

图 3.16

按键连结的动态本质(The Dynamic Nature of Key Bindings)

按键连结是由外挂程序提供，而在 Eclipse 中，可以新增或移除外挂程序。这样就能新增或移除由这些外挂程序所宣告的按键连结。Eclipse 在储存自订按键连结时，可以自动补偿这个问题。比方说，在上面的范例中，在 *Emacs* 配置中，Ctrl+Alt+W 是指派给剪下。假设使用者安装一个新的外挂程序，将 Ctrl+Alt+W 指定至特定指令。Eclipse 会将使用者的指派保留给剪下，但是会显示有小型「变更」图型的按键连结，而不会显示含有「加号」图型的按键连结。

冲突解决(Conflict Resolution)

只有少数简单、常用的按键作用可以指派给多个指令。许多配置、环境定义、平台和语言环境的所有分割键顺序在指派到网域中时，并没有彼此冲突。如果环境定义不存在，请考虑上述 Ctrl+B 的情况。有一个外挂程序将 Ctrl+B 指派给建置，则其它的外挂程序会将

Ctrl+B 指派给将文字变为粗体字。那么 Eclipse 将如何正确地解决这个冲突呢？

虽然可藉由上述的机制来大量减少冲突，但冲突仍然可能发生。两个相对独立的外挂程序可以将相同按键顺序指派给含相同环境定义、配置、平台和语言环境的不同指令。请考虑如果外挂程序于在窗口中环境定义中指派了 Ctrl+F4，且将预设配置指派给它的其中一个指令。这会与将 Ctrl+F4 指派给相同环境定义和配置中之关闭指令的 Eclipse 造成直接冲突。

这就是冲突。同时呼叫两个指令是不正确的，也不能只选择其中一个指令来接收按键作用。唯一能做的，就是忽略这两个按键连结，使 Ctrl+F4 在这个环境定义和配置中实际上没有用。

「按键」喜好设定页面显示这个本质的冲突。请注意红色的文字和 "[冲突]" 一字：

如果要解决这类冲突，使用者可以将按键顺序明确指派给其中一个指令。

另一类的冲突可能是因为按键顺序有多重按键作用。例如，在 Emacs 配置中，有许多多重按键作用的按键顺序是以 Ctrl+X 的按键作用作为开头。Ctrl+H K 是指派给关闭。Ctrl+X H 是指派给全选。

如同之前的说明，Emacs 配置会从标准配置借用按键连结。在标准配置中，Ctrl+X 是指派给剪下。虽然 Emacs 配置没有明确重新定义 Ctrl+X，但是它的许多按键连结都需要按下 Ctrl+X。在 Emacs 配置中，按下 Ctrl+X 时，就等于要进入其中一个可能已经指定的按键顺序。但我们并不希望在这时候呼叫剪下动作。

对于这类冲突，其规则是忽略已指派给剪下的 Ctrl+X。否则，就无法完成 Emacs 配置中的许多按键连结。

3.1.8 标签装饰(Label Decorations)

「卷标装饰」可让其余信息显示在项目的卷标和图标中。

「标签装饰」喜好设定页面提供每一个装饰的说明，并可以选择要让哪些装饰看得到。

「标签装饰」喜好设定页面看起来如下：

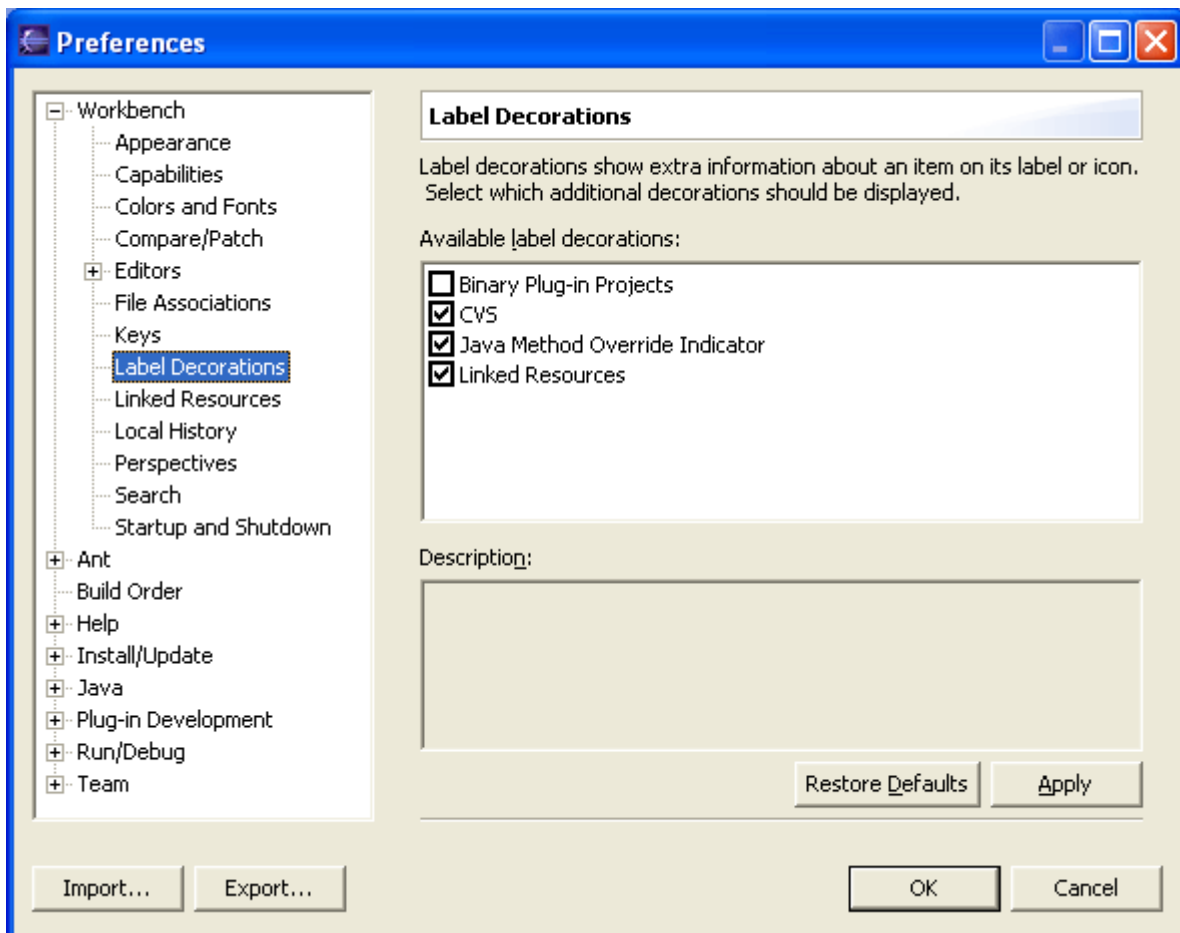


图 3.17

3.1.9 链接资源(Linked Resources)

在使用链接资源时，会使用这个喜好设定页面。Enable Linked Resources 喜好设定是用来整体启用或停用整个工作区的链接资源特性。依预设，链接资源是启用的。如果停用链接资源，就无法建立任

何新的链接资源或汇入含有链接资源的现有项目。

并非所有的工作台版本都支持链接资源并且可将它们识别为链接信息。如果打算与其它使用者共享工作区数据，可能不要使用链接资源。如果其它使用者无法使用链接资源，请停用这个喜好设定。

这个页面的其它部分是用来定义在建立链接资源时所使用的路径变量。请使用 New 按钮来定义新的变量，也可以使用 Edit 按钮来变更现有变量的值，或者使用 Remove 按钮来移除现有的变量。请注意，如果变更的路径变量正在使用中，就需要对这些项目执行本端重新整理，以“探索”档案系统中是否有任何不同之处。可以开启该项资源的「导览器」快速菜单，然后选取 Refresh，来重新整理资源。建议不要移除目前正在使用的路径变量。

「链接资源」喜好设定页面的外观如下：

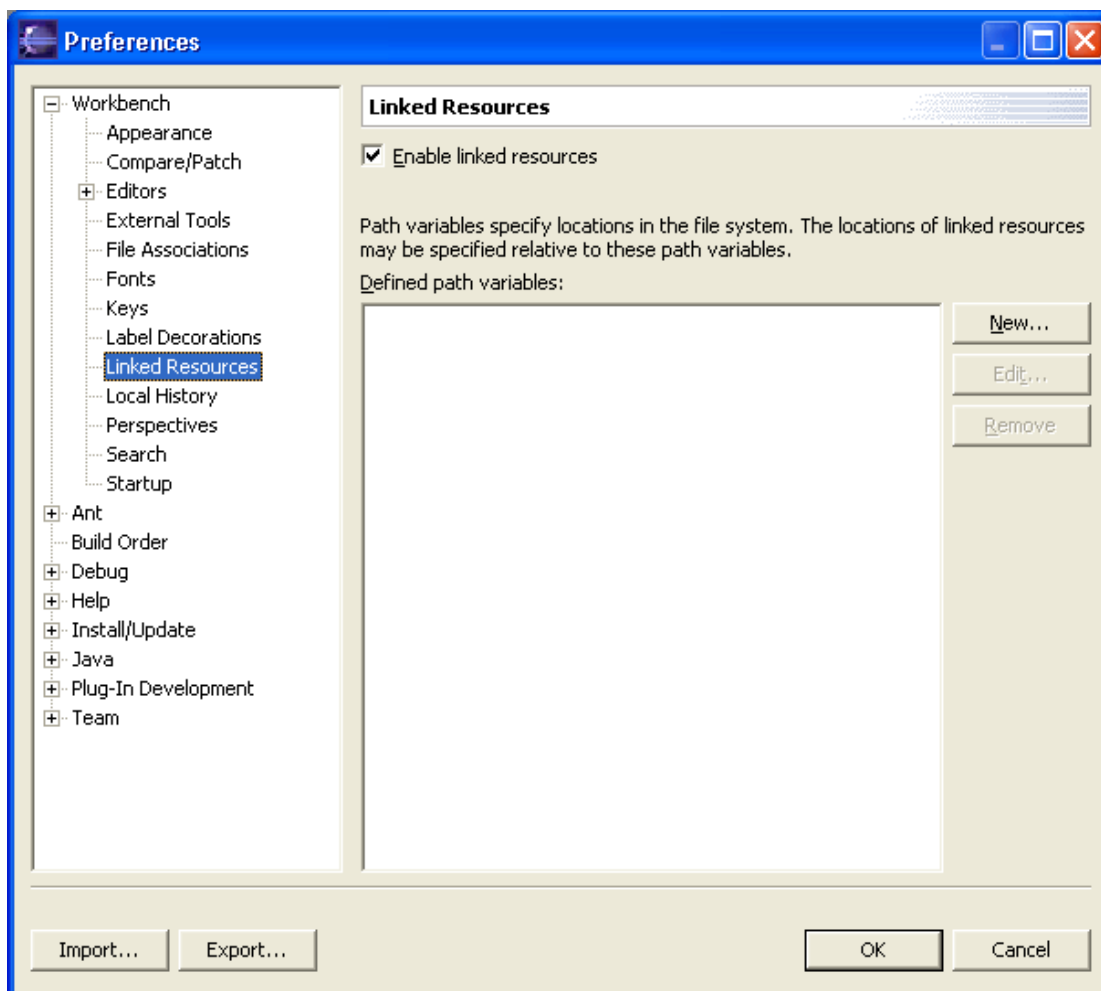


图 3.18

3.1.10 历史纪录(Local History)

可以在「历史纪录」页面中变更下列喜好设定。

选项	说明	默认值
Days To Keep Files(档案的天数)	指出要在历史纪录中维护变更多少天。这个值以外的历程状态将会流失。	7 天
Entries Per File(档案的项目数)	指出在历史纪录中每个档案要维护多少历程状态。如果超过这个值，将会失去较旧的历程，以挪出空间供新历程使用。	50 个项目
Maximum File Size(MB)(最大档案大小(MB))	指出历程储存库中的个别状态的大小上限。如果档案超过这个大小，它将不会被储存。	1 MB

「历史纪录」喜好设定页面看起来如下：

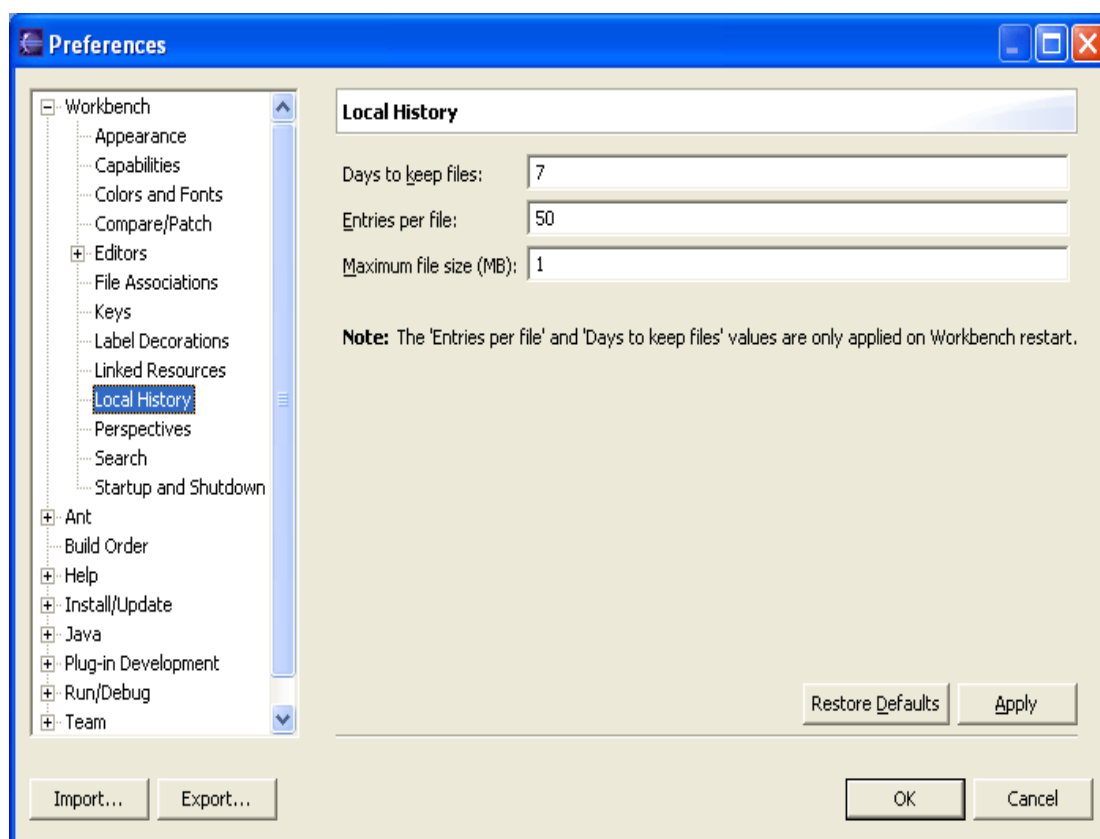


图 3.19

3.1.11 视景

在「视景」喜好设定页面上，可以管理各种定义于工作台的视景。

选项	说明	默认值
Open a new perspective(开启新视景)	请使用这个选项来设定开启新视景时会发生的情况。要在现行工作台窗口内或是新窗口内开启视景？	在相同窗口中
Open a new view(开启新视图)	请使用这个选项来指定新的视图开启时会发生什么情况。它会在其位于现行视景内的预设位置上开启，或是开启为快速视图并定置到现行视景的侧边。	在视景内
New project options(新项目选项)	请使用这个选项来指定建立新项目时的视景行为。可以将它设定为：将现行视景切换至与项目类型相关联的视景，并在同一个工作台窗口中开	在同一窗口开启视景

选项	说明	默认值
	启视景以作为现行视景、切换视景并在新的工作台窗口中加以开启，或是完全不切换视景。	

可用的视景选项：

选项	说明	默认值
Make Default (设为默认值)	将所选取的视景设为预设视景。	资源
Reset (重设)	将选取视景的定义重设成预设配置。这个选项仅适用于已经使用「窗口」「另存新视景...」来改写的内建视景。	n/a
Delete (删除)	删除所选取的视景。这个选项仅适用于使用者定义的视景（无法删除内建视景）。	n/a

「视景」喜好设定页面看起来如下：

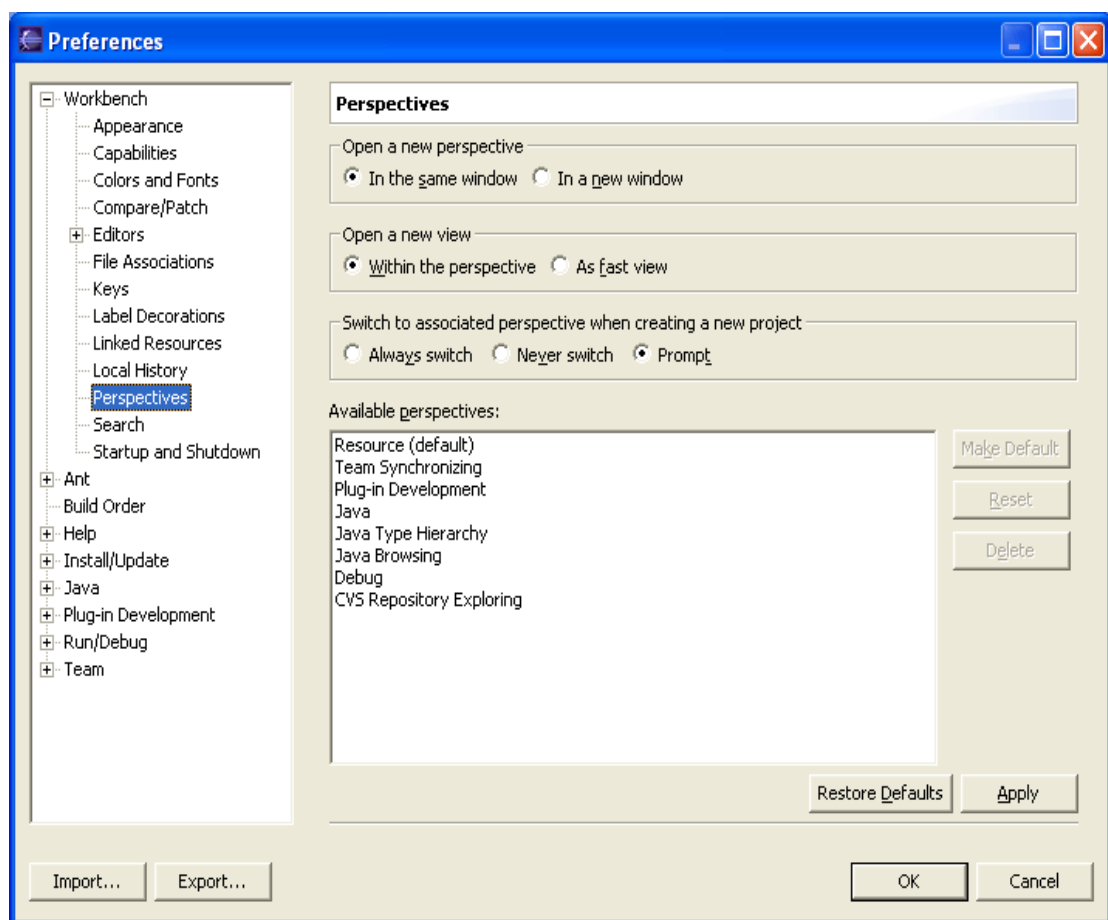


图 3.20

3.1.12 搜寻(Search)

「搜寻」喜好设定页面可让使用者设定搜寻的喜好设定。Reuse editors to show matches(重复使用编辑器来显示相符项目)选项可让使用者继续使用同一个编辑器来搜寻结果,以减少开启的编辑器数目。Emphasize inexact matches(强调不完全相符)的项目是一个选项,可以在「搜寻」视图中强调显示这些项目。如果搜寻引擎不完全确定相符项目,则该相符项目会被视为不精确。也可以设定 Foreground color for inexact matches (不完全相符项目的前景颜色)。如果只要察看完全相符的项目,请勾选 Ignore inexact matches (忽略不完全相符的项目)。Default perspective for the Search view(视图的预设视景)可以定义有新搜寻结果时要把哪一个视景移至最前面。

「搜寻」喜好设定页面看起来如下：

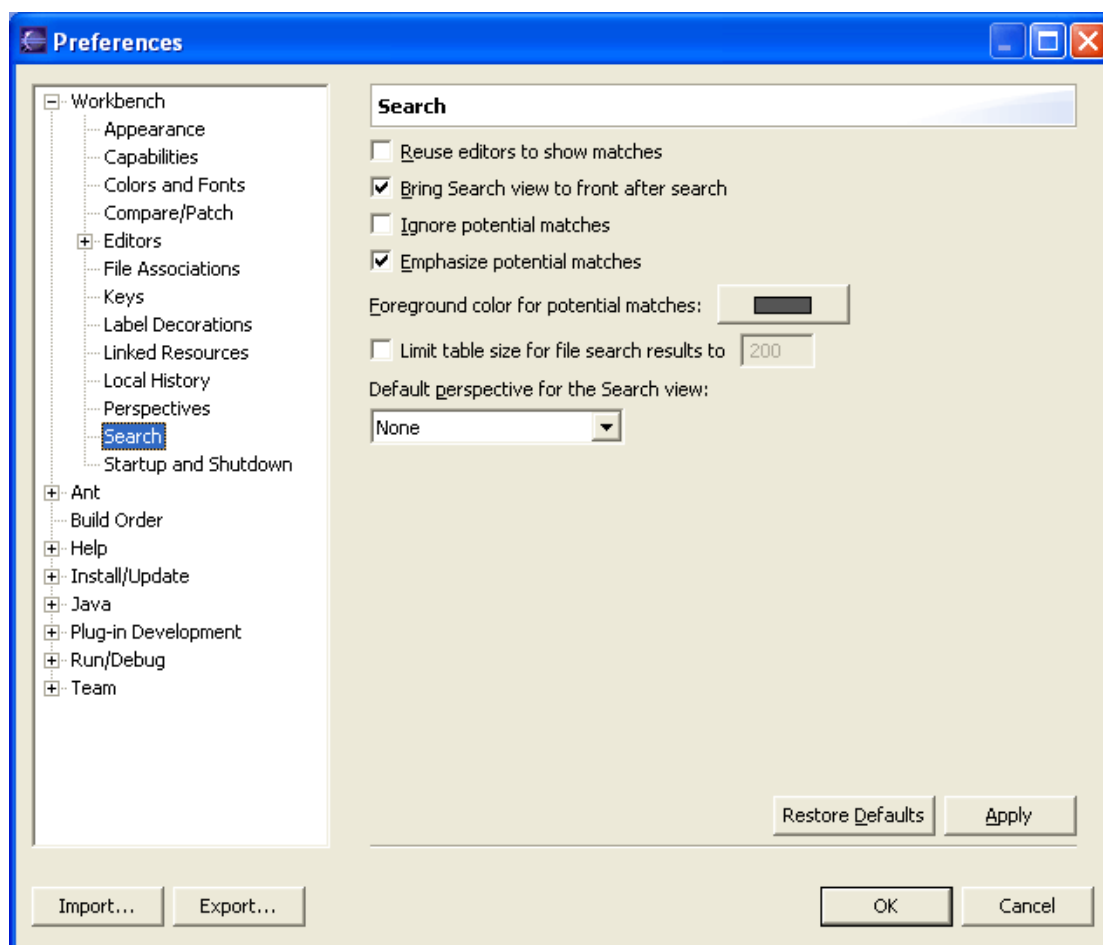


图 3.21

3.1.13 启动和关闭(Startup and Shutdown)

「启动和关闭」喜好设定页面可以选择要在工作台启动期间自动启动的外挂程序。

一般而言，要到需要外挂程序时才会加以启动。不过，某些外挂程序可能会指定它们希望在启动期间被启动。这个喜好设定页面可以选择在启动期间将会实际启动这其中的哪些外挂程序。

选项	说明	默认值
Prompt for workspace on startup(启动时发出工作区	如果开启这个选项，每次启动工作台时，工作台都会提示使用者要用哪个工作区。	开启

选项	说明	默认值
提示)		
Refresh workspace on startup(启动时重新整理工作区)	如果开启这个选项,在启动时,工作台会与档案系统同步化它的内容。	关闭
Confirm exit when closing last window(关闭最后一个窗口时确认结束)	如果开启这个选项,当关闭最后一个窗口时,工作台都会问使用者是否要结束。	开启

