions toolbar button in the History View. This filters out all revisions except for Local History revisions.
4. Select a revision from the History View. The Text Compare editor opens.
5. 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.
6. Click OK when finished.

---

## 14 New JAR File - Creation

In order to create a new JAR file in the workbench:

1. In the Package Explorer, it is possible to optionally pre-select one or more Java elements to export.

Either from the context menu or from the menu bar's File menu, select Export.

2. Select JAR file, then click Next.
3. In the JAR Package Specification page, select the resources that are to be exported in the Select the resources to export field.
4. Select the appropriate checkbox to specify whether there is need to Export generated class files and resources or Export Java source files and resources.  
Selected resources will be exported in both cases.

5. In the Select the export destination field, either type or click Browse to select a location for the JAR file.

6. Select or clear the Compress the contents of the JAR file checkbox.

7. Select or clear the Overwrite existing files without warning checkbox.

If this checkbox is cleared, then there will be a prompt to confirm the replacement of each file that will be overwritten.

8. The overwrite option is applied when writing the JAR file, the JAR description, and the manifest file.

9. There will be two options:

- Click Finish to create the JAR file immediately.
- Click Next to use the JAR Packaging Options page to set advanced options, create a JAR description, or change the default manifest.

---

## 15 Refactoring Scripts - Creation

In order to create refactoring scripts:

1. Select Refactor...Create Script... from the refactoring menu to show the create refactoring script dialog.
2. The create refactoring script dialog displays the entire refactoring history of your workspace. Drill down to an individual refactoring and select it to see more details about the performed refactoring in the details view below.
3. Refactoring scripts can either be copied to the clipboard or saved to a file in your local file system. Select Clipboard to copy the created refactoring script to the clipboard, or File to store the script to a file. Click on Browse... to select the location where to store the refactoring script.
4. Select all refactorings that will be saved into a refactoring script. Select years, months, weeks or days to select all associated refactorings at once. Select a project to save all refactorings associated with the project to a refactoring script.
5. In order to be able to better select refactorings from the refactoring history, there are two ways to group refactorings:

Group by Project	Refactorings are grouped by project. Every refactoring is associated with the project that contains the element which served as input to the recorded refactoring.  There are certain instances of refactorings which cannot be uniquely associated with a project. These refactorings are associated with the entire workspace.
Group by Date	Refactorings are grouped by date. The refactoring history dialog presents the refactorings as a history, last recently executed refactorings first.

6. Click Create to create the refactoring script and close the dialog

---

## 16 Refactoring Scripts - Applying

In order to apply Refactoring Scripts:

1. Select Refactor...Apply Script... from the refactoring menu to show the apply refactoring script wizard.
2. On the first page of the Apply Script wizard, specify the refactoring script that should be applied to the workspace.  
Click on Browse... to browse for the location where the refactoring script is stored.
3. Once a valid refactoring script has been entered, click Next to continue.
4. On the second page of the Apply Script wizard, the wizard displays a preview of the refactorings to be replayed.  
Drill down to an individual refactoring and select it to see more details about the performed refactoring in the details view below.
5. Click OK to replay the refactorings and close the dialog.
6. Click Next to replay the refactorings and preview each refactoring before applying it to the workspace.

---

## 17 Breakpoints

A breakpoint suspends the execution of a program at the location where the breakpoint is set.

Breakpoints can be enabled and disabled via the context menu in the Breakpoints View, or via the context menu in the Java Editor ruler.

- An enabled breakpoint causes a thread to suspend whenever the breakpoint is encountered. Enabled breakpoints are drawn with a blue circle and have a checkmark overlay once successfully installed.
- A breakpoint can only be installed when the class the breakpoint is located in has been loaded by the VM.
- A disabled breakpoint will not cause threads to suspend. Disabled breakpoints are drawn with a white circle.

Line breakpoints are set on an executable line of a program.

1. In the editor area, open the file where you want to add the breakpoint.
2. Directly to the left of the line where the breakpoint is to be added, open the marker bar (vertical ruler) pop-up menu and select Toggle Breakpoint. An alternative way of achieving this will be by double-clicking on the marker bar next to the source code line.  
A new breakpoint marker appears on the marker bar, directly to the left of the line where the breakpoint is added.

Method breakpoints are used when working with types that have no source code (binary types).

1. Open the class in the Outline View, and select the method where a method breakpoint is to be added.
2. From the method's pop-up menu, select Toggle Method Breakpoint.
3. A breakpoint appears in the Breakpoints View. If source exists for the class, then a breakpoint also appears in the marker bar in the file's editor for the method that was selected.
4. While the breakpoint is enabled, thread execution suspends when the method is entered, before any line in the method is executed.

A hit count can be applied to line breakpoints, exception breakpoints, watchpoints and method breakpoints. When a hit count is applied to a breakpoint, the breakpoint suspends execution of a thread the nth time it is hit, but never again, until it is re-enabled or the hit count is changed or disabled.

In order to set a hit count on a breakpoint:

1. Select the breakpoint to which a hit count is to be added.
2. From the breakpoint's pop-up menu, select Hit Count.
3. In the Enter the new hit count for the breakpoint field, type the number of times that the breakpoint is to be before suspending execution.

---

## 18 Java Scrapbook Page - Creation

The scrapbook allows Java expressions, to be run, inspected, and displayed under the control of the debugger. Breakpoints and exceptions behave as they do in a regular debug session.

Code is edited on a scrapbook page. A VM is launched for each scrapbook page in which expressions are being evaluated. The first time an expression is evaluated in a scrapbook page after it is opened, a VM is launched. The VM for a page will remain active until the page is closed, terminated explicitly in the debugger or via the Stop the Evaluation button in the editor toolbar, or when a `System.exit()` is evaluated.

There is more than one way to open the New Java Scrapbook Page wizard.

- Create a file with a `.jpage` extension
- From the menu bar, select `File...New...Other....` Then select `Java...Java Run/Debug...Scrapbook Page`. Then click `Next`.

Once the New Java Scrapbook Page wizard has been opened:

1. In the Enter or select the folder field, type or click `Browse` to select the container for the new page.
2. In the File name field, type a name for the new page.  
The `.jpage` extension will be added automatically if it has not been typed in.
3. Click `Finish` when complete.  
The new scrapbook page opens in an editor.

Inspecting shows the result of evaluating an expression in the Expressions view.

4. In the scrapbook page, either type an expression or highlight an existing expression to be inspected.
5. Click the `Inspect` button in the toolbar or select `Inspect` from the selection's pop-up menu.
6. The result of the inspection appears in a pop-up.
7. The result can be inspected like a variable in the debugger.

Displaying shows the result of evaluating an expression in the scrapbook editor.

1. In the scrapbook page, either type an expression or highlight an existing expression to be displayed.
2. Click the Display button in the toolbar or select Display from the selection's pop-up menu.
3. The result of the evaluation appears highlighted in the scrapbook editor.