「启动和关闭」喜好设定页面看起来如下：

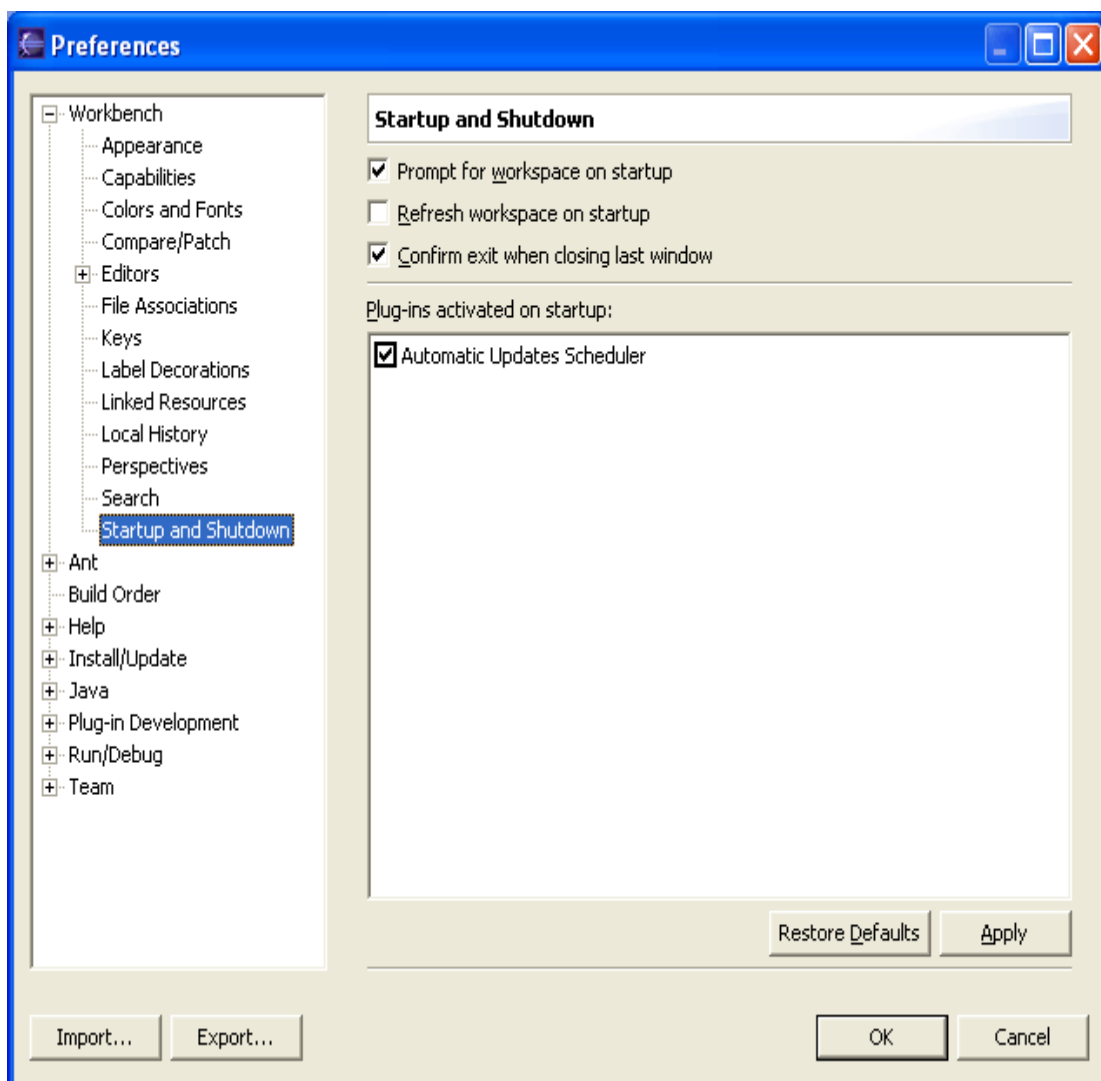


图 3.22

3.2 Ant

可以在 Ant 页面中变更下列喜好设定。

可以配置 Ant 的 "-find" 模拟要搜寻的建置档。

也可以配置 Ant 建置输出的颜色。

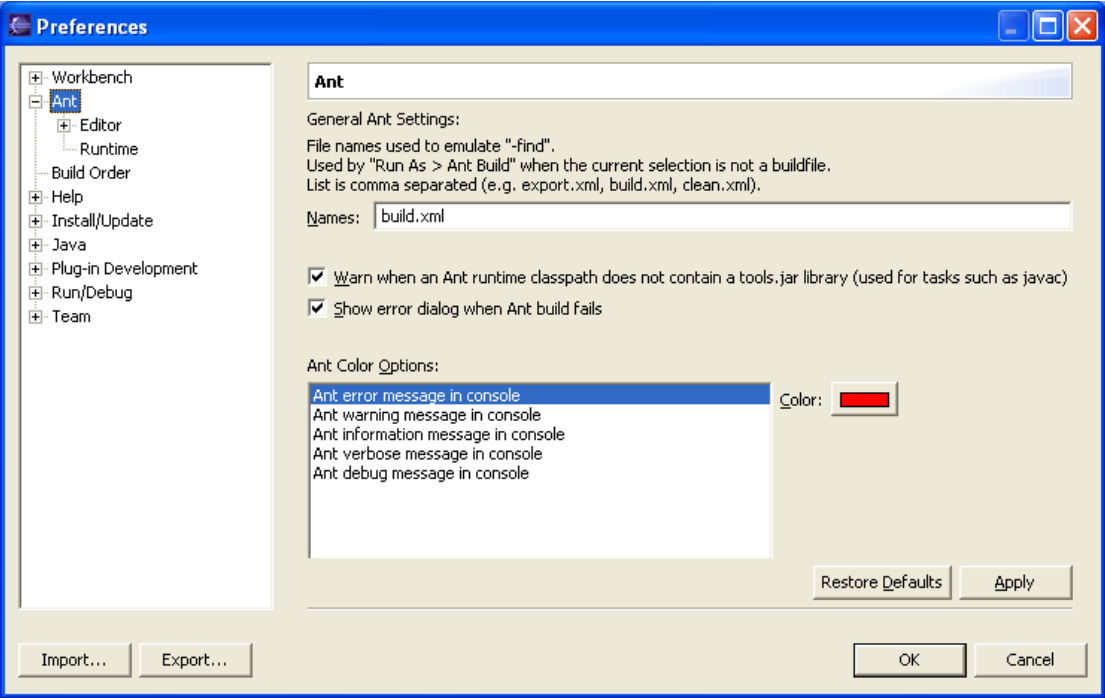


图 3.23

3.2.1 Ant 编辑器(Ant Editor)

可以在「Ant 编辑器」页面中变更下列喜好设定。

外观选项(Appearance Options)

选项	说明	默认值
Print margin column(打印边距直栏)	这个选项可以设定 Ant 编辑器的打印边距。	80

选项	说明	默认值
Displayed tab width(显示的栏标宽度)	这个选项用来控制在 Ant 编辑器中，要用多少空格来显示栏标。	4
Insert spaces for tabs when typing(在输入时插入栏标空格)	这个选项用来控制在 Ant 编辑器中输入时，是否要用空格来取代栏标。	关闭
Show overview ruler(显示概观标尺)	这个选项用来控制 Ant 编辑器右侧是否要显示概观标尺。	开启
Show line numbers(显示行号)	这个选项用来控制 Ant 编辑器左侧是否要显示行号。	关闭
Highlight current line(高亮度显示现行行)	这个选项用来控制是否要强调显示现行行。	开启
Show print margin(显示打印边距)	这个选项控制是否可以看到打印边距。	关闭
Appearance color options(外观颜色选项)	这个选项控制各种外观颜色。	

在外观页面中，提供 Ant 编辑器的外观选项。

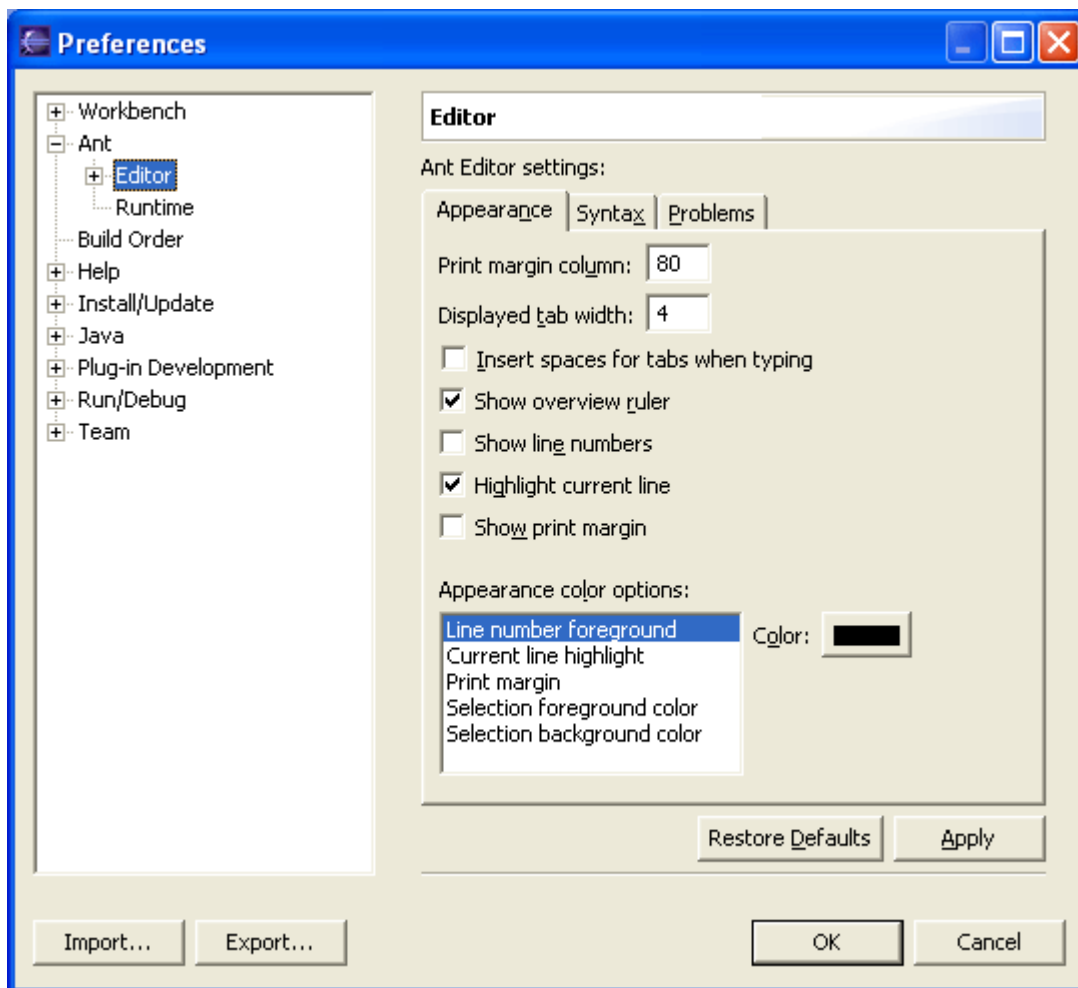


图 3.24

3.2.2 Ant 执行时期(Ant Runtime)

在类别路径页面上，可以将定义作业和类型的其它类别新增至 Ant 类别路径。

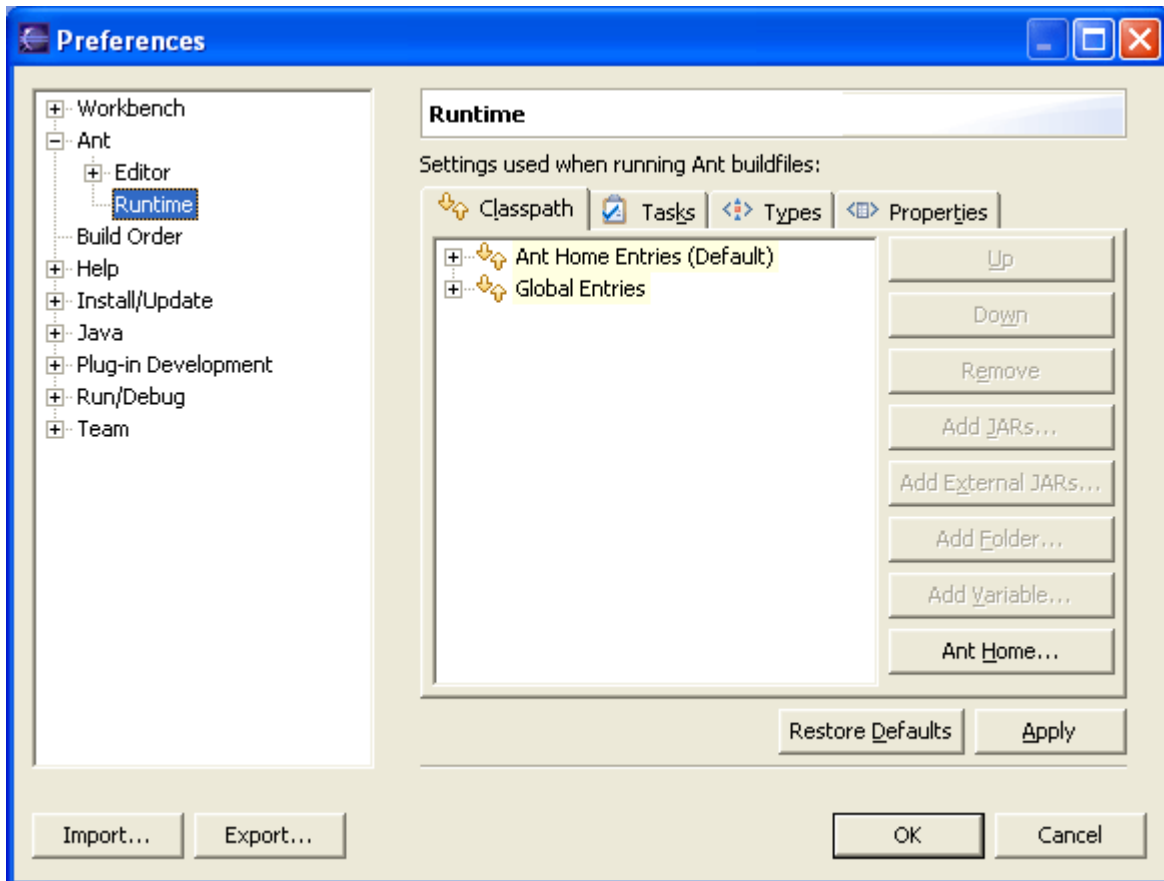


图 3.25

在作业页面上，可以新增定义于类别路径上的其中一个类别的作业。

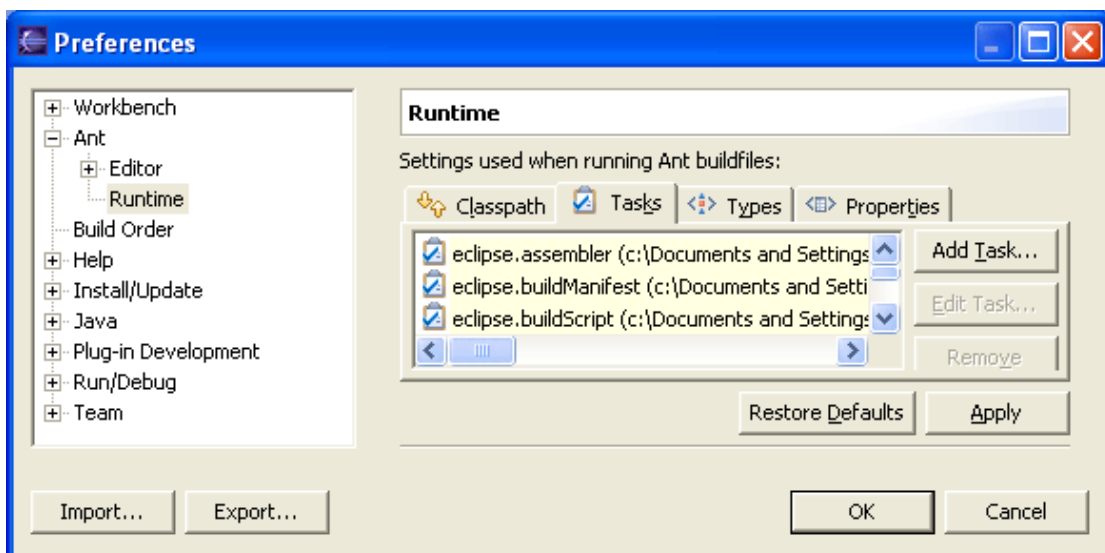


图 3.26

在类型页面上，可以新增定义于类别路径上的其中一个类别的类

型。

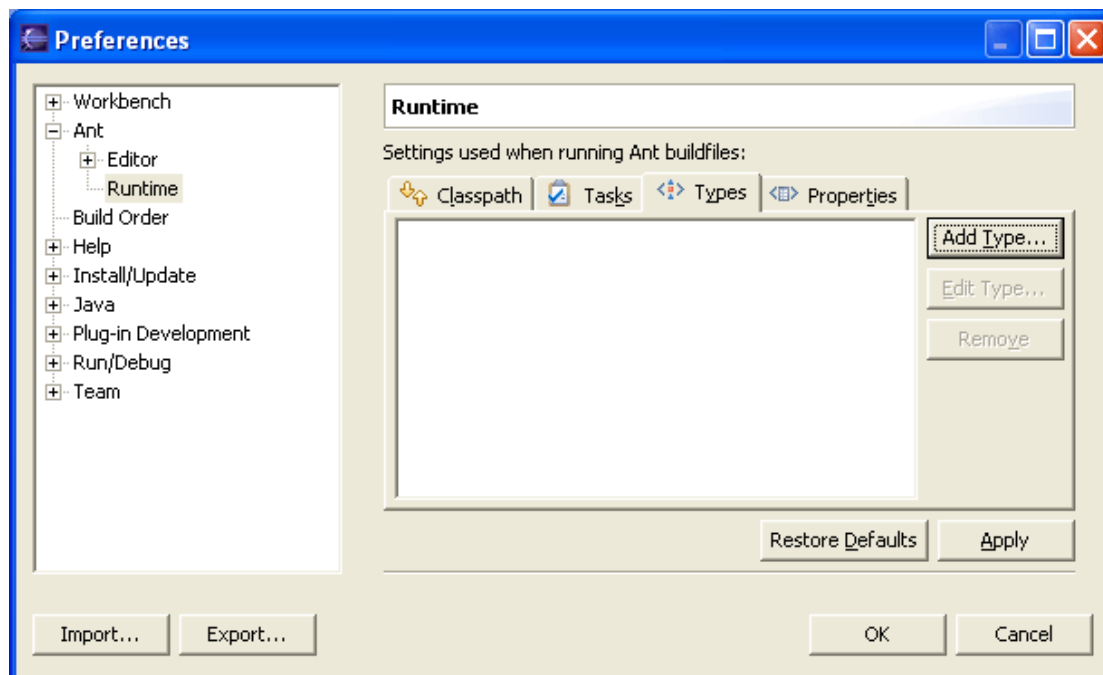


图 3.27

在内容页面中，可以新增要传送到 Ant 的内容和内容档。

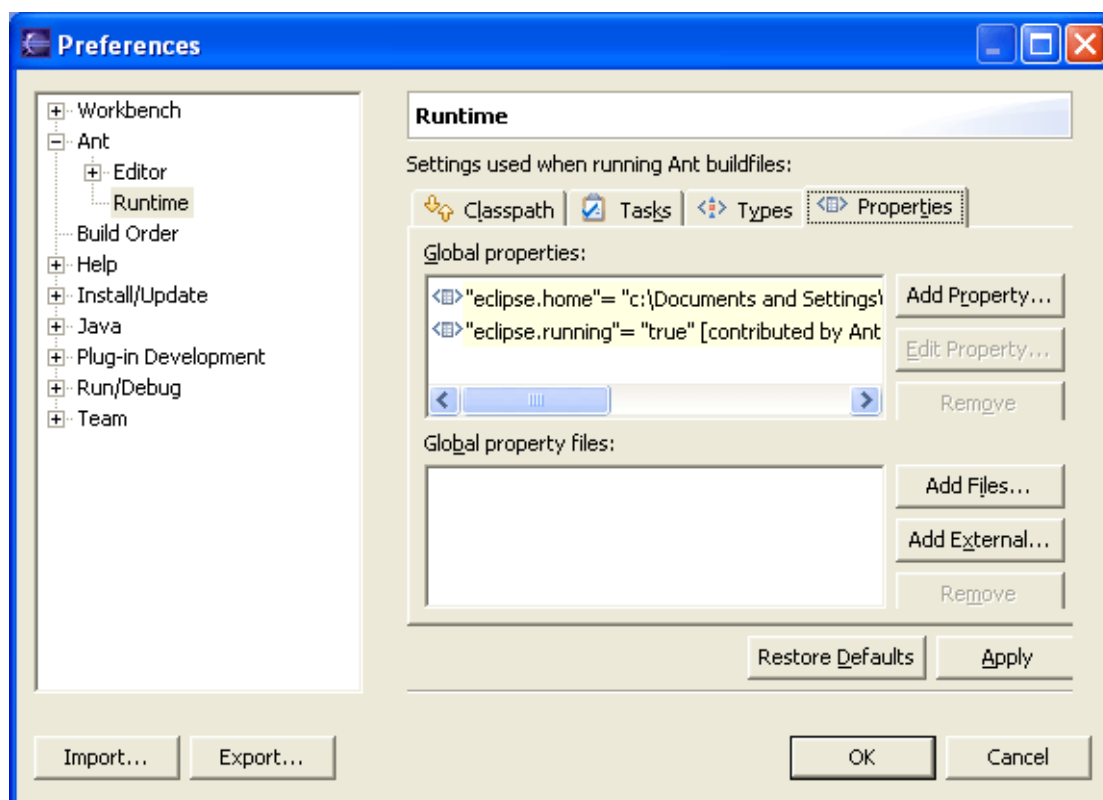


图 3.28

3.3 建置次序(Build Order)

通常，项目建置的次序是很重要的。比方说，如果某个项目需要在另一个项目中定义的 Java 类别，则必须先建置第一个项目的必备类别，然后再建置第一个项目。工作台可以让使用者明确地定义建置项目的次序。此外，使用者可以让平台藉由将项目参照解译为必备关系，来计算建置次序。在建置整个工作区或项目群组上，都会套用建置次序。

可以在「建置次序」喜好设定页面中变更这个次序。一开始，*使用预设建置器次序*选项是开启的，在此情况下，平台会计算建置次序。关闭这个选项时，就可以存取项目清单，而且可以操作项目的次序。请选取项目并使用上和下按钮来变更建置次序。可以使用 Add Project 及 Remove Project 按钮，在建置次序中新增和移除专案。从建置次序中移除的项目 *将会*被建置，但是会在建置次序中的所有项目都已建置完成之后才会建置。

在这个页面底端，有一个喜好设定可用来处理包含循环的建置次序。在理想的状况下，应该避免在项目之间使用循环参照。包含循环的项目在逻辑上仍属于单一项目，所以它们会尽可能收合成为一个项目。然而，如果一定要有循环，建置次序可能需要好几个迭代，才能正确地建置每一个项目。变更这个喜好设定时，会改变工作台尝试在建置次序上进行迭代多少次之后，才会放弃。

「建置次序」喜好设定页面看起来如下：

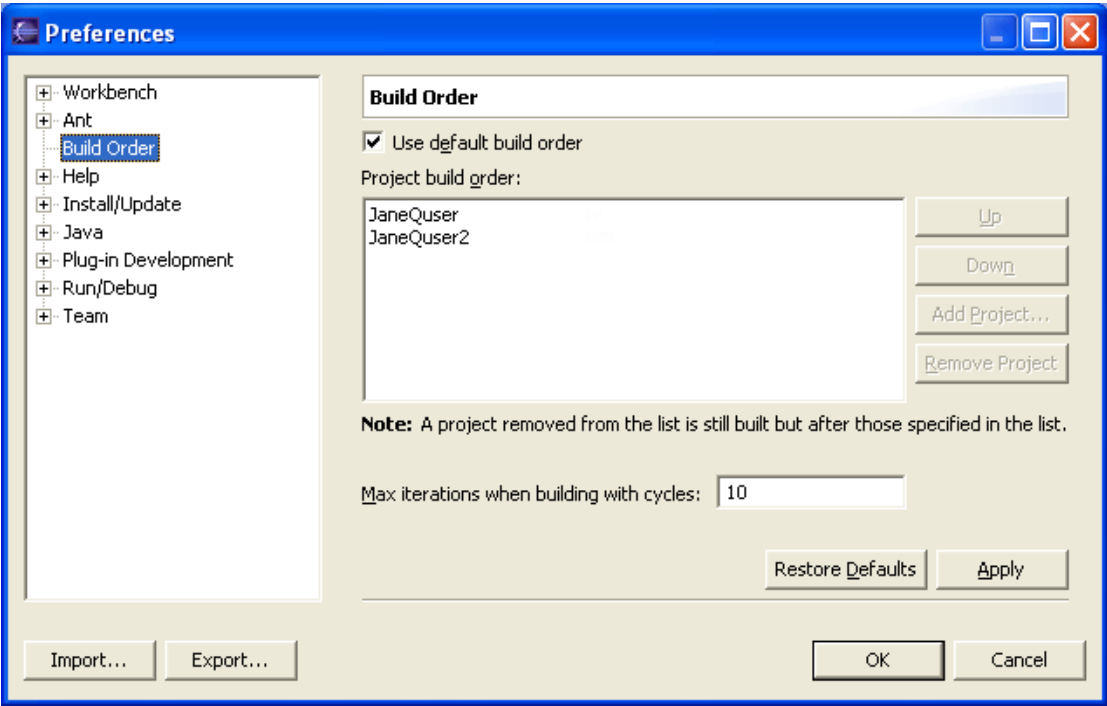


图 3.29

3.4 说明(Help)

在「说明」喜好设定页面上，可以指出如何显示说明书籍。

选项	说明	默认值
Always use external browsers(固定使用外部浏览器)	如果的系统支持内嵌 Web 浏览器，可能的话，说明会利用内嵌说明浏览器来显示说明，但仍可以使用这个选项。请选取它来强迫说明使用清单中的外部浏览器。	关闭
Current web browser adapter(现行 Web 浏览器配接器)	说明系统利用 Web 浏览器配接器，在外部浏览器中显示在线说明。如果有多个配接器可以在的系统开启浏览器，就会使用选取的配接器来显示说明。	视操作系统而定

<p>Custom browser command(自订浏览器指令)</p>	<p>如果从浏览器配接器清单中选取「自订浏览器」，就会使用这个字段来指定要启动浏览器程序的指令。如果 URL 不是浏览器程序可以接受的最后一个参数，请使用 "%1" 字符串来指出 URL 在指令中的位置。</p>	<p>"C:\Program Files\Internet Explorer\IEXPLORE.EXE" %1 - Windows , mozilla %1 - 其它平台（会显示这个字段）</p>
--	--	--

「说明」喜好设定页面看起来如下：

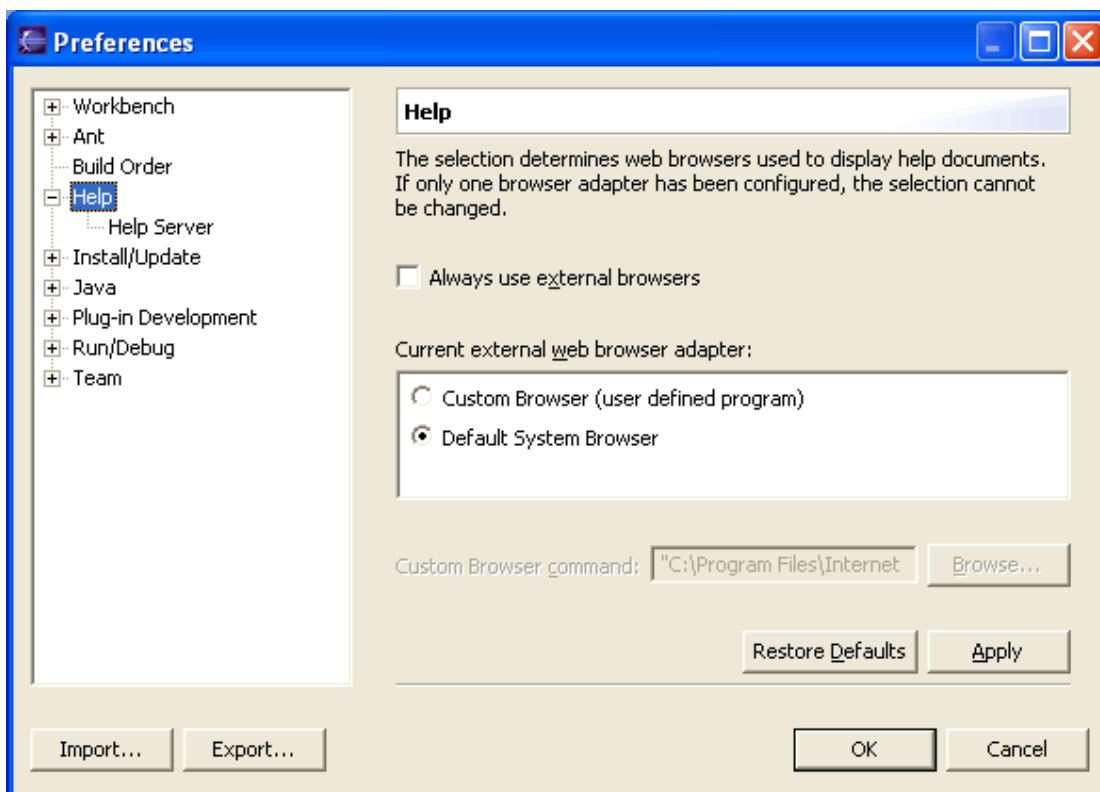


图 3.30

附注：在这个页面中所执行的选项会影响说明视图的呈现方式。如果选取的浏览器没有和 Internet Explorer 或 Mozilla 完全兼容，或者已经停用 JavaScript，则浏览器中所显示的说明视图可能是简化的版本。

3.4.1 说明服务器(Help Server)

说明系统包含一个内部服务器，可提供说明内容给浏览器。可以使用这个喜好设定页面来变更服务器所使用的接口和端口。只有当遇到问题且无法使用预设的喜好设定来检视说明时，才应该变更这些设定。

选项	说明	默认值
Host (主机)	服务器所使用的本端 IP 接口的名称或地址。	空白
Port (埠)	服务器要接听的 IP 埠。如果将这个值指定成 0，就会由操作系统来指定这个端口。	0

「说明服务器」喜好设定页面的外观如下：

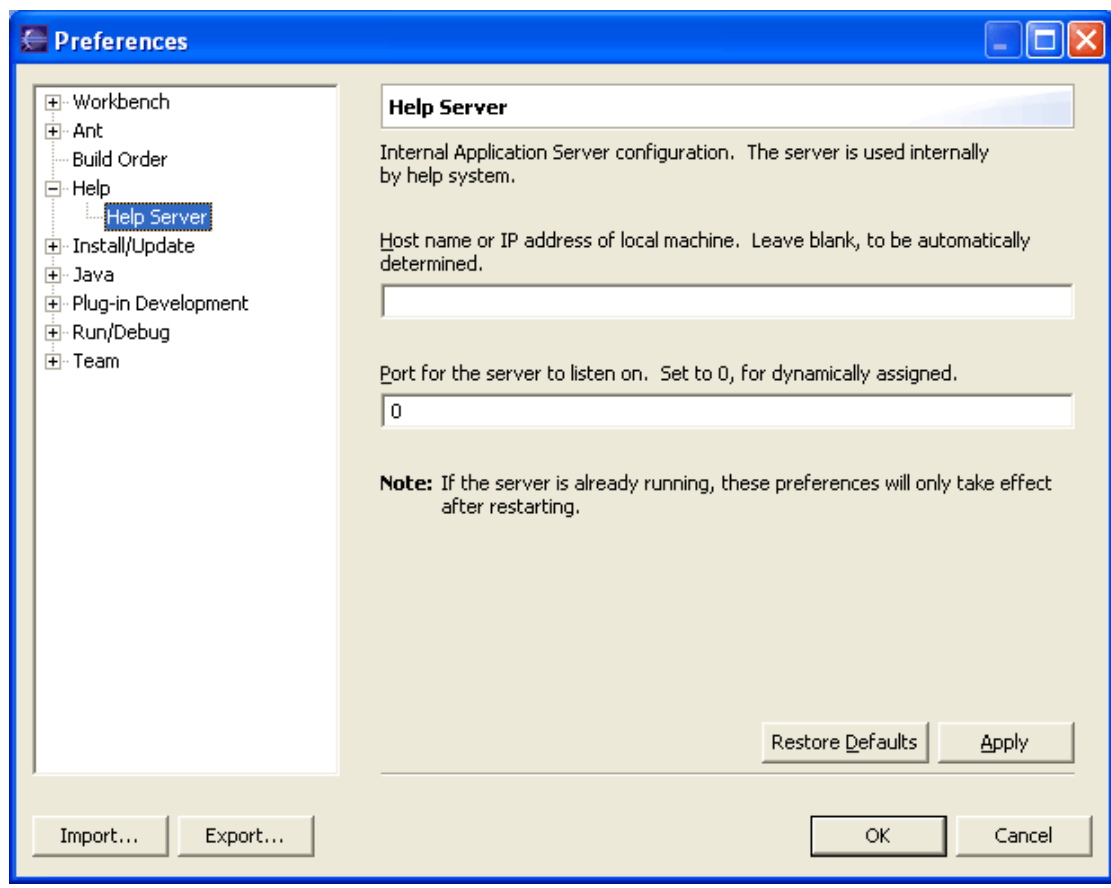


图 3.31

3.5 自动更新(Install/Update)

可以在「自动更新」页面中变更下列喜好设定：

选项	说明	默认值
Automatically search for updates and notify me (自动搜寻更新项目并通知我)	当选取时，更新管理程序会依照更新时程表所定义来自动搜寻更新项目	未选取
Update Schedule	每次启动时寻找更新项目，或在每天或每周某天的某个预定时间寻找更新项目。	在启动时

选项	说明	默认值
(更新时程表)		

3.6 Java

这个页面可以指出在一般 Java 设定方面的喜好设定。

选项	说明	默认值
Update Java views(更新 Java 视图)	<p>只在储存时(On save only) : 在储存编译单元前,不会更新「概要」视图以外之所有视图中显示的 Java 元素。视图会反映工作区的现行状态,但不会顾及到工作副本。</p> <p>编辑时(While editing) : 所有视图中显示的 Java 元素恒会反映工作区的现行状态,包括工作副本。</p>	编辑时
Action on double click in the Package Explorer(在「套件浏览器」中按两下后的动作)	<p>进入所选元素(Go into the selected element) : 当按两下储存器时,就会执行 Go Into 指令。请从导览菜单查看 Go Into。</p> <p>展开所选元素(Expand the selected element) : 当按两下储存器时,会将之展开,并显示其子项。</p>	展开所选取的元素
When opening a Type Hierarchy(当开启类型阶层时)	<p>开启新的「类型阶层」视景(Open a new Type Hierarchy Perspective) 只要一开启「类型阶层」视图,即开启新的「类型阶层」视景。</p> <p>在现行视景中显示「类型阶层」视图(Show the Type Hierarchy View in the current perspective) 在现行视景中显示「类型阶层」视图。</p>	在现行视景中显示「类型阶层」视图

选项	说明	默认值
	附注：在「工作台」喜好设定页面中，可以选择要在新窗口或现行窗口中开启新视景，或者以新视景取代现有的视景。	

3.6.1 外观(Appearance)

在这个喜好设定页面上，可以配置视图中之 Java 元素的外观。

选项如下：

选项	说明	默认值
Show method return types(显示方法传回类型)	当启用时，视图中的方法会显示传回类型。	关闭
Show override indicators in outline and hierarchy(以概要及阶层显示置换指示器)	当启用时，则在「概要」与「类型阶层」视图中，会针对已置换与实作的方法显示一个指示器。	开启
Show members in Package Explorer(在套件浏览器中显示成员)	当启用时，亦会显示 Java 文件与类别文件层次下的 Java 元素。	开启
Compress package name segments(压缩套件名称区段)	当启用时，则会根据压缩型样压缩套件名称。	关闭

选项	说明	默认值
Stack views vertically in the Java Browsing perspective(垂直堆栈「Java 浏览」视景中的视图)	当启用时，则「Java 浏览」视景中的视图会采用垂直（而非水平）方式堆栈。	关闭

3.6.2 类别路径变量(Classpath variables)

可配置的变量(Configurable variables)

可在「Java 建置路径」中使用类别路径变量，以免参照本端档案系统。当使用变量项目时，类别路径中只含有一个变量，且建置路径可供整个团队共享。必须在这个页面中配置变量的值。

指令	说明
New...(新建...)	新增变量项目。在产生的对话框中，指定新变量的名称和路径。可以按一下 File 或 Folder 按钮来浏览以找出路径。
Edit...(编辑...)	可让编辑所选取的变量项目。在产生的对话框中，编辑该变量的名称与（或）路径。可以按一下 File 或 Folder 按钮来浏览以找出路径。
Remove(移除)	移除选取的变量项目。

保留类别路径变量(Reserved class path variables)

某些类别路径变量会设定在内部，且无法在「类别路径变量」喜好设定中变更：

- JRE_LIB：这个保存档中含有目前所用之 JRE 的执行时期 JAR 檔。
- JRE_SRC：为目前所用 JRE 的程序文件保存文件。
- JRE_SRCROOT：为目前所用 JRE 之程序文件保存文件中的根路径。

3.6.3 程序代码格式制作器(Code Formatter)

这个页面中的预览窗格会示范这些选项对编辑器中之 Java 程序代码所产生的结果。

选项	说明	默认值
Insert a new line before an opening brace(在左大括号前插入新行)	编辑器在新的左大括号前插入换行符号。换句话说，左大括号恒起自新行的开头。	关闭
Insert new lines in control statements(在控制陈述式中插入新行)	编辑器在新的控制陈述式前插入一行。换句话说，控制陈述式（如：if、else、catch、finally 等）恒起自新行的开头。	关闭
Clear all blank lines(清除所有空白行)	编辑器删除档案中的所有空白行。	关闭
Insert new line between 'else if'(在	编辑器在 else-if 陈述式中之 "else" 与 "if" 字之间插入一行新行。	关闭

选项	说明	默认值
'else if'之间插入新行)		
Insert a new line inside an empty block(在空区块内插入新行)	编辑器在空大括号间插入一个换行。换句话说，在一个空大括号组中的左右大括号会出现在不同行。一个例外情况是，如果右大括号后跟着一个关键词，则两个大括号会出现在同一行。	开启
Maximum line length(行长度上限)	这是任何单行的最大长度。当字行超过这个长度时，则会折行。如果输入 0，则完全停用折行特性。	80
Compact assignment(精简指派)	编辑器会移除变数与指派陈述式间的任何空格，使其不对称（如 a= b; ）。	关闭
Insert a space after a cast(在强制转型后插入空格)	编辑器会在强制转型与下列表示式之间插入一个空格。	开启
Insert tabs for indentation, not spaces(插入 tab（而非空格）以内缩)	编辑器使用 tab（而非空格）来呈现内缩。	开启
Number of spaces representing an indentation level(代表内缩层次的空格数目)	如果编辑器是使用空格而非 tab 来呈现内缩，则这是指出一个内缩是由多少空格组成。	4

选项	说明	默认值
Preview pane(「预览」窗格)	使用这个页面目前所示的设定值，以范例显示 Java 程序代码的模样。	n/a

3.6.4 程序代码产生(Code generation)

程序代码产生喜好设定分成两个区段：

- 名称
- 程序代码和批注

名称(Names)

这个页面定义字段 (Static 和非 Static)、参数和区域变量的命名惯例。对于每一个变量类型，有可能配置前缀或字尾清单，或两者的清单。

产生 Getter 和 Setter 动作，以及所有建立字段、参数和区域变量的动作和「快速修正」提议，都会使用命名惯例。

动作	说明
Edit...(编辑...)	开启一个对话框，编辑目前选取之变量类型的前缀和字尾清单

程序代码和批注(Code and Comments)

程序代码和批注页面含有产生程序代码之动作所使用的程序代码模板。模板含有当套用模板时将替代的变量。某些变量可用在所有模板中，某些变量则是模板特有的。

动作	说明
Edit...(编辑...)	开启一个对话框，编辑目前选取的程序代码范本。
Import...(汇)	从档案系统汇入程序代码模板。

动作	说明
入...)	
Export...(汇出...)	汇出所有选取的程序代码模板至档案系统。
Export All...(汇出全部...)	汇出所有程序代码模板至档案系统。
Automatically add comments for new methods and types(自动新增方法和类型的批注)	这个设定指定批注程序代码模板是否会自动新增至所有新的方法。如果停用，仅在明确地新增批注（如使用新增 Javadoc 批注动作）时，才会使用批注程序代码模板。请注意，这个设定并不套用至程序代码模板（如新建 Java 档案）中所含的批注

批注范本(Comment templates)

批注模板可包含 `${tags}` 变量，这个变量将被已加注元素的标准 Javadoc 标示 (`@param`, `@return..`) 所替代。此外，「置换方法」批注可包含 `${see_to_overridden}` 范本

- 建构子批注：指定新建构子批注的范本
- 类型批注：指定新类型批注的模板。请注意，可在「新建 Java 文件」范本中参照这个范本
- 方法批注：指定新方法（不置换基础类别中的方法）批注的范本
- 置换方法批注：指定新方法（置换基础类别中的方法）批注的范本。依预设，批注会定义成非 Javadoc 批注（Javadoc 将把这个批注换成已置换方法的批注）。如果想要的话，可以将这个批注变更为真实的 Javadoc 批注

新建 Java 文件范本(New Java files template)

当建立新档案时，「类别」和「接口」精灵就会使用「新建 Java 文件」模板。模板可以指定要新增批注之处。请注意，模板可以含有 `${typecomment}` 变量，这个变量将被类型批注模板的评估值所替代。

Catch 区块主体范本(Catch block body template)

当建立 catch 区块主体时，就会使用「Catch 区块主体」模板。它可以使用 `${exception_type}` 和 `${exception_var}` 变数。

方法主体范本(Method body template)

当建立含有主体的新方法时，就会使用「方法主体」模板。它含有解析为 return 陈述式或/和 super 呼叫的 `${body_statement}` 变量。

建构子主体范本(Constructor body templates)

当建立含有主体的新方法或建构子时，就会使用「建构子主体」模板。它含有解析 super 呼叫的 `${body_statement}` 变量。

「程序代码范本」对话框(Code Template dialog)

对话框中的字段与按钮如下：

选项	说明
Description(说明)	范本的说明
Pattern(型样)	范本的型样。
Insert Variables...(插入变数...)	显示预先定义之模板特有变量的清单。

3.6.5 编译器(Compiler)

下列各段将分别说明编译器的喜好设定：

■ 问题

- 样式
- 相容和类别档
- 建置路径

问题(Problems)

选项	说明	默认值
Unreachable code(无法呼叫到的程序代码)	无法呼叫到程序代码，可选择性地报告成错误、警告，或者加以忽略。字节码一律产生最佳化程序代码。请注意，根据 Java 语言规格，无法呼叫到的程序代码应该是一个错误。	错误
Unresolvable import statements(无法解析的 import 陈述式)	无法解析的 import 陈述式可选择性地报告成错误、警告，或加以忽略。请注意，根据 Java 语言规格，无法解析的 import 陈述式应该是一个错误。	错误
Unused local variables (i.e. never read)(未使用的区域变量（如从未读取）)	当启用时，编译器会针对未用的区域变量（亦即：从未读取的变量），发出错误或警告。	忽略
Unused parameters (i.e. never read)(未使用的参数（如从未读取）)	当启用时，编译器会针对未用的方法参数（亦即：从未读取的参数），发出错误或警告。	忽略
Unused imports(未用的汇入)	当启用时，编译器会针对未用的汇入参照，发出错误或警告。	警告
Unused private types, methods or fields (未用的 private 类型、方法或字段)	当启用时，每当宣告 Private 方法或字段时，但从未在同一单元内使用时，编译器将发出错误或警告。	忽略

Usage of non-externalized strings(未提出字符串的用法)	当启用时，编译器将为未提出的字符串文字发出错误或警告（如，未标示的 <code>// \$NON-NLS-<n>\$</code> ）。	忽略
Usage of deprecated API(已停用的 API 的用法)	当启用这个选项时，编译器会将使用已停用的 API 标为错误或警告。	警告
Signal use of deprecated API inside deprecated code(已停用的程序代码内之已停用的 API 的信号使用)	一旦启用，编译器将发出信号，指出在已停用的程序代码内使用已停用的 API。问题的严重性是由「已停用的 API 的用法」选项来控制。	关闭
Maximum number of problems reported per compilation unit(各编译单元所能报告的问题数上限)	指定各编译单元所能报告的问题数上限。	100

样式(Style)

选项	说明	默认值
Methods overridden but not package visible(已置换但套件看不到的方法)	套件的预设方法在另一套件中看不到，因此无法置换。当启用这个选项时，编译器会将这类情况标为错误或警告。	警告
Methods with a constructor name(建构子名称中的方法)	如果以建构子名称来命名方法，通常会被视为较差的程序设计风格。当启用这个选项时，编译器会将这类情况标为错误或警告。	警告

Conflict of interface method with protected 'Object' method(接口方法与受保护的「对象」方法发生冲突)	当启用时，每当接口定义一个与非继承「对象」方法不兼容的方法时，编译器将发出错误或警告。直到解决这个冲突之前，将无法实作如此的接口，如 <pre>interface I { int clone(); }</pre>	警告
Hidden catch blocks(隐藏的 catch 区块)	在本端环境下对于 try 陈述式而言，某些 catch 区块可能会隐藏其它者，例如： <pre>try { throw new java.io.CharConversionException(); } catch (java.io.CharConversionException e) { } catch (java.io.IOException e) {}.</pre> 当启用这个选项时，编译器会针对对应至所检查之异常状况的快取区块隐藏，发出错误或警告。	警告
Non-static access to a static member(Static 成员的非 Static 存取权)	当启用时，每当以表示式接收器存取 Static 字段或方法时，编译器将发出错误或警告。应该以类型名称限定 Static 成员的参照。	警告
Access to a non-accessible member of an enclosing type(存取含括类型中无法存取的成员)	当启用时，只要其模拟存取含括类型中无法存取的成员，编译器即会发出错误或警告。这类存取可能涉及效能。	忽略
Assignment has	当启用时，每当指派没有效果（如 'x = x'）时，	警告

no effect (e.g. 'x = x')(指定没有生效 (例如 'x=x'))	编译器将发出错误或警告。	
Using a char array in string concatenation(在字符串连结中使用 char 数组)	当启用时，每当在下列「字符串」连结中使用 char[] 表示式时，编译器就会发出错误或警告： "hello" + new char[]{'w','o','r','l','d'}	警告

相容和类别档(Compliance and Class files)

选项	说明	默认值
Compiler compliance level(编译器兼容层次)	指定 JDK 编译器兼容层次。	1.3
Use default compliance settings(使用预设兼容设定)	当启用时，在编译器的兼容层次方面，会套用预设的兼容设定。	开启
Generated class files compatibility(所产生的类别档兼容性)	指定所产生的类别档兼容性。	1.1
Source compatibility(程序文件兼容性)	指定程序文件和 1.3 或 1.4 兼容。从 1.4 开始，"assert" 为保留给主张支持的关键词。	1.3
Report 'assert' as identifier(将	当启用时，只要 'assert' (为 JDK 1.4 中的保留关键词) 被当成识别码使用，编译器即会发出错误或警告。	忽略

'assert'报告成识别码)		
Add variable attributes to generated class files(新增变量属性到产生的类别文件中)	当启用时，会在类别文件中新增变量属性。这会让区域变量名称显示在除错器中(位于明确指定变量之处)。产生的 .class 档会变大。	开启
Add line number attributes to generated class files(新增行号属性到产生的类别文件中)	当启用时，会在类别文件中新增行号信息。这会在除错器中强调显示出程序代码。	开启
Add source file name to generated class file(新增程序文件名称到产生的类别档中)	当启用时，会在类别文件中新增程序文件名称。这会让除错器显示对应的程序代码。	开启
Preserve unused local variables(保留未用的区域变量)	当启用时，则不会将未用的区域变量(亦即，从未读取)从类别档中除去。如果除去这项，有可能会改变除错。	开启

建置路径(Build Path)

选项	说明	默认值
Incomplete build path(不完整的建置路径)	指出当类别路径上的项目不存在、不合规定或看不见(如关闭了参照项目)时，所报告的问题的严重性。	错误
Circular	指出在循环中并入项目时所报告的问题的严重	错误

dependencies(循环相依项)	性。	
Duplicated resources(重复的资源)	指出当多次出现的资源将复制到输出位置时所报告的问题的严重性。	警告
Abort building on build path errors(当建置路径错误时中止建置)	容许如果类别路径无效，将建置器切换至中止。	开启
Scrub output folders on full build(进行完整建置时清除输出数据夹)	指出是否容许「Java 建置器」在执行完整建置作业时清除输出数据夹。	开启
Enable using exclusion patterns in source folders(启用在来源数据夹中使用排除型样)	当停用时，项目类别路径上没有项目可与排除型样相关联。	开启
Enable using multiple output locations for source folders(启用对来源数据夹使用多个输出位置)	当停用时，项目类别路径上没有项目可与特定输出位置相关联，因而防止使用多个输出位置。	开启
Filtered resources(过滤)	以逗点分格方式列出不复制到输出数据夹中的档案型样。	''

的资源)		
------	--	--

3.6.6 Java 编辑器(Java editor)

这个页面可以设定如下的 Java 编辑器喜好设定：

- 外观
- 语法
- 程序代码协助
- 问题指示

外观(Appearance)

外观指定 Java 编辑器的外观。

选项	说明	默认值
Displayed tab width(Tab 的显示宽度)	指定 Tab 的显示宽度 (以空格为单位)。	4
Print margin column(打印边距直栏)	指定其后要显示打印边距的直栏。 如果要显示打印边距，请启 Show print margin 选项；如果要指定打印边距的颜色，请使用 Appearance color options 喜好设定。	80
Synchronize outline selection on cursor move(在光标移动时将概要选项同步化)	当启用时，「概要」视图恒会在 Java 编辑器中选取含括光标的 Java 元素。	关闭
Show overview ruler(显示概观标尺)	当启用时，Java 编辑器右边框中会出现概观标尺，并显示整个可视文件的问题。	开启

选项	说明	默认值
Show line numbers(显示行号)	当启用时，Java 编辑器左边框中的垂直标尺会显示可视文件的行号。 可在 Appearance color options 中指定行号的颜色。	关闭
Highlight matching brackets(强调显示对称的括号)	当启用时，只要光标是位于小括号、方括号或大括号旁，即会强调显示其相对的左或右括号。 可在 Appearance color options 中指定括号的强调显示颜色。	开启
Highlight current line(强调显示现行行)	当启用时，则会强调显示光标所在现行行的背景。 可在 Appearance color options 中指定现行行的背景颜色。	开启
Show print margin(显示打印边距)	当启用时，会显示打印边距。 可使用 Print margin column 和 Appearance color options 喜好设定，来判定打印边距的位置与颜色。	关闭
Appearance color options(外观颜色选项)	可在这里指定各种 Java 编辑器外观特性的颜色。 <ul style="list-style-type: none"> ■ Line number foreground(行号前景)：行号的颜色。 ■ Matching brackets highlight(相符的方括号强调显示)：括号的强调显示颜色。 ■ Current line highlight(现行行强调显示)：现行行的强调显示颜色。 ■ Print margin(打印边距)：打印边距的颜色。 ■ Find scope(寻找范围)：寻找范围的颜色。 ■ Linked position(链接的位置)：程序代码辅助中所用链接位置的颜色。 ■ Link(链接)：链接颜色 	预设颜色

语法(Syntax)

语法指定如何展现 Java 程序代码。

选项	说明	默认值
Background color(背景颜色)	System default(系统默认值)：操作系统所提供的预设背景颜色。 Custom(自订)：使用者定义的背景颜色。	系统默认值
Foreground(前景)	<p>下列的 Java 程序文件片段可以不同的颜色与样式展现：</p> <ul style="list-style-type: none"> ■ Multi-line comment(多行批注)：批注的格式为 '/*...*/' ■ Single-line comment(单行批注)：批注的开头为 '//' ■ Keywords(关键词)：所有 Java 关键词。 ■ Strings(字符串)：Java 字符串和字符，以单引号与双引号括住 ■ Others(其它)：预设 Java 程序代码 ■ Task tags(作业标示)：批注中的作业标示 ■ Javadoc keywords(Javadoc 关键词)：Javadoc 中所用的关键词，开头为 '@' ■ Javadoc HTML tags(Javadoc HTML 标示)：Javadoc 中所用的 HTML 标示 ■ Javadoc links(Javadoc 链接)：{@link reference} 标示 ■ Javadoc others(Javadoc 其它)：预设 Javadoc 文字 	预设颜色和样式
Preview(预览)	显示和现行颜色和样式有关之 Java 程序代码的预览。	n/a

程序代码辅助(Code assist)

程序代码辅助指定程序代码辅助的行为与外观。

选项	说明	默认值
Completion inserts/Comple	如果开启了完成插入(Completion inserts)，完成文字将插入在脱字符号 (^) 位置，所以它绝	完成插入

选项	说明	默认值
ion overwrites(完成 插入/完成改写)	不会改写任何现有的文字。 如果开启了完成改写(Completion overwrites),完成文字将取代脱字符号 (^) 位置之后直到字尾的字符。	
Insert single proposals automatically(自动插入单一提 议)	当启用时,程序代码辅助会自动选择与插入单一 提议。	开启
Show only proposals visible in the invocation context(在呼叫 环境定义中只显 示可见的提议)	当启用时,则会使用显示规则来限制 Java 元素 的提议。例如,不显示其它类别的 private 字 段提议。	开启
Present proposals in alphabetical order(按字母顺 序来提供提议)	当启用时,则提议会按字母顺序排序。	关闭
Automatically add import instead of qualified name(自动新增汇 入取代完整名称)	当启用时,则位于其它套件中的类型提议会呼叫 以新增对应的汇入宣告。否则,会完整插入类型。	开启
Fill argument names on method completion(方法	当启用时,则在选择方法提议时,会插入该方法 之宣告中指定的自变量名称。	关闭

选项	说明	默认值
完成时填入自变量名称)		
Enable auto activation(启用自动启动)	当启用时，可自动呼叫程序代码辅助。可在自动启动延迟(Auto activation delay)、自动启动 Java 触发(Auto activation triggers for Java)以及自动启动 Javadoc 触发(Auto activation triggers for Javadoc)中指定自动呼叫的条件。	开启
Auto activation delay(自动启动延迟)	一旦从遇到自动启动触发字符起到输入新字符止的时间，超过自动启动延迟，即会呼叫程序代码辅助。	500
Auto activation triggers for Java(自动启动 Java 触发)	如果 Java 程序代码内键有一个触发字符(但不是键于 Javadoc 批注内)，则一旦在自动启动延迟逾时前未输入其它字符，即会呼叫程序代码辅助。	'.'
Auto activation triggers for Javadoc(自动启动 Javadoc 触发)	如果 Javadoc 内键有一个触发字符，则一旦在自动启动延迟逾时前未输入其它字符，即会呼叫程序代码辅助。	'@'
Code assist color options(程序代码辅助颜色选项)	<p>下列程序代码辅助 UI 元素所用的颜色：</p> <ul style="list-style-type: none"> ■ Completion proposal background(完成提议背景)：完成提议窗口的背景颜色 ■ Completion proposal foreground(完成提议前景)：完成提议窗口的前景颜色 ■ Method parameter background(方法参数背景)：参数窗口的背景颜色 ■ Method parameter foreground(方法参数前景)：参数窗口的前景颜色 ■ Completion overwrite background(完成改写背景)：完成改写窗口的背景颜色 	预设颜色

选项	说明	默认值
	■ Completion overwrite foreground(完成改写前景)：完成改写窗口的前景颜色	

附注(Annotations)

附注指定何时及如何显示附注。

选项	说明	默认值
Analyze annotations while typing(输入时分析附注)	如果启用，则在使用者输入时，就会更新附注。否则，直到编译 Java 档案之前，都不会更新附注。	开启
Indicate annotations solvable with Quick Fix in vertical ruler(在垂直标尺中指出可利用快速修正解决的注释)	针对每一个可透过快速修正解决的注释，在 Java 编辑器左边框的垂直标尺中，显示一个灯泡。	开启
Annotation presentation(附注呈现方式)	对于每一类型的附注，可以指定以文字、以概观标尺或两者显示附注以何种颜色展现附注	

范本(Templates)

「模板」喜好设定页面可以建立新模板与编辑现有模板。模板可方便程序设计师快速插入常重复出现的程序代码型样。

下列按钮可以操作与配置模板：

动作	说明
New... (新建...)	开启对话框以建立新模板。

动作	说明
Edit...(编辑...)	开启对话框以编辑目前所选取的范本。
Remove(移除)	移除所有选取的范本。
Import...(汇入...)	从档案系统汇入模板。
Export...(汇出...)	将选取的所有模板汇出至档案系统。
Export All...(汇出全部...)	将所有模板汇出至档案系统。
Enable All(全部启用)	启用所有范本。
Disable All(全部停用)	停用所有范本。
Use Code Formatter(使用程序代码格式制作器)	当启用时，在插入之前，会先根据程序代码格式制作器(Code Formatter)喜好设定中指定的程序代码格式设定规则，来设定模板的格式。否则，范本会按原样插入，但内缩将不正确。 请参阅「程序代码格式制作器」喜好设定页面

「范本」对话框(Template dialog)

新建模板与编辑现有模板所用的对话框相同，以下是其说明。

对话框中的字段与按钮如下：

选项	说明
Name(名称)	模板的名称。
Context(环境定义)	环境定义是决定哪些地方可使用模板，以及决定可用的一组预先定义的模板变量。 ■ Java：Java 环境定义

选项	说明
	■ Javadoc : Javadoc 环境定义
Description(说明)	模板的说明，会在使用者选择模板时显示。
Pattern(型样)	范本的型样。
Insert Variables...(插入变数...)	列出预先定义的环境定义特定变量。

模板变量(Template variables)

Java 和 Javadoc 环境定义皆会定义下列变量：

变数	说明
<code>\${cursor}</code>	指出当离开模板编辑模式时的光标位置。当离开模板编辑模式时，如果希望光标应移至另一位置，而非移至模板尾端，则可善用这项。
<code>\${date}</code>	评估成现行日期。
<code>\${dollar}</code>	评估成钱币符号 '\$'。 另外，可以使用两个货币符号： '\$\$'。
<code>\${enclosing_method}</code>	评估成含括名称的名称。
<code>\${enclosing_method_arguments}</code>	评估成含括方法的以逗点区隔之自变量名称清单。这个变量有助于为许多方法产生 log 陈述式。
<code>\${enclosing_package}</code>	评估成含括套件的名称。
<code>\${enclosing_project}</code>	评估成含括项目的名称。
<code>\${enclosing_type}</code>	评估成含括类型的名称。

变数	说明
<code>\${file}</code>	评估成档案的名称。
<code>\${return_type}</code>	评估成含括方法的传回类型。
<code>\${time}</code>	评估成现行时间。
<code>\${user}</code>	评估成使用者名称。

此外，Java 环境定义会定义下列变量：

变数	说明
<code>\${array}</code>	评估成已宣告之数组名的提议。
<code>\${array_element}</code>	评估成已宣告之数组的元素名称的提议。
<code>\${array_type}</code>	评估成已宣告之数组的元素类型的提议。
<code>\${collection}</code>	评估成实作 <code>java.util.Collection</code> 之已宣告集成的提议。
<code>\${index}</code>	评估成未宣告之数组索引迭代的提议。
<code>\${iterator}</code>	评估成未宣告之集成迭代的提议。

3.6.7 JRE 安装(JRE installations)

「类别路径变量」喜好设定

选项	说明
Add...(新增...)	<p>将新的 JRE 定义新增至工作台中。在产生的对话框中，指定下列：</p> <ul style="list-style-type: none"> ■ JRE 类型：(从下拉列表中选取一种 VM 类型) ■ JRE 名称：输入这个 JRE 定义的名称 ■ JRE 起始目录：输入或浏览以选取这项 JRE 安装的根目录 ■ Javadoc URL：输入或浏览以选取 URL 位置。这个位置

选项	说明
	<p>供 Javadoc 汇出精灵做为默认值，并供「开启外部 Javadoc」动作使用。除错器逾时值：输入这个 JRE 之除错器的预设逾时值（以毫秒为单位）</p> <p>■ 如果要让这个 JRE 采用预设链接库位置，请勾选这个勾选框；如果要输入或浏览以便为下列档案指定链接库位置，请清除这个勾选框：</p> <ul style="list-style-type: none"> □ JAR 檔（例如：classes.zip） □ 程序文件（例如：source.zip） <p>可以按一下「Browse」按钮，以浏览并找出所要的路径。</p>
Edit...(编辑...)	可让编辑所选取的 JRE。
Remove(移除)	将所选取的 JRE 从工作台中移除。
Search...(搜寻...)	自动搜寻已安装在本端档案系统的 JRE 并在工作区中建立对应的 JRE 定义。

3.6.8 JUnit

选项	说明	默认值
Show the JUnit results view only when an error or failure occurs (只有在发生错误或失败时,才显示 JUnit 结果视图)	当启用时，则只有在发生错误或失败时才会将 JUnit 视图带至前面。	开启
Stack trace filter patterns(堆栈追踪过滤器型样)	测试失败时，不应该显示在堆栈追踪的套件、类别或型样。	预设过滤器型样

3.6.9 新专案(New project)

选项	说明	默认值
Source and output folder(程序文件与输出数据夹)	<ul style="list-style-type: none"> ■ 专案：将程序文件与输出位置皆设为项目的根目录。 ■ 资料夹：可个别设定程序文件和输出位置。如果要指定程序文件与输出位置，请设定来源数据夹名称(Source folder name)与输出位置名称(Output location name)。 	专案
Source folder name(来源资料夹名称)	程序文件的位置。	'src'
Output location name(输出位置名称)	输出文件的位置。	'bin'
As JRE library use(使用 JRE 链接库时)	指定所要使用的 JRE 链接库。 JRE 储存器 JRE 储存器。 JRE_LIB 变数 JRE_LIB 变数指定的 JRE。	JRE 储存器

3.6.10 组织汇入(Organize imports)

下列的喜好设定是定义「组织汇入」指令要如何在编译单元中产生 import 陈述式。

「组织汇入」喜好设定

选项	说明	默认值
Import order list(汇入顺序清	这份前缀清单显示套件汇入到 Java 编译单元中的顺序。每一个项目皆定义出一个区块。区块之	java javax

选项	说明	默认值
单)	间各空一行。	org com
New...(新建...)	新增套件名称前缀到汇入顺序清单中。在产生的对话框中，输入套件名称或套件名称前缀。	n/a
Edit...(编辑...)	变更现有套件名称前缀的名称。在产生的对话框中，输入套件名称或套件名称前缀。	n/a
Up(上)	在汇入顺序清单中，将所选套件名称前缀往上移。	n/a
Down(下)	在汇入顺序清单中，将所选套件名称前缀往下移。	n/a
Remove(移除)	将套件名称前缀从汇入顺序清单中移除。	n/a
Load...(载入...)	从档案加载套件名称前缀清单。	n/a
Save...(储存...)	将套件名称前缀清单储存至档案。	n/a
Number of imports needed before . * is used(使用.*之前所需的汇入数目)	在使用 <code><package>.*</code> 之前，可来自相同套件的完整 import 陈述式的数目。	99
Do not create imports for types starting with a lower case letter(不为以小写字母开头的类型建立汇入)	当启用时，则不会汇入以小写字母开头的类型。	开启

3.6.11 「重构」喜好设定(Refactoring preferences)

可在「重构」喜好设定页面中设定下列的喜好设定。(「Window」
「Preferences」
「Java」
「Refactoring」。)

选项	说明	默认值
Confirm the	在这个区段中，请选取哪些种类的问题会使精灵	错误

选项	说明	默认值
execution of the refactoring if (如果发生下列情况，请确认执行重构)	<p>在按下 Finish 之后，仍维持开启状态并显示问题：</p> <ul style="list-style-type: none"> ■ 会阻止实际重构执行的问题 ■ 搜寻工作台 ■ 工作台中的警告 ■ 前置条件检查所产生的信息 <p>当问题的严重性比所选的层次还低时，则可让进行重构，而不必先预览结果。</p>	
Save all modified resources automatically prior to refactoring (在重构前自动储存所有修改过的资源)	如果启用这个选项，则每当执行重构动作时，工作台会自动储存自前次储存以来所有已修改过的资源。	不勾选

3.6.12 作业标示(Task Tags)

在这个喜好设定页面上，可以配置作业标示。当标示清单不是空的时候，每当编译器在 Java 程序代码中的任何批注内遇到其中一个对应标示时，编译器将发出作业标记。所产生的作业讯息将包括标示，以及直到下一个字行分隔字符或批注结尾的范围。

指令	说明
New... (新建...)	新增作业标示。请在出现的对话框中，指定新作业标示的名称和优先级。
Remove (移除)	移除所选作业标示。
Edit... (编辑...)	可以编辑所选作业标示。请在出现的对话框中，编辑作业标示的名称和/或优先级。

有一个名为 TODO、优先级为 Normal 的预设作业标示。

3.7 团队(Team)

团队喜好设定页面包含会影响版本管理团队支持的选项。

选项	说明	默认值
Use compressed folders as default Synchronize view layout(利用压缩数据夹作为预设的「同步化」视图布置)	在初次将同步化加入「同步化」视图时，请利用这个选项来配置所用的预设布置。可以在视图下拉菜单中后续变更这个布置。	已启用
Show all synchronization information in a resource's text label(将所有同步化信息显示在资源的文字卷标中)	请利用这个选项，将资源的同步化状态显示成资源卷标中的文字。依预设，只会用一个图示装饰元来识别资源的同步化状态。	已停用
Switch to the associated perspective when a synchronize operation completes(当同步化作业完成时，切换到相关的视景)	请利用这个选项来配置执行同步化作业时会发生的状况。选项如下： <ul style="list-style-type: none"> ■ Always(固定)：一律切换视景 ■ Never(绝不)：绝不切换视景 ■ Prompt(提示)：提示切换视景 	提示

选项	说明	默认值
Perspective Switching(视景切换)	请利用这个选项来配置执行同步化作业时所显示的视景。	「团队同步化」视景

3.7.1 CVS

在 CVS 喜好设定页面中，可以自订 CVS 外挂程序的若干项目。

一般 CVS 喜好设定(General CVS Preferences)：

选项	说明	默认值
Prune empty directories(删改空目录)	可以使用这个选项来指定在更新以及同步处理视图中删改空白的目录。虽然删改的目录不会显示在工作台中，在储存库中仍会存有一个空目录。这是有帮助的，因为 CVS 不会让客户端从服务器移除目录。	已启用
Consider file contents in comparison(在比较时考虑档案内容)	当比较 CVS 资源时，请利用这个选项来比较所找到已变更的档案之内容。通常会利用时间戳记来比较 CVS 档案，这是目前最快的方法。不过，在某些情况下，经由比较档案内容可得到较为精确的比较。停用这个选项会加快比较的速度，但可能会产生内容相同的比较项目。这个选项只适用于比较，不适用于合并和工作区同步化。	已启用
Delete unmanaged resources on replace(取代时删除未管理的资源)	可以使用这个选项，在取代为储存库的资源时，允许删除不在 CVS 控制下的资源。	已启用
Treat all new files as binary(将所有新	可以使用这个选项来置换档案内容设定以及将所有新档案视为二进制文件。	已停用

选项	说明	默认值
档案视为二进制文件)		
Validate server version compatibility on first connection(第一次联机时验证服务器版本的兼容性)	可以使用这个选项，在第一次联机时查询 CVS 服务器版本，以判断服务器的兼容性。服务器版本会输出到主控台，如果侦测到不兼容，就会在联机时记载警告讯息。	已启用
Confirm move tag on tag operation(确认在标示作业上移动标示)	请利用这个选项，以便在选取「移动标示」选项时得到提示。	已启用
Display detailed protocol output to stdout(将详细的通讯协议输出显示在标准输出中)	请利用这个选项来显示工作台和 CVS 服务器之间的通讯追踪。	已停用
Convert text files to use platform line neding(转换文字文件来使用平台行尾)	请利用这个选项，将文字文件的行尾转换成平台所用的行尾。如果将资源移出到 Windows 机器所装载的 *nix 磁盘驱动器，可以停用这个选项。	已启用
Show revision comparisons in dialog(在对话框中显示修订比较)	请利用这个选项，将修订比较显示在对话框中，而不是显示在比较编辑器中。	已停用

选项	说明	默认值
Communication timeout(通讯超时)	可以使用这个选项来配置在逾时前要等待联机到 CVS 服务器的时间总数 (以秒为单位)。	60 秒
Quietness level(安静层次)	设定针对指令的 CVS 打印状态信息总数。在有点安静(Somewhat quiet)模式中, 会抑止打印不重要的参考信息。重要性的考虑是根据每个指令。在非常安静(Very quiet)模式中, 除了完成指令的必要输出外, 所有的输出都会被抑止。在非常安静(Very quiet)模式中, 有些 CVS 服务器可能无法告知一些已经发生的错误的重要信息。可能要考虑改用有点安静(Somewhat quiet)模式。	细节
Default keyword substitution(预设的关键词替代)	可以使用这个选项来设定文字文件的预设关键词替代。	包含关键词展开项 -kkv 的 ASCII
Save dirty editors before CVS operations(在 CVS 作业之前储存变动过的编辑器)	可以使用这个选项来配置在执行 CVS 作业时, 如果已开启的编辑器包含未储存的变更时, 会发生什么情况。 选项包括: <ul style="list-style-type: none"> ■ Never(绝不): 继续执行 CVS 作业, 即使已开启的编辑器中有尚未储存的变更。 ■ Prompt(提示): 询问使用者要如何处理已开启的编辑器中的未储存变更。 ■ Auto-save(自动储存): 在每一个 CVS 作业前自动储存已开启编辑器中尚未储存的变更。 	提示

主控台喜好设定(Console preferences):

选项	说明	默认值
Console text color settings(主控台文字的颜色设定)	可以使用这些选项来变更「CVS 主控台」中所显示的文字的颜色。 <ul style="list-style-type: none"> ■ 指令行文字 (黑色) ■ 消息正文 (蓝色) ■ 错误文字 (红色) 	

选项	说明	默认值
Show CVS output in Console view(在「主控台」视图中显示 CVS 输出)	请利用这个选项，将 CVS 指令输出显示在「主控台」视图中。启用这个选项可以显示非常有用的信息，但指令作业速度会变慢。	已停用

「Ext 联机方法」喜好设定(Ext Connection Method preferences)：

Use external program vs. Use internal connection method(使用外部程序和使用内部联机方法)	这个页面可以配置 ext 联机方法来使用外部程序，或配置另一个联机方法来连接到服务器。后面一个选项是要让 extssh 之类的自订联机方法保持与外部 CVS 客户端工具兼容。	使用外部程序
CVS_RSH(CVS_RSH)	可以使用这个选项来配置在在联机到远程 CVS 服务器时要呼叫的程序。呼叫 RSH 指令时会使用下列呼叫型样： <i>CVS_RSH 参数 CVS_SERVER</i>	ssh
Parameters(参数)	可以使用这个选项来配置传送到 CVS_RSH 程序的参数。预设参数型样为 {host} -l {user}。可使用 {host}、{user}、{password} 与 {port} 等变量来加以修改。	{host} -l {user}
CVS_SERVER(CVS_SERVER)	可以使用这个选项来配置要执行的远程 CVS 服务器程序的名称。只有当远程 CVS 服务器二进制名称与默认值不同时，才变更这个设定	cvs
Connection type(联机类型)	如果启用使用另一个联机方法的选项，请利用这个选项来设定使用 ext 联机方法的储存库位置所用的联机方法。	

「标签装饰」喜好设定(Label Decorations preferences)：

Text(文字)	可以使用这个页面中的选项来配置如何将 CVS 信息新增至文字卷标。
Icons(图)	可以使用这个页面中的选项来配置可用哪些图标作为覆盖，以便在

示)	视图中显示 CVS 特定的信息。
General (一般)	<p>可以使用这个页面中的选项来配置有关装饰元的几个喜好设定：计算数据夹的深度送出状态(Compute deep outgoing state for folders)</p> <p>可以使用这个选项来配置是否应该计算数据夹的送出指示器。停用这个选项时，可增进 装饰元的效能，因为计算数据夹的已用过状态时，需要计算所有子项资源的已用过状态。(预设是启用的)</p>

密码管理(Password Management)：

这个喜好设定页面可以查看哪些储存库位置的密码快取在金钥环档案中，且可以除去这些密码。

SSH2 联机方法(SSH2 Connection Method)：

General (一般)	请利用这个页面中的选项来配置 ssh 金钥目录的位置，以及联机时要将哪些金钥传给服务器。
Proxy	请利用这个页面中的选项来配置 HTTP 或 SOCKS5 Proxy。
Key Management (金钥管理)	请利用这个页面中的选项来建立、管理和汇出金钥

监视/编辑喜好设定(Watch/Edit preferences)：

Configure projects to use Watch/edit on checkout (配置项目以便在移出时使用监视/编辑)	可以使用这个选项来指出从储存库移出的档案都要变成只读。	停用
When read-only files are modified in an editor (在编辑器中修改)	<p>可以使用这个选项来配置当另一个工具或已开启的编辑器在修改只读档时，会发生何种状况。</p> <p>■ 传送 cvs 编辑通知给服务器(Send a cvs edit notification to the server)：在容许写入档案前发出 cvs 编辑通知给服务器。如果档案中有其它编辑器，就会提示使用者是否要继续</p>	传送 cvs 编辑通知给服务器

只读文件的时机)	<p>续或取消。</p> <ul style="list-style-type: none"> ■ 编辑档案而不通知服务器(Edit the file without informing the server)：使档案变成只读而不通知服务器。 	
Before a CVS edit notification is sent to the server(在传送 CVS 编辑通知给服务器之前)	<p>请利用这个选项来配置当开启的编辑器或另一个工具在修改只读档，且启用了将 CVS 编辑通知传给服务器(Send a cvs edit notification to the server)时，会发生何种状况。</p> <ul style="list-style-type: none"> ■ Always Prompt(固定提示)：固定提示使用者进行确认 ■ Only prompt if there are other editors(只有在有其它编辑器时才提示)：向使用者显示现行编辑器的清单，让使用者确认或取消编辑。 ■ Never Prompt(绝不提示)：传送编辑通知，但不提示 	只有在有编辑器时才提示

3.7.2 忽略的资源(Ignored Resources)

在「Team」「Ignored Resources」喜好设定页面中，可以指定要排除在版本控制管理系统之外的文件名称型样。它有一份档案型样清单，资源必须符合这些档案型样，才能成为版本控制候选项。这些型样可含有万用字符 "*" 和 "?"。型样 "*" 代表任何零或多个字符的序列。型样 "?" 代表任何一个字符。例如，可以指定型样 "*~"，其将会比对任何以 "~" 结尾的暂存档。在更新或确认作业期间，将会忽略任何符合任一型样的档案或目录。

如果要将档案类型新增至忽略清单，只需要按一下 Add 按钮。在对话框中，请输入档案类型（例如 *.class）。如果要从忽略清单中移除档案类型，只需要选取忽略清单中的档案类型，按一下 Remove 按钮。

可以从清单中取消选取某个档案型样，暂时停用这个档案型样的忽略功能。不必从清单中移除指定的档案型样，就可以暂时停用它。

3.7.3 档案内容(File Content)

在「Team」「File」喜好设定页面中，可以将扩展名关联到档案所包含的数据类型。档案内容类型的两个选项是 ASCII 和二进制。然后，像 CVS 这类的储存库提供者就可以使用这项信息来为内容类型提供适当的行为。例如，在 ASCII 档案中，CVS 可确保行终止器符合 OS 平台的行终止器。

将项目新增至「档案内容」页面的方法有两种。第一个方法是透过工作台外挂程序的帮助。整合到工作台的工具可以为工作台提供工具专用的扩展名的档案内容类型。工作台本身也会定义在工作台中经常使用和出现的扩展名的档案内容类型（例如，html、gif 等等）。

第二个方法是让使用者在「档案内容」喜好设定页面中明确地新增档案内容类型。如果要这么做，使用者只需要按一下 Add 按钮，然后输入扩展名。之后，他们可以切换类型与扩展名之间的关联，方法是在表格中选取扩展名的项目，然后按一下 Change。使用者可以选取项目，然后按一下 Remove，将项目从清单中移除。

4. Java 程序开发

在 Eclipse 中做任何事之前，都必须新增一个项目。Eclipse 可透过外挂支持数种项目(如 EJB 或 C/C++)，预设支持下列三种项目：

- ◆ Java Project – Java 开发环境
- ◆ Plug-in Project – 自行开发 plug-in 的环境
- ◆ Sample Project – 提供操作文件的一般环境

如图 4.1

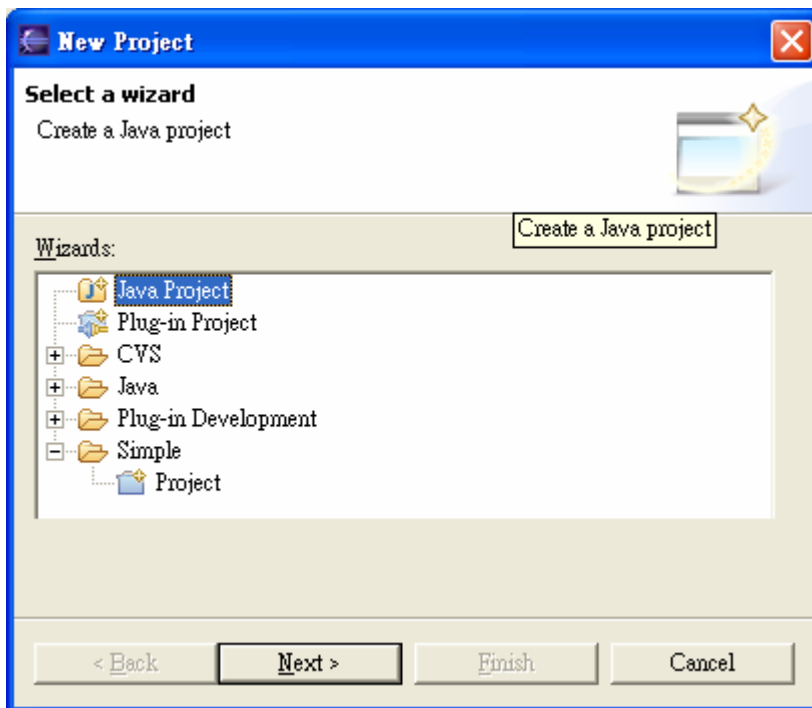


图 4.1

4.1 建立 Java 项目

新增 Java 项目的步骤：

1. 选择「File」 「New」 「Project」

(或是在『Package Explorer』窗口上按鼠标右键，选择「New」
「Project」选单选项)

(或是按工具列上 New Java Project 的按钮)

II. 在 New Project 对话框(图 4.1)，选 Java Project

(或是展开 Java 的数据夹，选 Java Project，如图 4.2)

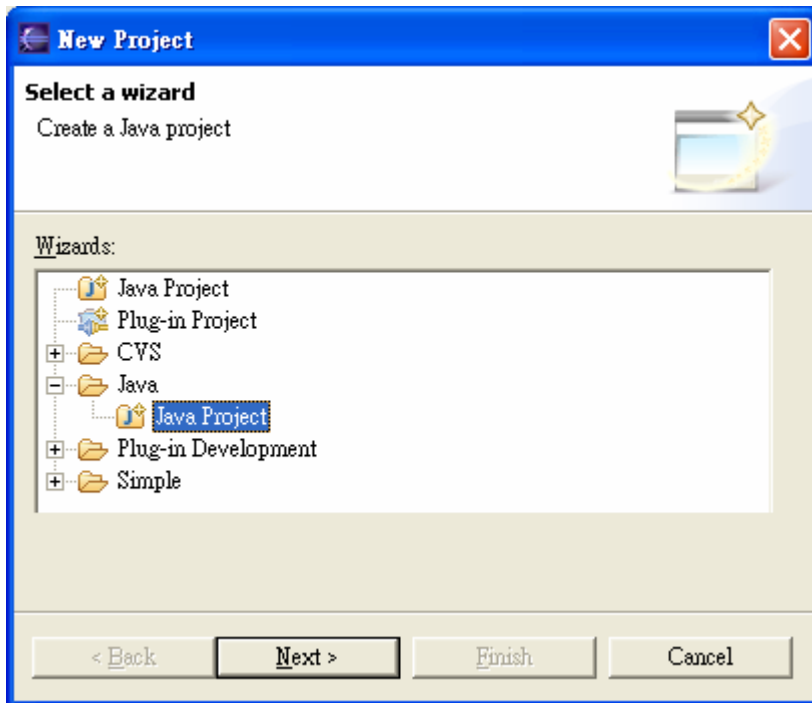


图 4.2

III. 在 New Java Project 的窗口中输入 Project 的名称，如图 4.3

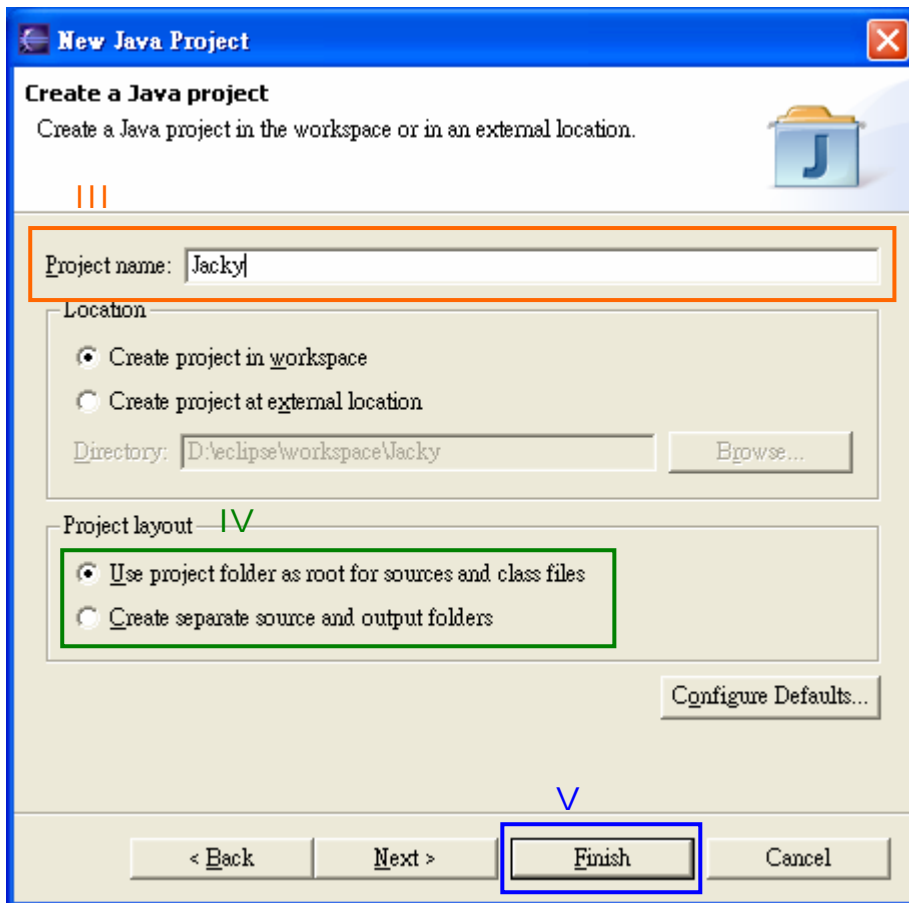



图 4.3

IV. 在 Project Layout 中可以选择编译好的档案是否要和原始档放在同一个目录下，如图 4.3

V. 按下 Finish

4.2 建立 Java 类别

新增 Java 类别的步骤：

- I. 选择「File」 「New」 「Class」
(或是在『Package Explorer』窗口上按鼠标右键，选择「New」 「Class」选单选项)
(或是按工具列上 New Java Class 的按钮)
- II. 在 New Java Class 窗口中，Source Folder 字段默认值是项目的数据夹，不需要更改。

III. Package 字段输入程序套件的名称

IV. Name 字段输入 Class Name

V. 在 Which method would you like to create 的部份 ,有勾选 public static void main(String[] args)的话 , 会 generate main method

VI. 按 Finish , 会依套件新增适当的目录结构及 Java 原始文件

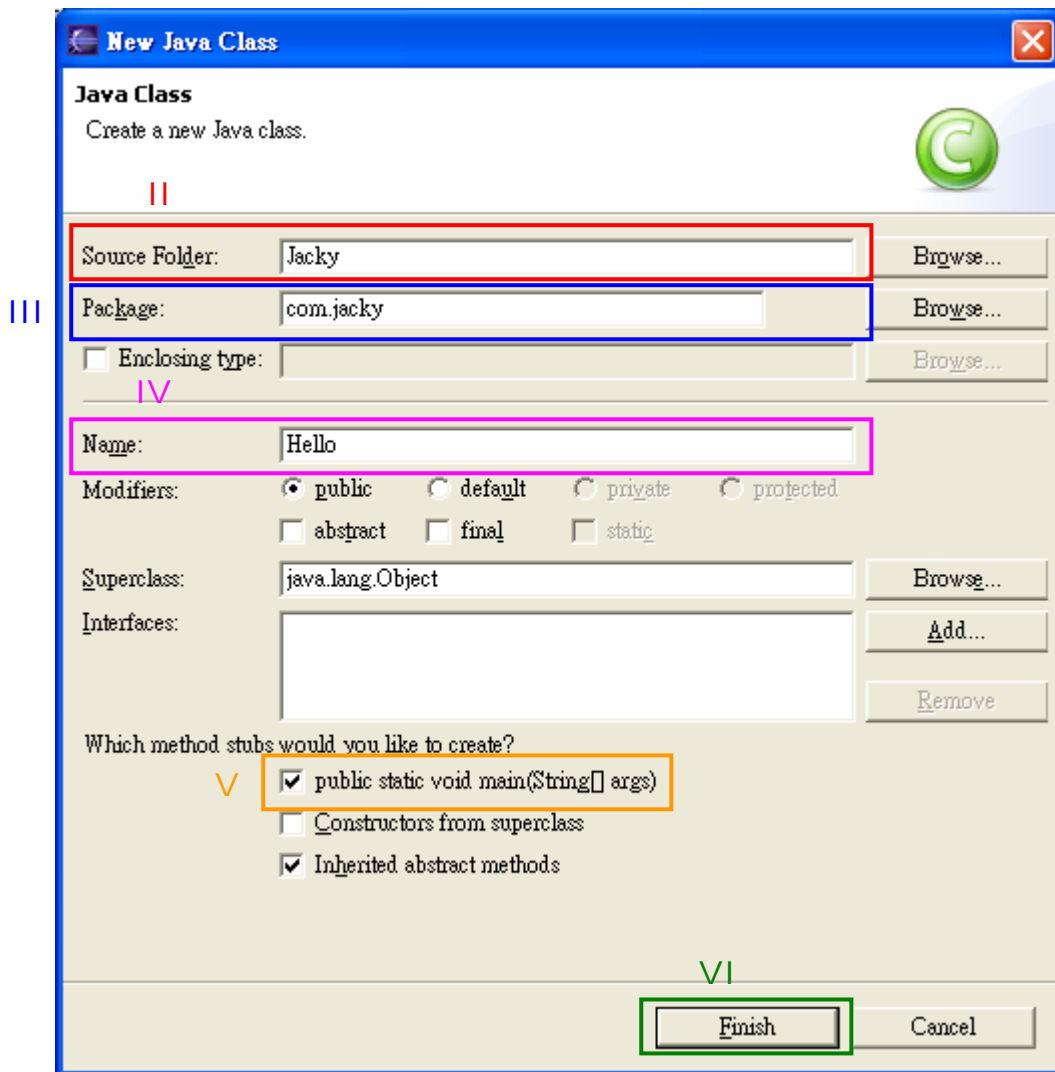


图 4.4

- 在 Package Explorer 的视图中可以看到程序的结构
- 在 Navigator 的视图中可以看到套件的目录架构

4.3 程序代码完成功能

4.3.1 Code Completion

在 Eclipse 中打左括号时会立刻加上又括号；打双引号(单引号)时也会立刻加上双引号(单引号)。

4.3.2 Code Assist

在输入程序代码时，例如要打 System.out.println 时，打完类别名称后暂停一会儿，Eclipse 会显示一串建议清单，列出此类别可用的方法和属性，并附上其 Javadoc 批注。可以直接滚动选出然后按 Enter。

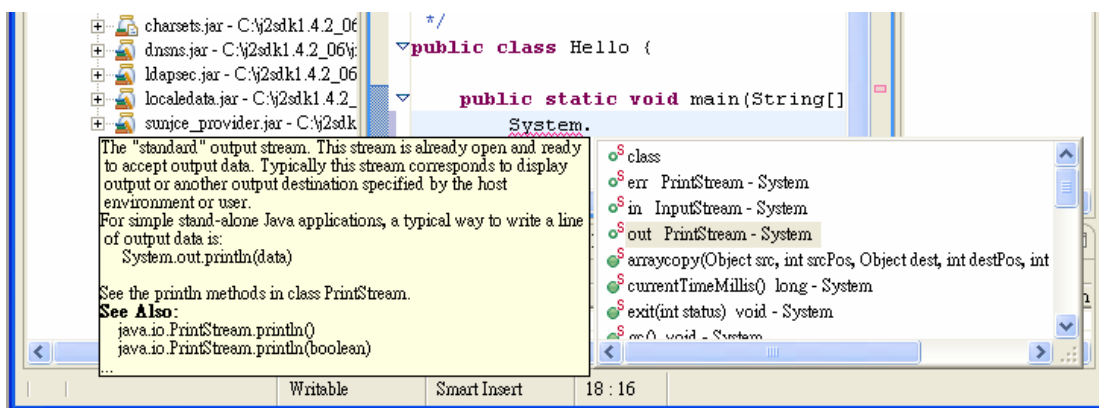


图 4.5

也可以只打类别开头的字母，然后按 Alt - /，一样会显示一串建议清单。

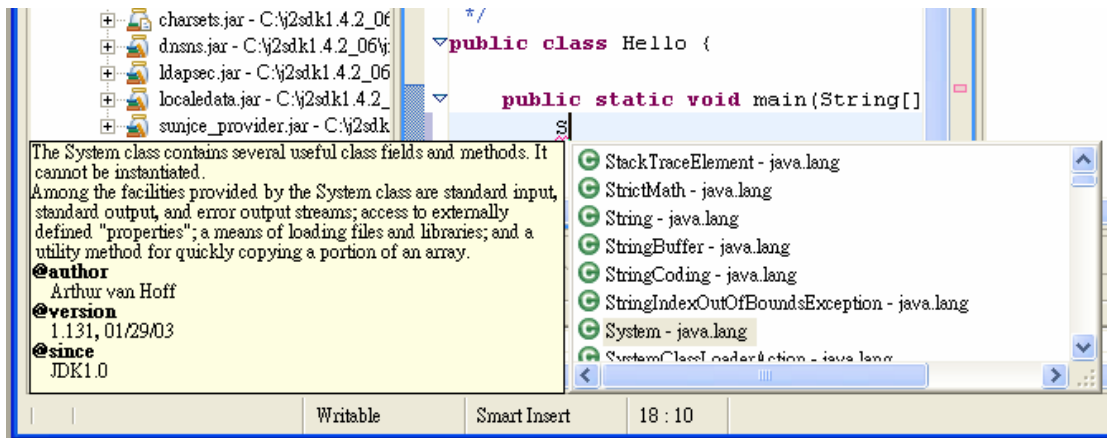


图 4.6

Alt - /这个组合键不仅可以显示类别的清单，还可以一并显示已建立的模板程序代码，例如要显示数组的信息，只要先打 for，在按 Alt - /这个组合键，就会显示模板的清单。

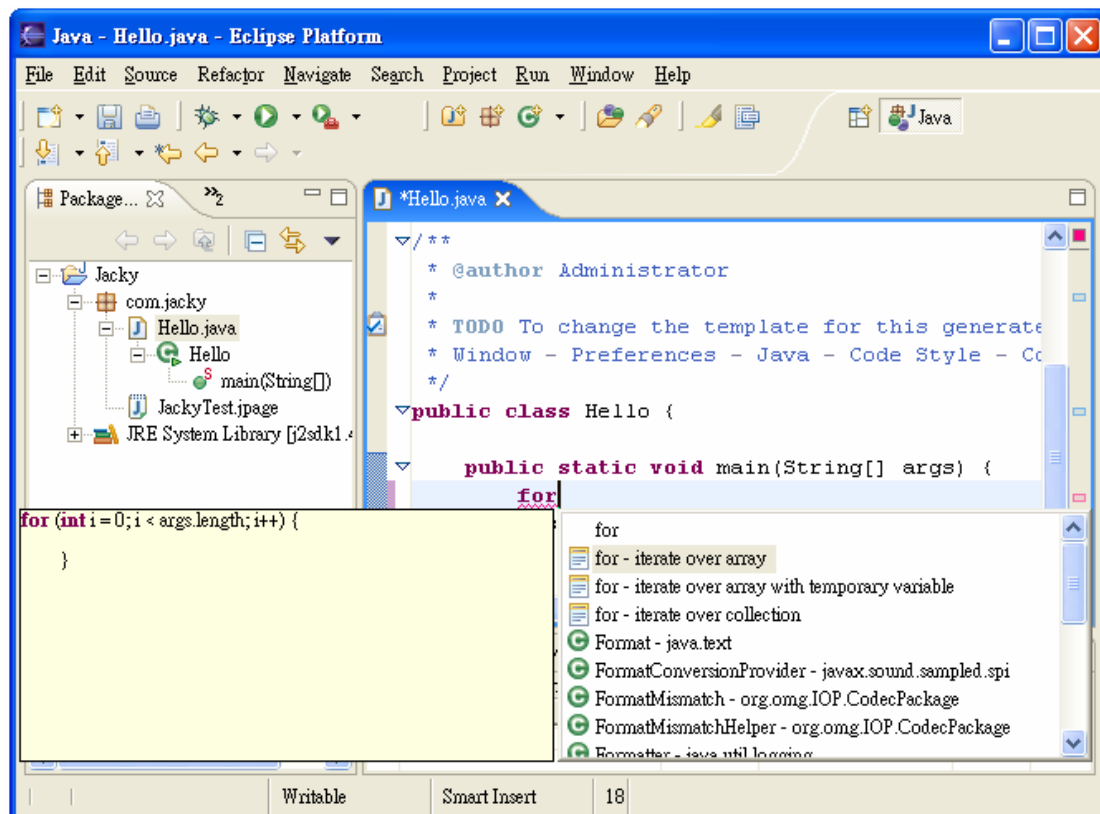


图 4.7

4.4 执行 Java 程序

大多数的程序不需特定的启动组态(Launch Configuration)，首先确定要执行的程序代码在编辑器中有选到(页签变蓝色)，再执行下列步骤：

- I. 选单选「Run」「Run as」「Java Application」
- II. 若有修改过程序，Eclipse 会询问在执行前是否要存档
- III. Tasks 试图会多出 Console 页签并显示程序输出

程序若要传参数、或是要使用其它的 Java Runtime...等等，则需要设定程序启动的相关选项，执行程序前，新增一个启动组态或选用现有的启动组态。

- I. 选单选「Run」「Run」，开启 Run 的设定窗口
 - Main 标签用以定义所要启动的类别。请在项目字段中，输入内含所要启动之类别的项目名称，并在主要类别字段中输入主要类别的完整名称。如果想要程序每当在除错模式中启动时，在 main 方法中停止，请勾选 Stop in main 勾选框。
 - 附注：不必指定一个项目，但这样做可以选择预设类别路径、来源查阅路径，以及 JRE。
 - 自变量(Arguments)标签用以定义要传递给应用程序与虚拟机（如果有的话）的自变量。也可以指定已启动应用程序要使用的工作目录。
 - JRE 卷标用以定义执行或除错应用程序时所用的 JRE。可以从已定义的 JRE 选取 JRE，或定义新的 JRE。
 - 类别路径(Classpath)卷标用以定义在执行或除错应用程序时所用类别文件的位置。依预设，使用者和 bootstrap 类别位置是从相关联项目的建置路径衍生而来。可以在这里置换这些设

定。

- **程序文件(Source)**卷标用以定义当除错 Java 应用程序时，用来显示程序文件之程序文件的位置。依预设，这些设定是从相关联项目的建置路径衍生而来。可以在这里置换这些设定。
- **环境(Environment)**标签会定义在执行 Java 应用程序或者对它进行除错时，所要使用的环境变量值。依预设，这个环境是继承自 Eclipse 执行时期。可以置换或附加至继承的环境。
- **共享(Common)**卷标定义有关启动配置的一般信息。可以选择将启动配置储存在特定档案，以及指定当启动配置启动时，哪些视景将变成作用中。

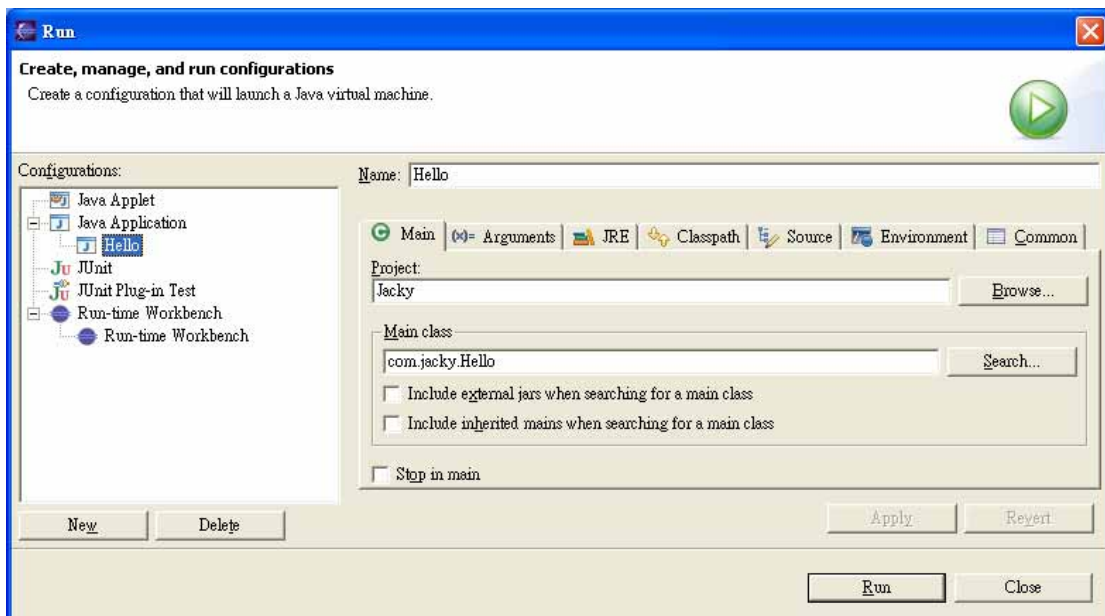


图 4.8

II. 在 Arguments 的页签中输入要传入的值，若是多值的话，用空格键隔开

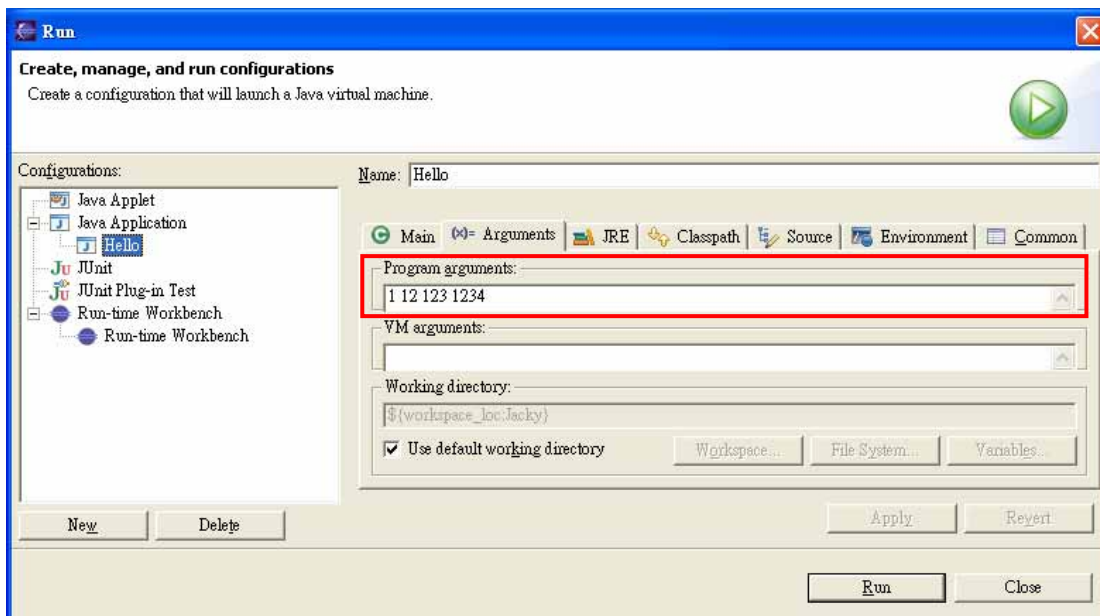


图 4.9

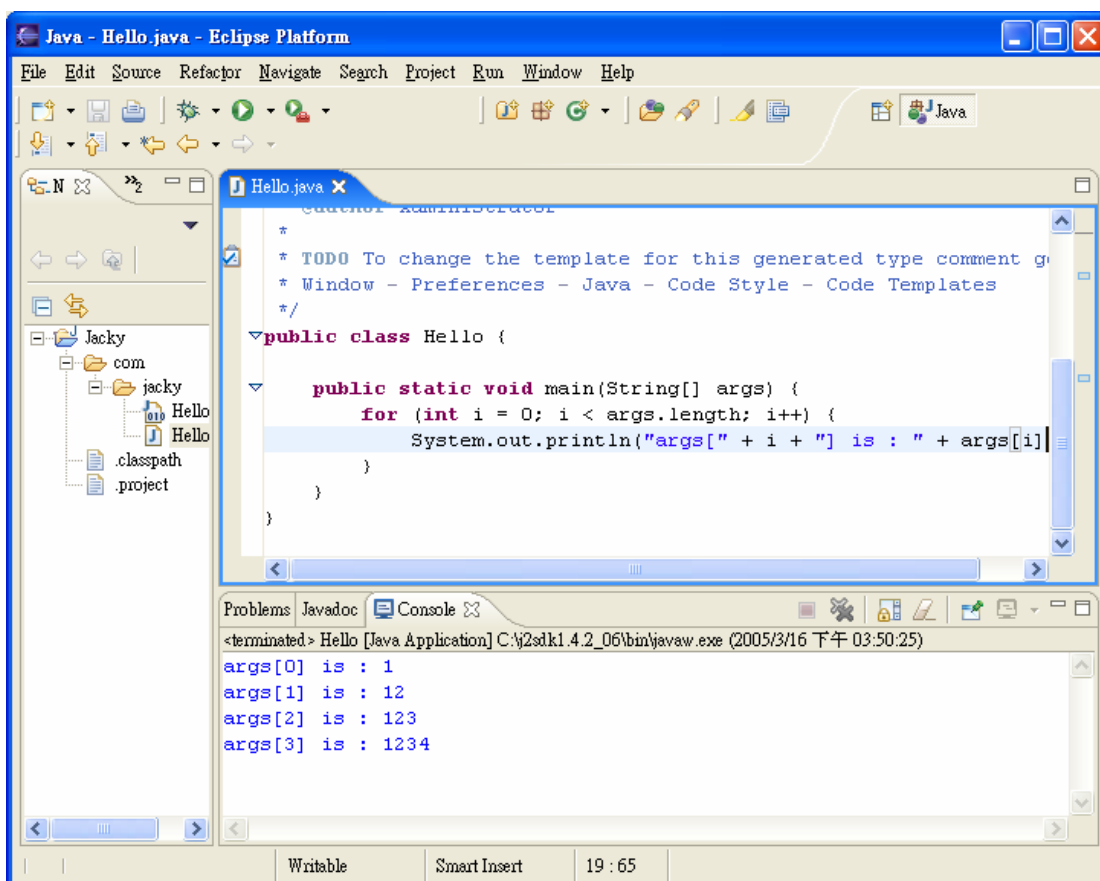


图 4.10

4.5 Java 实时运算簿页面(Java Scrapbook

Page)

写程序时可能会些其它的想法，但不知是否可行：多数情况是直接写到程序再来 debug，或是另外写各小程序。Eclipse 提供一种轻巧的替代方式，Java 实时运算簿页面(Java Scrapbook Page)，藉由渐进式编译器，可以在实时运算簿写入任意的 Java 程序代码并执行，不需另写在类别或方法中。

I. 切换至 Java 视景

II. 「File」 「New」 「Other...」 「Java」 「Scrapbook Page」 (在专案上按右键,「New」 「Other...」 「Java」 「Scrapbook Page」)

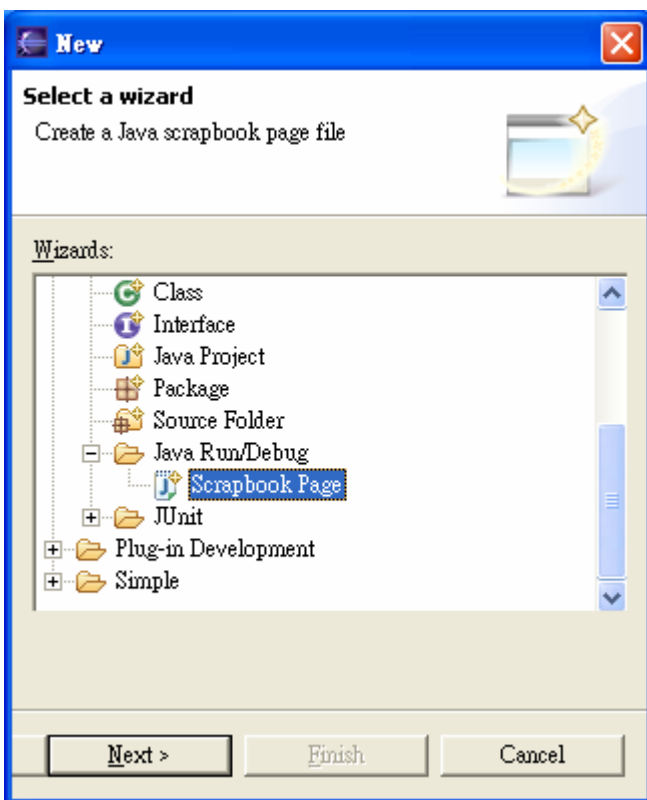


图 4.11

III. 选择要存放的地方

IV. 输入档名

IV. 按下 Finish

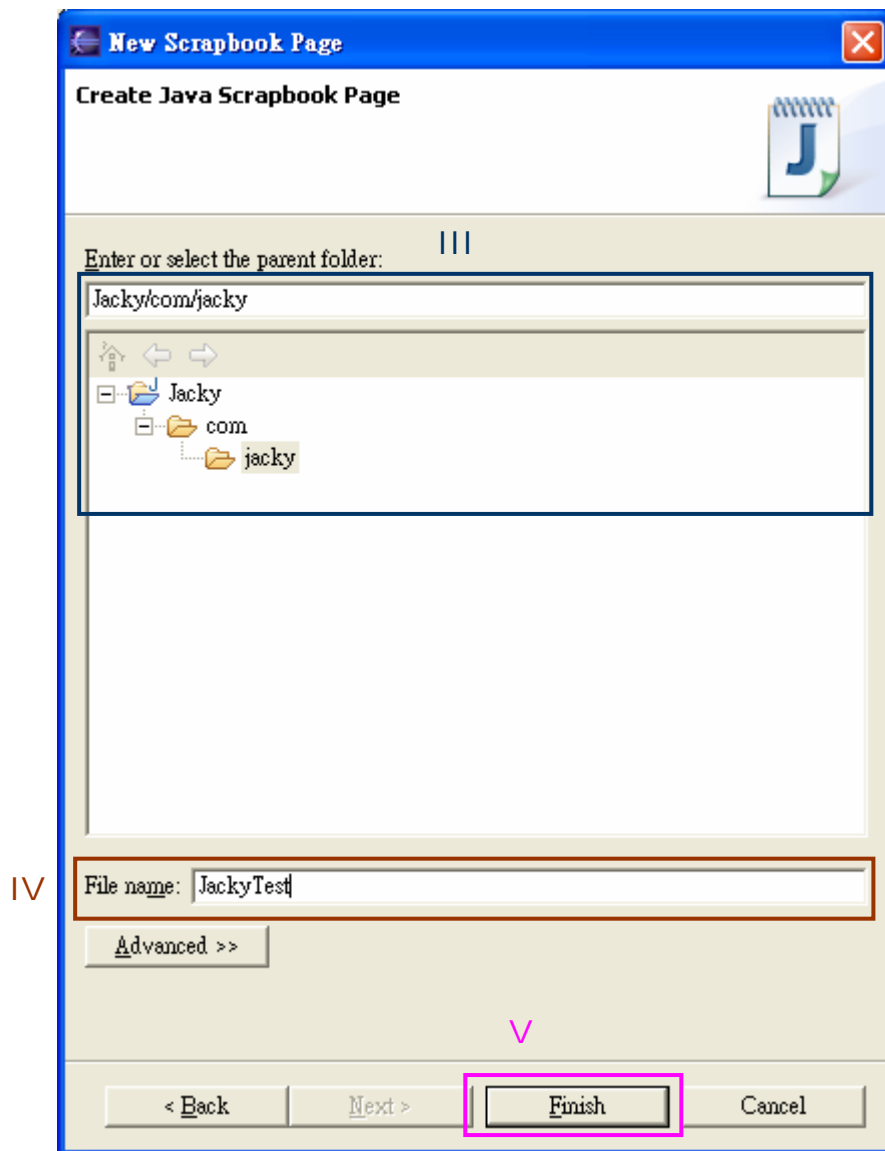


图 4.12

在「Package Explorer」或是「Navigator」视图会显示刚刚建立的 JackyTest.jpge 档案。

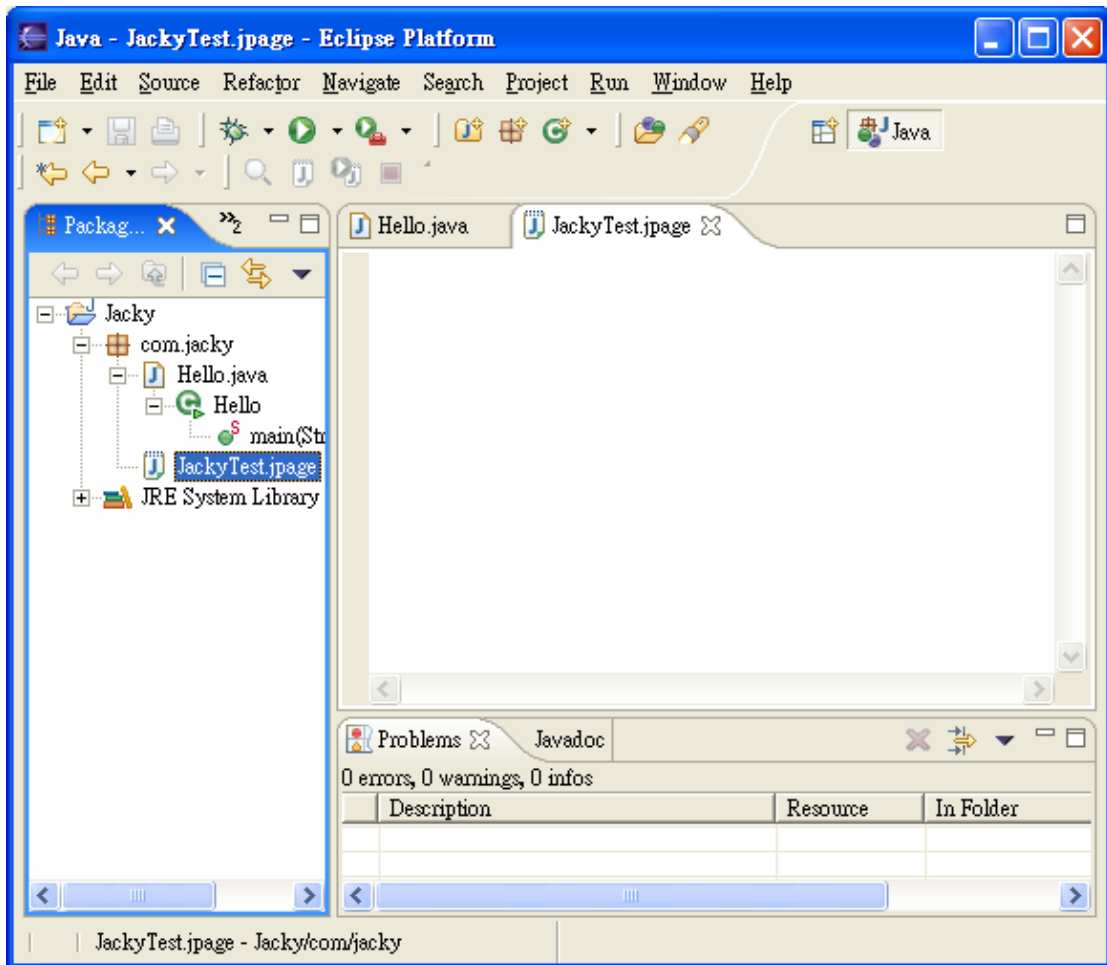


图 4.13

可以输入要测试的 Java 程序代码，例如

```
for (int i = 0; i < 5; i++) {
    System.out.println(Integer.toString(i));
}
```

- I. 将这段程序代码反白
- II. 在这段程序上按右键，选择 Execute
- III. Console 视图会显示执行的结果

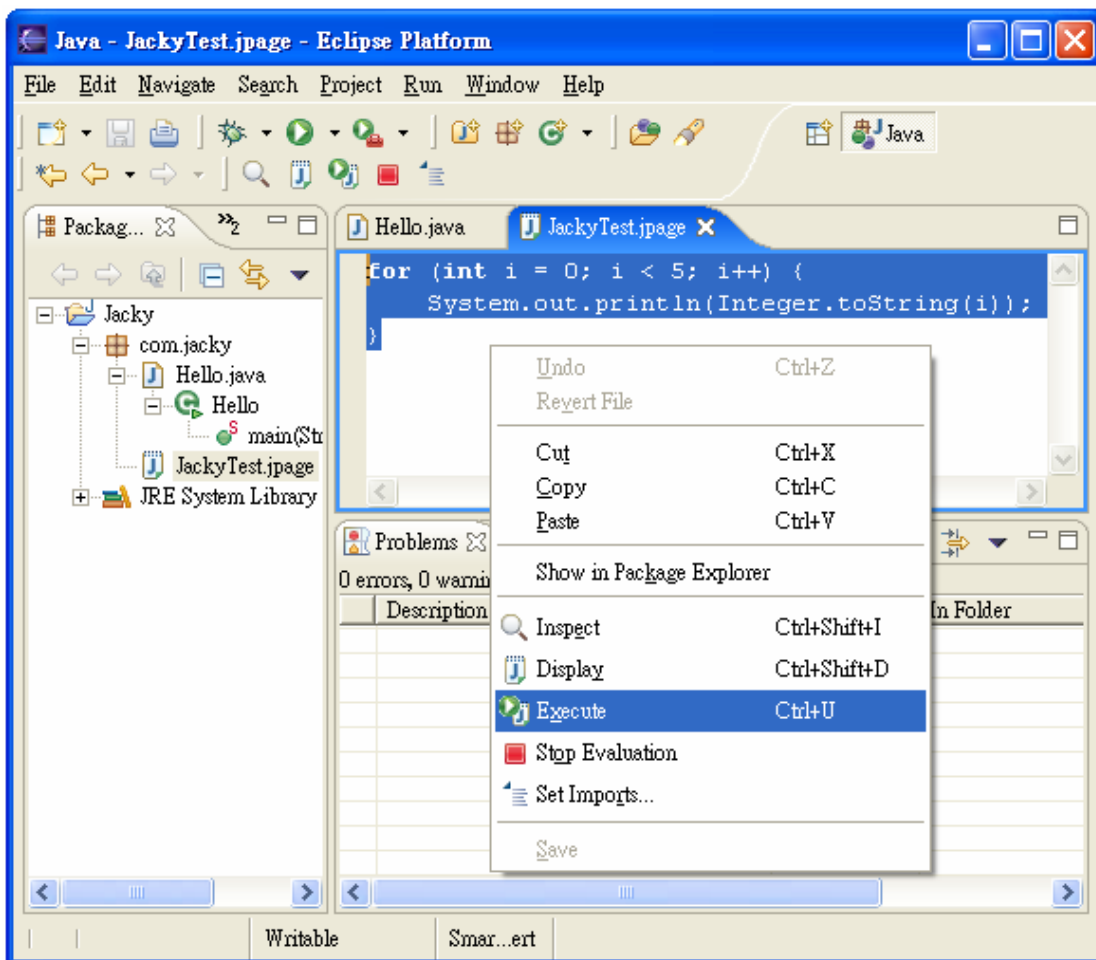


图 4.14

若需要汇入套件；

1. 在编辑器窗口内按右键，选 Set Import...

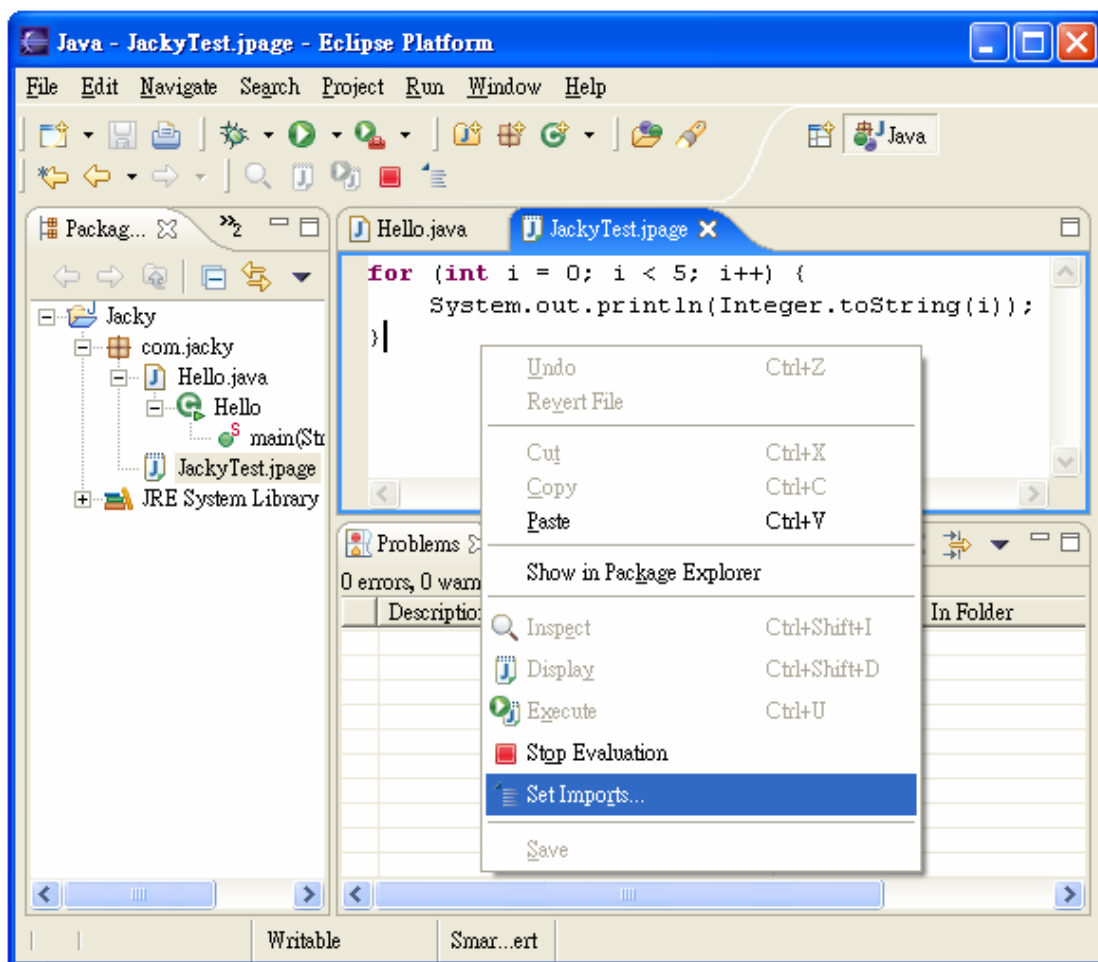


图 4.15

II. 在 Java Snippet Imports 窗口中，按 Add Packages 的按钮

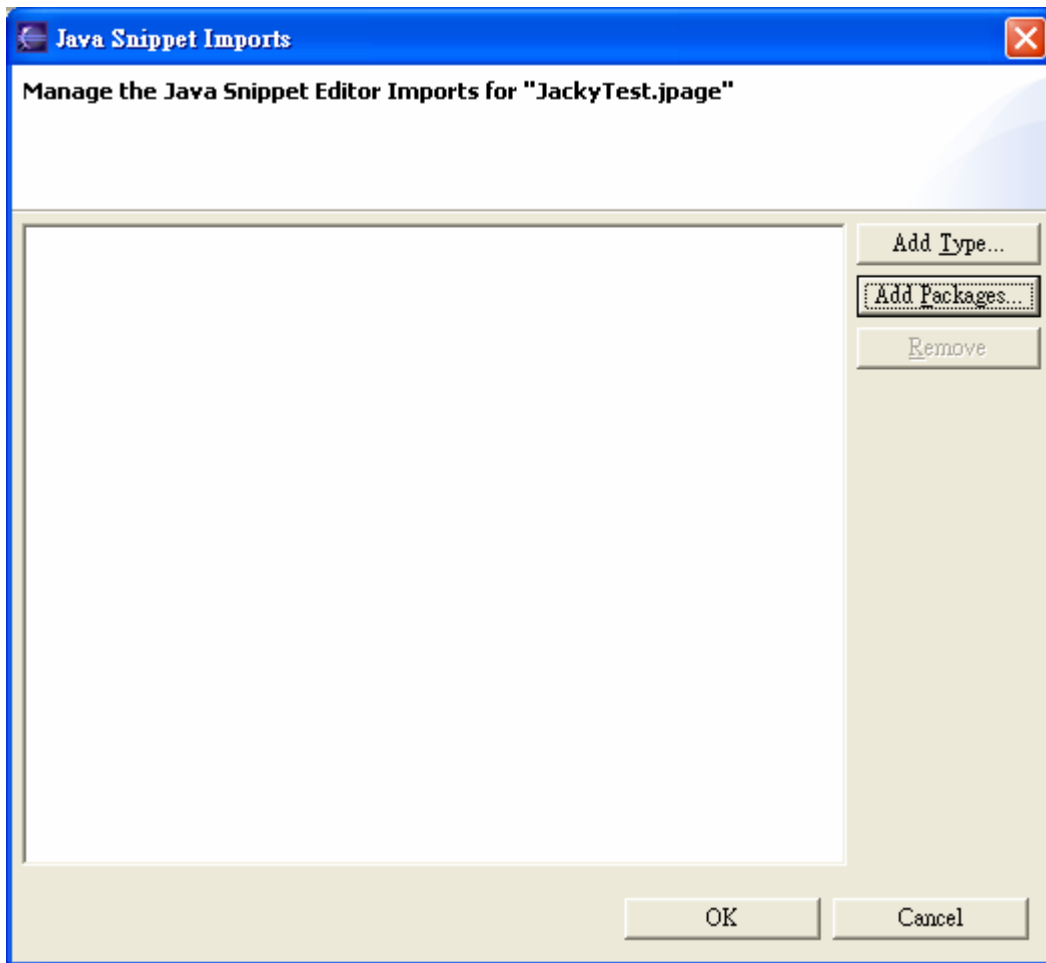


图 4.16

III. 在 Add Packages as Imports 的窗口中，挑选要 import 的 package

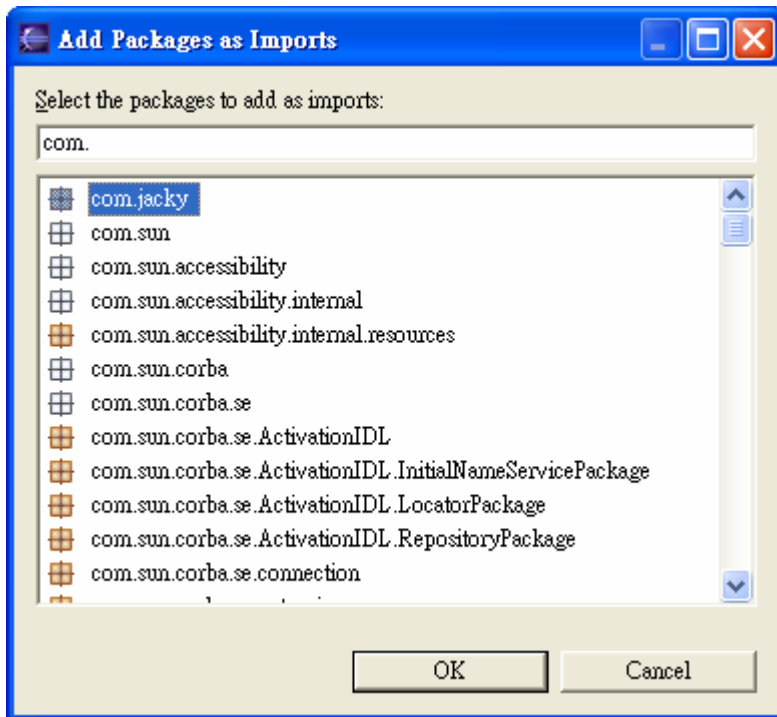


图 4.17

IV. 按 OK

4.6 自订开发环境

4.6.1 程序代码格式

在「Window」 「Preferences」 「Java」 「Code Style」
「Code Formatter」

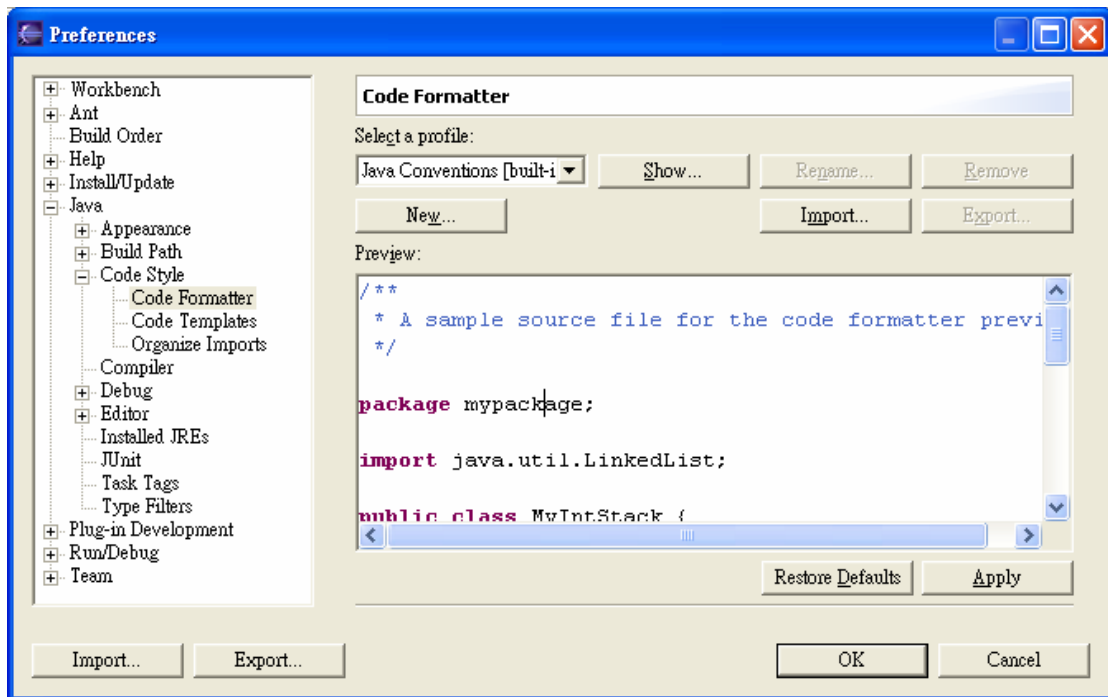


图 4.18

按 Show 的按钮，出现 Show Profile 的窗口，里面的各个页签，可以设定 Java Code Style

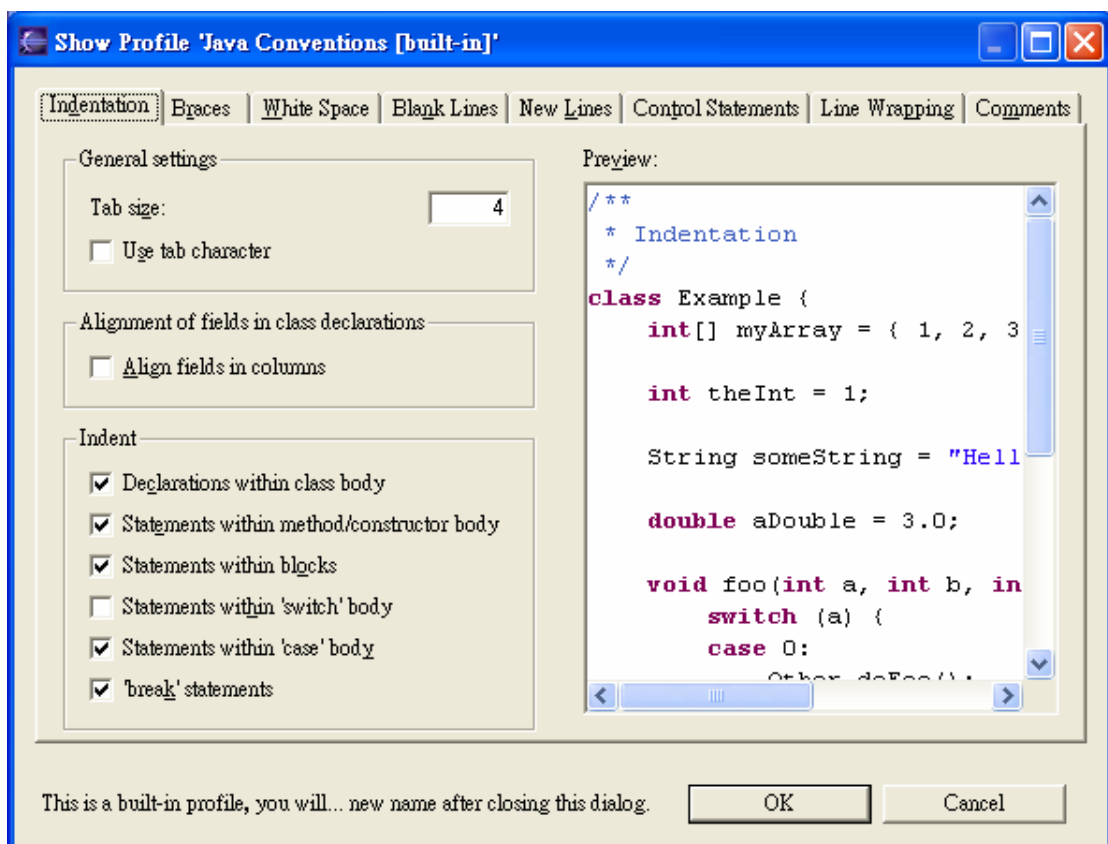


图 4.19

设定完成后，可以 Export 成一个档案；以利下次设定 Java Code Style 时，可以利用 Import 的方式，产生一致的程序风格。

4.6.2 程序代码产生模板

在开发 Java 时，可以把常用的流程控制建构式或是常用到的 API，建立一个模板，可以加速程序开发。接下来以 System.out.println() 为例子，来说明如何建立模板：

I. 「Window」 「Preferences」 「Java」 「Editor」 「Templates」

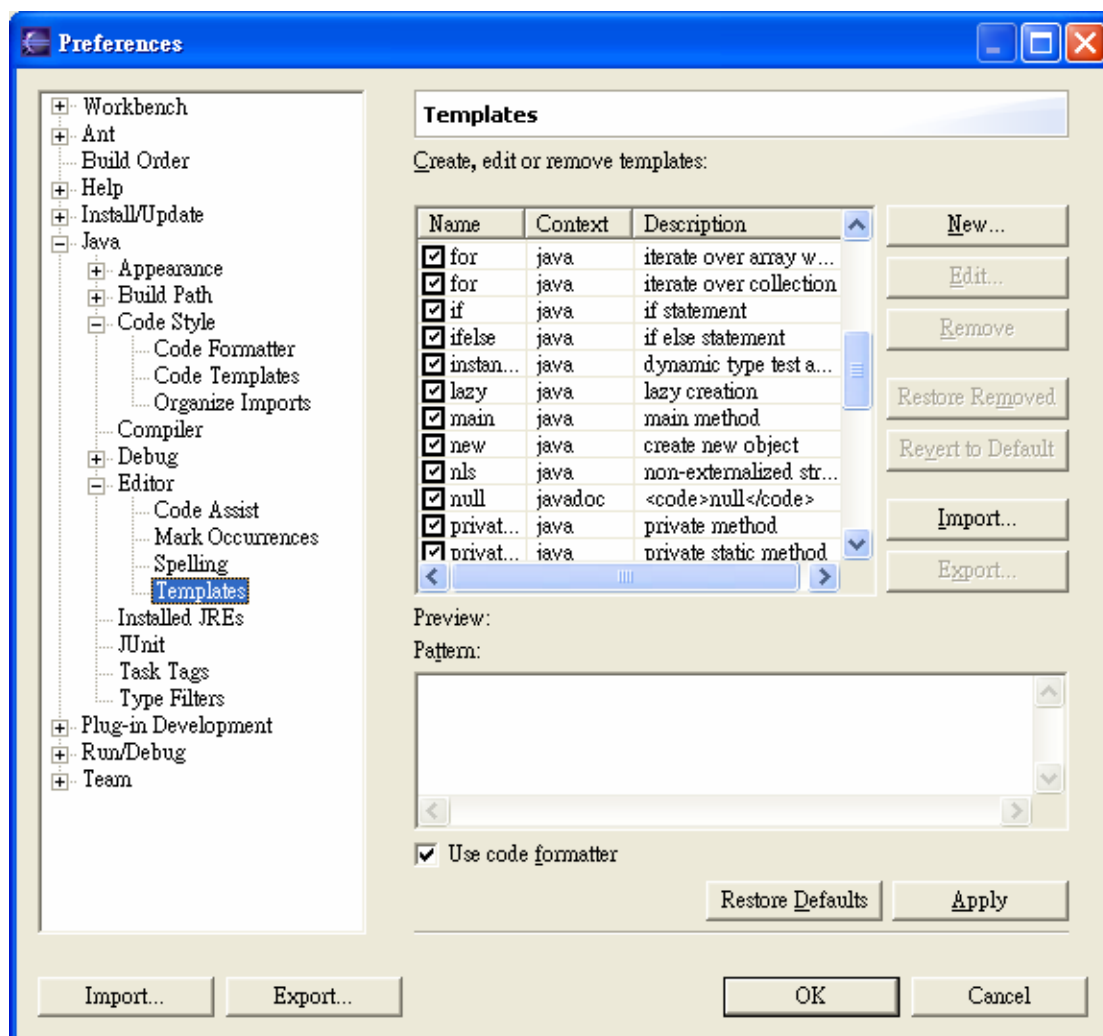


图 4.20

II. 在 Preferences 窗口按 New 的按钮

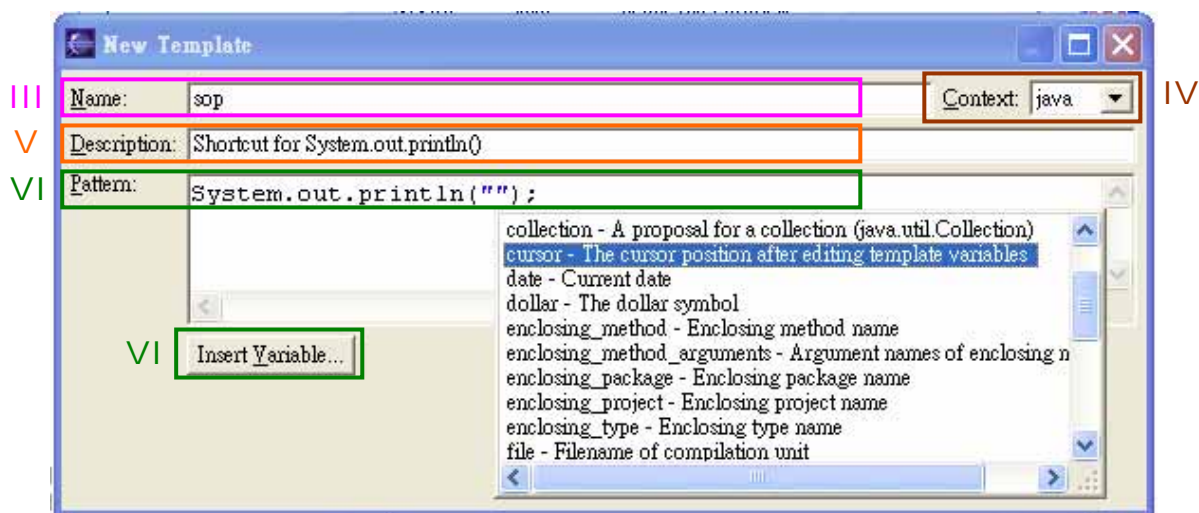


图 4.21

III. 在 Name 的字段输入自己想要的名称

IV. Context 选 java

V. 在 Description 的字段输入简短的说明

VI. 在 Pattern 的字段输入 System.out.println("")后；把光标移到两个双引号的中间，再按下面 Insert Variable 的按钮，选择 cursor

VII. 再按两次 OK

这里的\${cursor}变量代表插入模板的程序代码后，光标所在的位置。

使用此新模板，打 s(或是 sop)再按 Alt - /，从清单中选 sop，再按 Enter 即可。

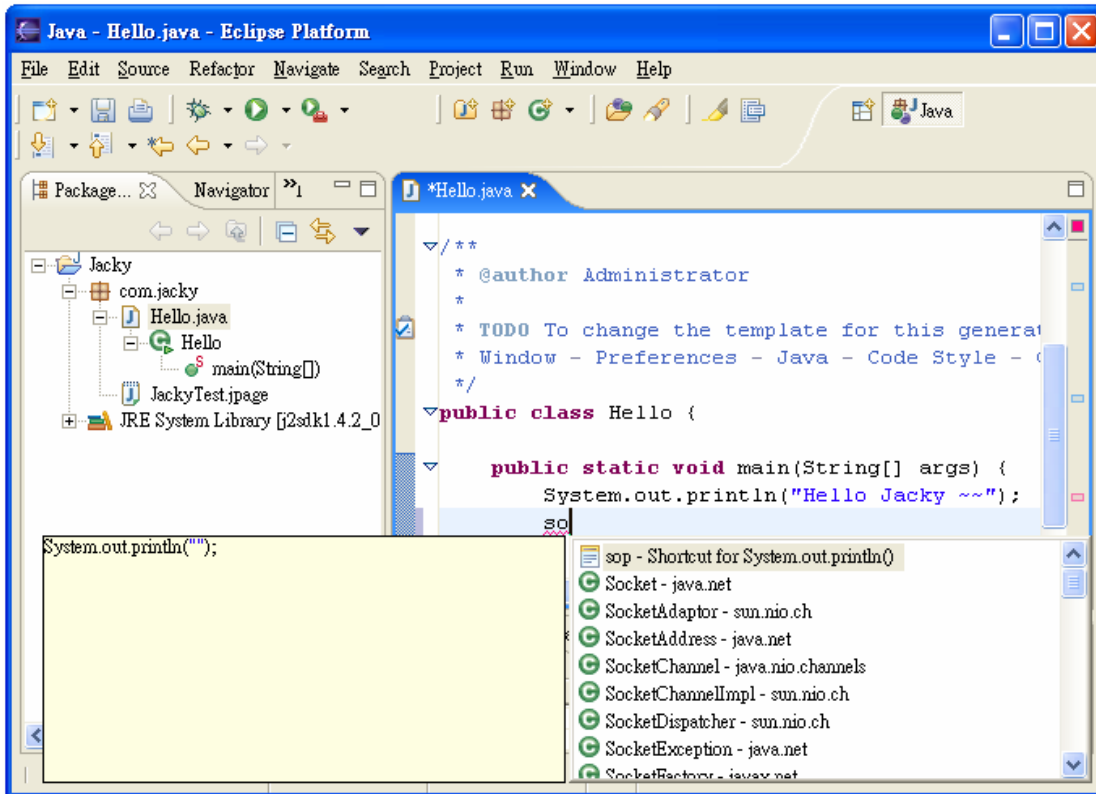


图 4.22

4.6.3 Javadoc 批注

编辑新增类别后出现的文字。移除 ” To change the template for this generated...” 这段前置文字，并自行扩充 Javadoc 批注。

- I. 「Window」 「Preferences」 「Java」 「Code Style」 「Code Templates」
- II. 选右边画面的「Code」 「New Java files」, 按 Edit 按钮

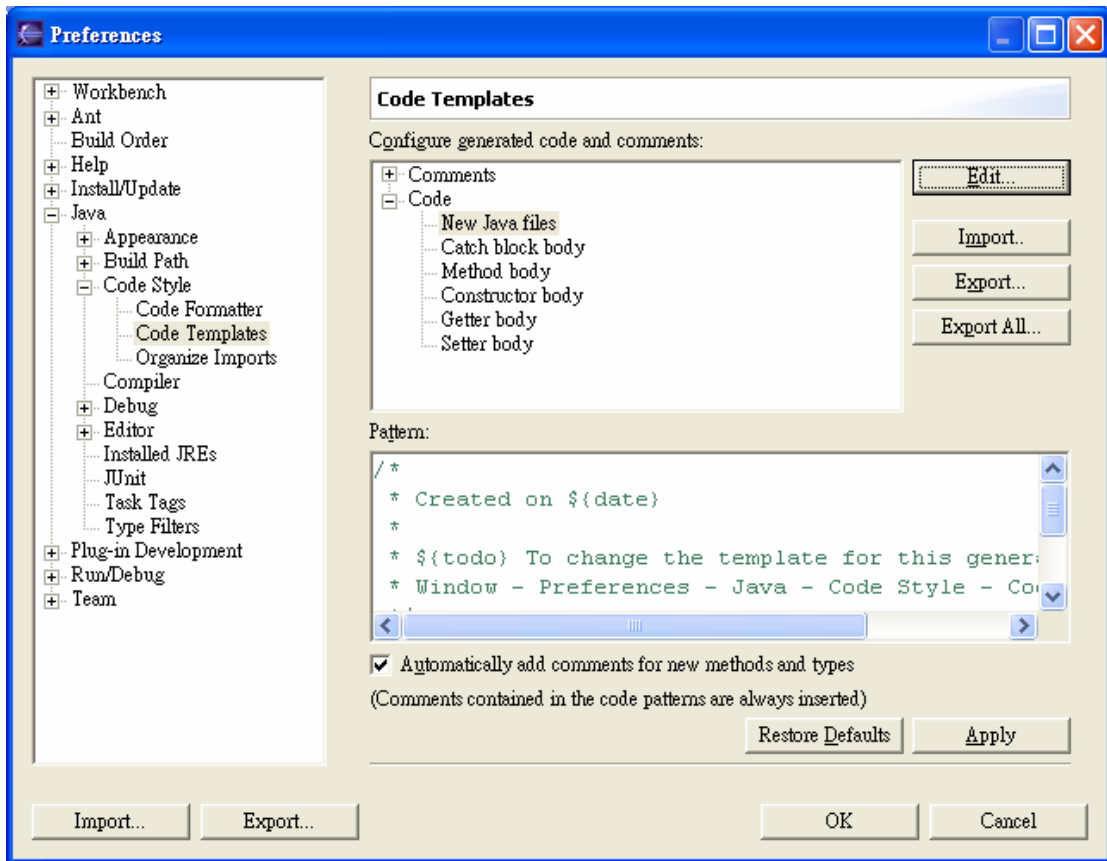


图 4.23

III. 修改成需要的格式

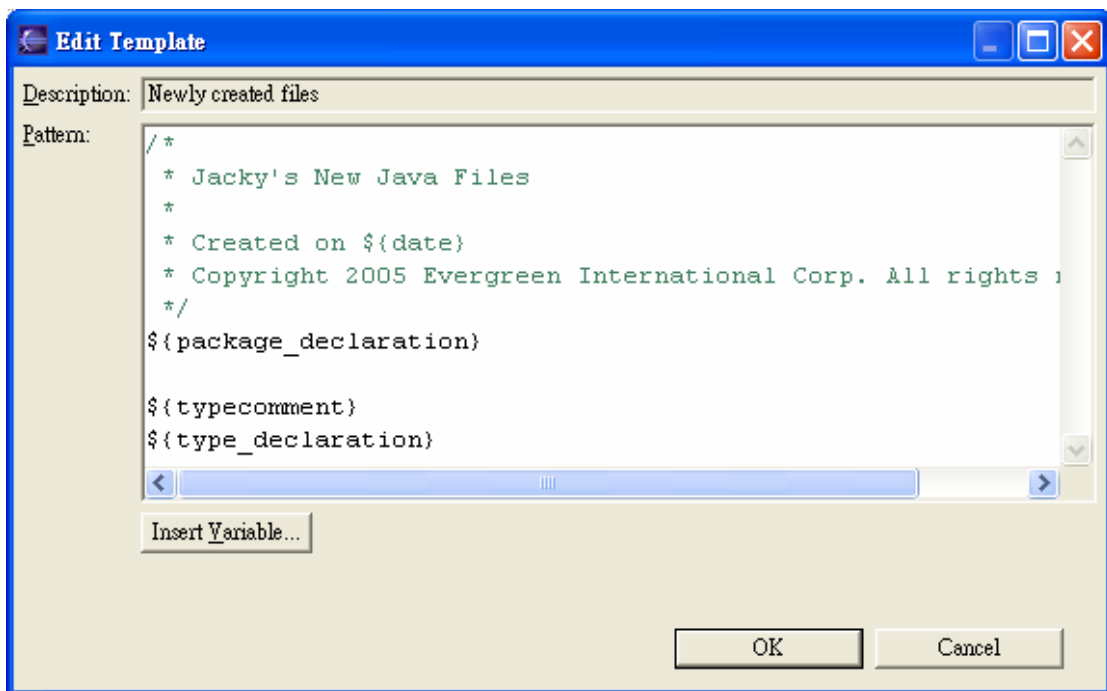


图 4.24

IV. 按 OK

除了 New Java file 的模板外，还需要修改另一个模版-类型批注(Typecomment)。

- I. 「Window」 「Preferences」 「Java」 「Code Style」 「Code Templates」
- II. 选右边画面的「Comments」 「Types」, 按 Edit 按钮

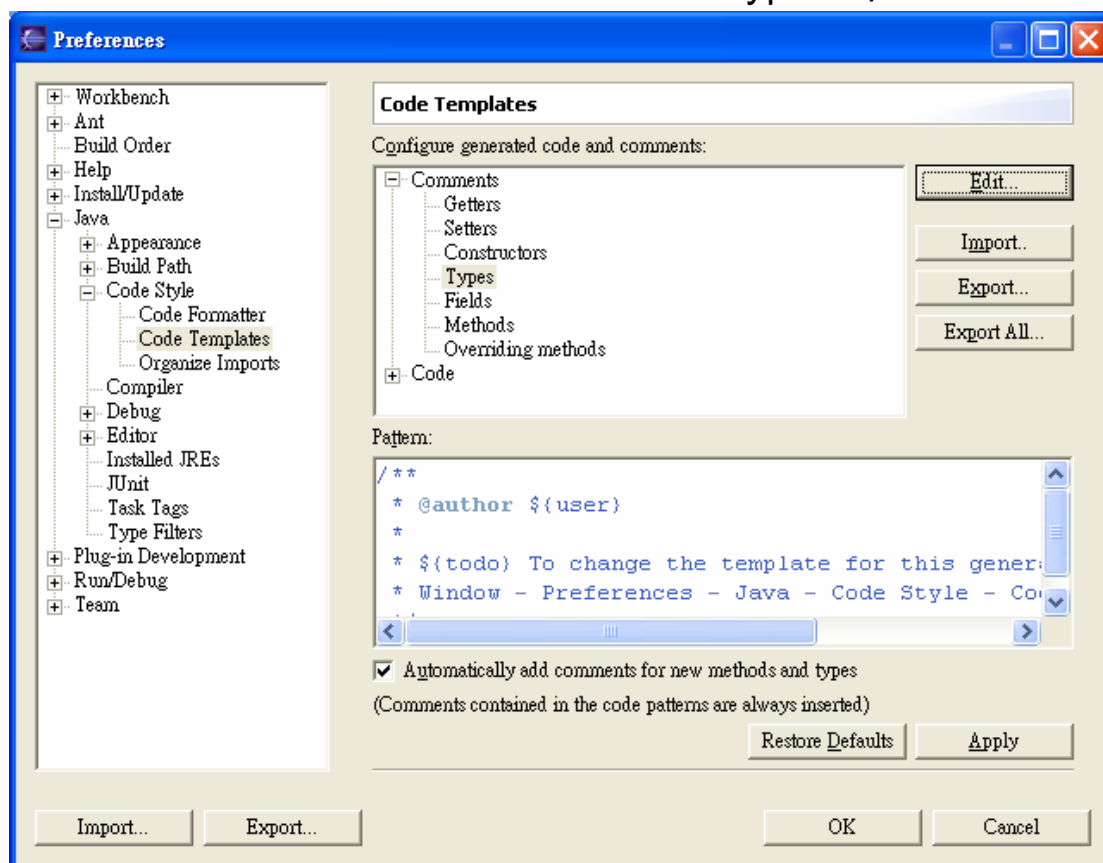


图 4.25

- III. 修改成需要的格式

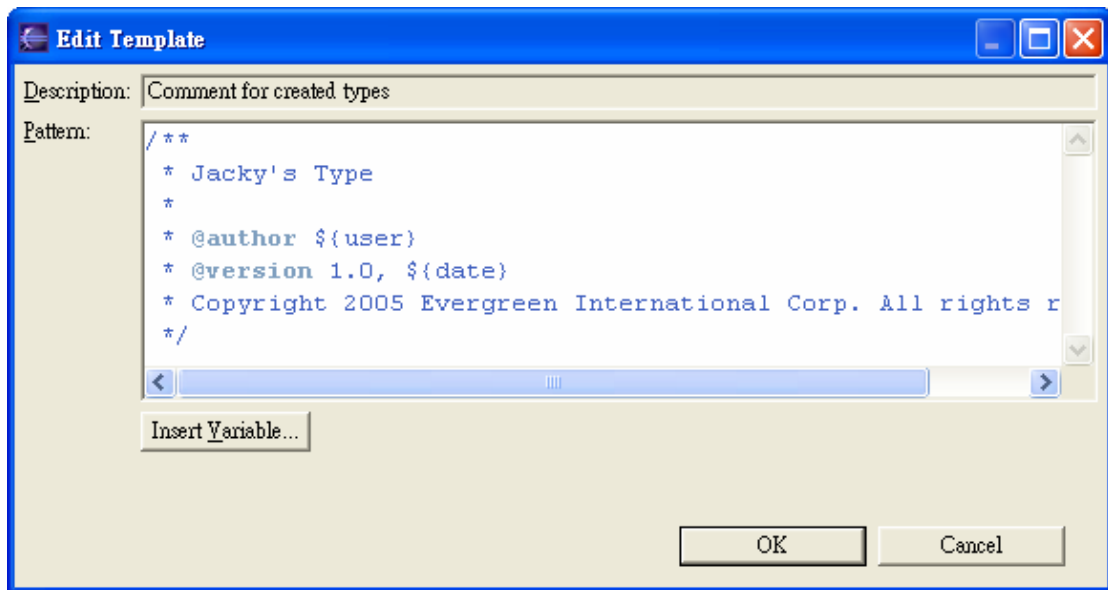


图 4.26

IV. 按 OK

往后新增的类别档案，就会套用现在批注。

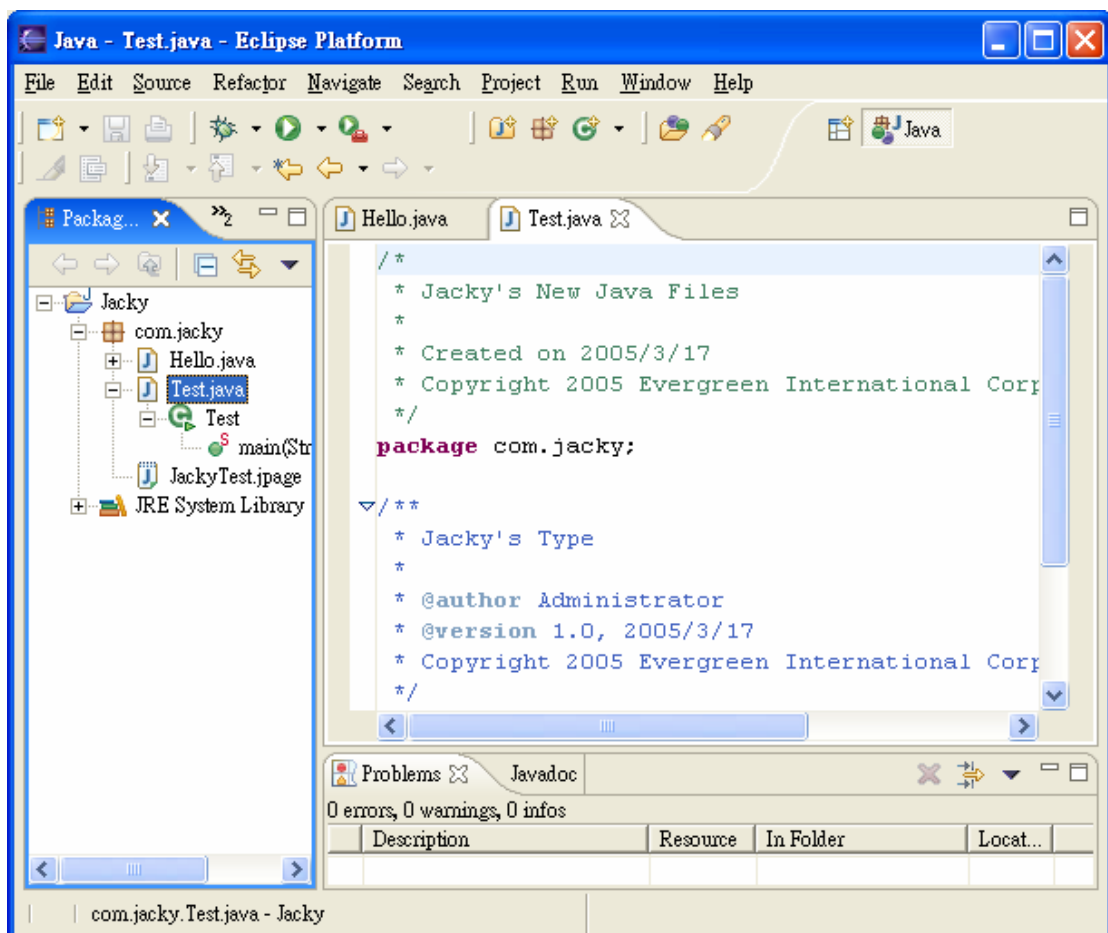


图 4.27

Javadoc 也可以产生模板 ,做法跟 [4.6.2 程序代码产生模板](#)类似 ,差别在于 Context 改选 javadoc。

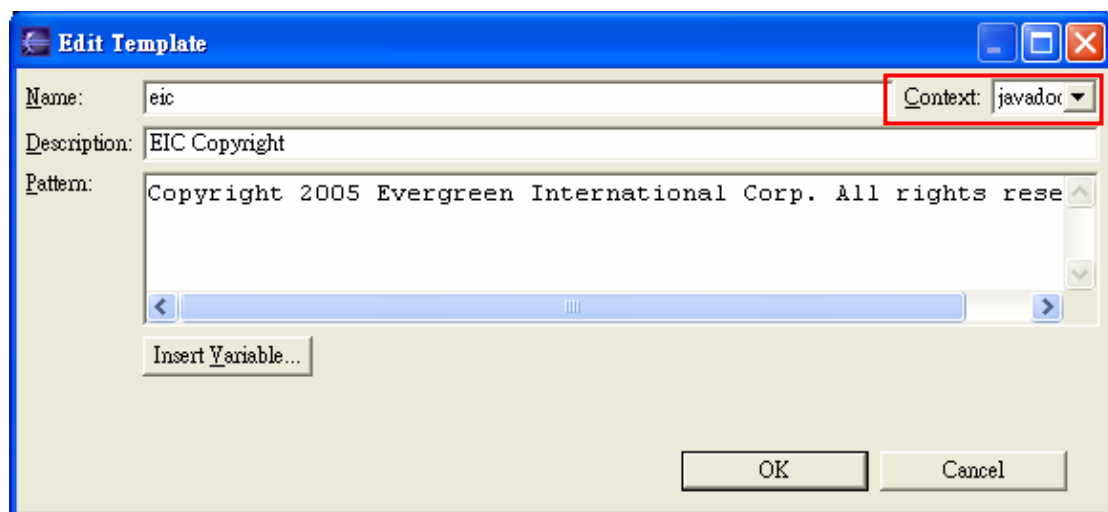


图 4.28

在程序批注的地方 ,打 eic 再按 Alt - / ,就可以出现清单可以选择。

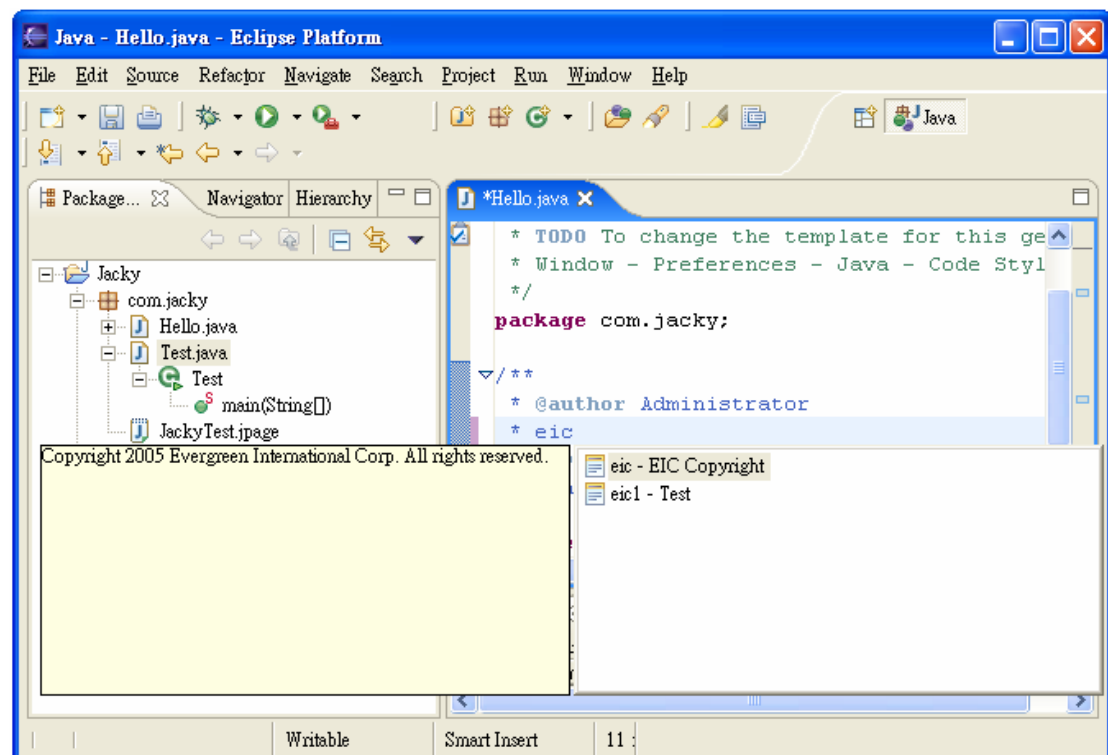


图 4.29

4.7 产生 getter 与 setter

Java 编辑器可以为编译单元内的类型字段，产生存取元 (accessors，也就是 getter 和 setter 的 method)。

I. 「Source」 「Generate Getter and Setter...」

(或是在 Java 编辑器按右键，「Source」 「Generate Getter and Setter...」)

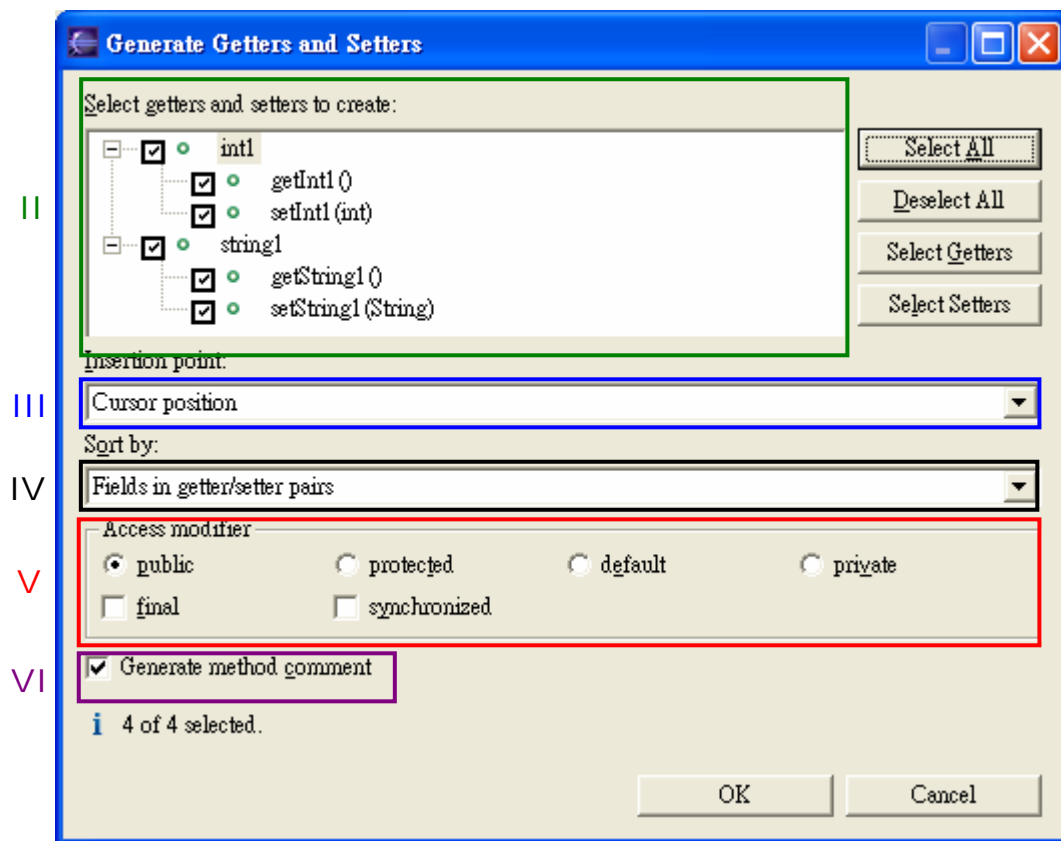


图 4.30

II. 挑选哪些需要建立 getter 和 setter 的 method

III. 选择 method 要建立的地方

IV. 排序的方式

V. 选择 Access modifier

VI. 选择是否需要建立批注

VII. 按 OK

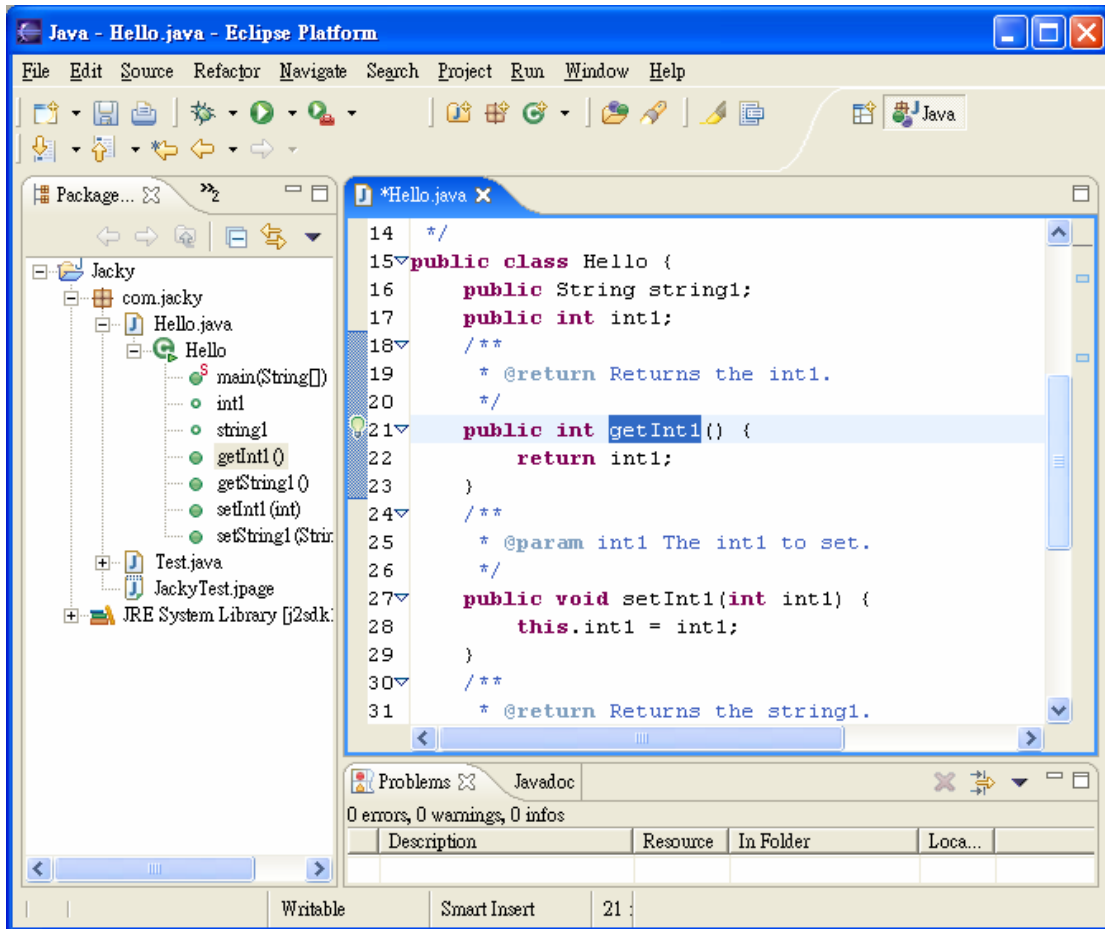


图 4.31

4.8 建立 JAR 档案

4.8.1 建立新的 JAR 档案

如果要在工作台中建立新 JAR 档，请执行下列动作：

- I. 在「Package Explorer」中，可以选择性地预选一或多个要汇出的 Java 元素。（在步骤 IV 中，这些会在 JAR Package Specification 精灵页面中自动选出。）
- II. 从快速菜单或从菜单列的 File 菜单，选取 Export。
- III. 选取 JAR file，然后按一下 Next。

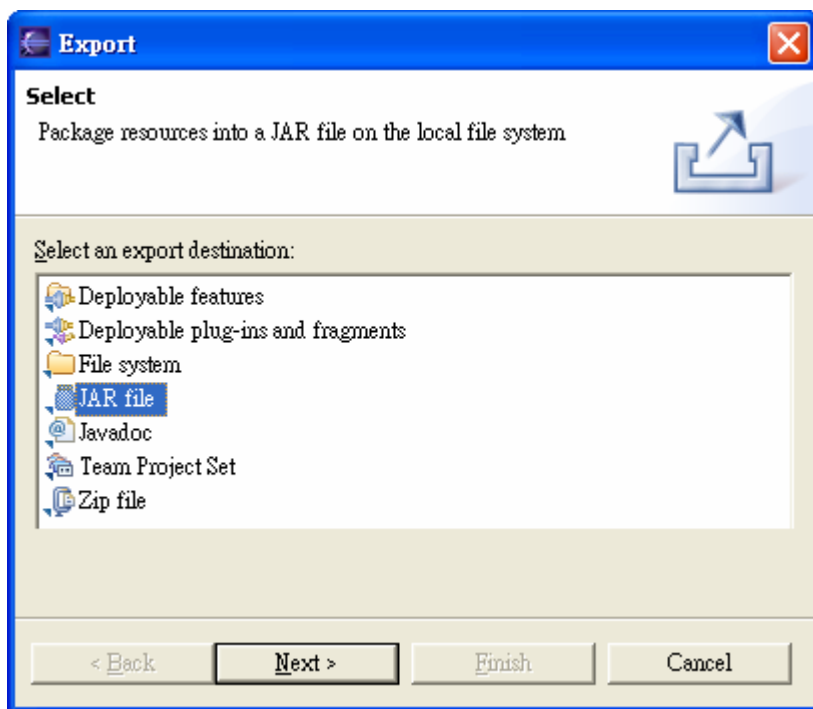


图 4.32

- IV. 在 JAR Package Specification 页面的 Select the resources to export 字段中，选取要汇出的资源。
- V. 选取适当的勾选框，以指出想 Export generated class files and resourcess 或 Export java source files and resources。附注：这两种情况皆会汇出所选的资源。
- VI. 在 Select the export destination 字段中，输入或按一下 Browse 以选取 JAR 文件的位置。
- VII. 选取或清除 Compress the contents of the JAR file 勾选框。
- VIII. 选取或清除 Overwrite existing files without warning 勾选框。如果清除这个勾选框，则会提示确认是否要更换每一个将被改写的档案。
- IX. 附注：在撰写 JAR 档、JAR 说明与 Manifest 档时，会套用改写选项。
- X. 有两项选择：
 - 按一下 Finish 来立即建立 JAR 档。
 - 按一下 Next，使用「JAR 套装选项」页面，以设定进阶选

项，建立 JAR 说明，或变更预设 manifest。

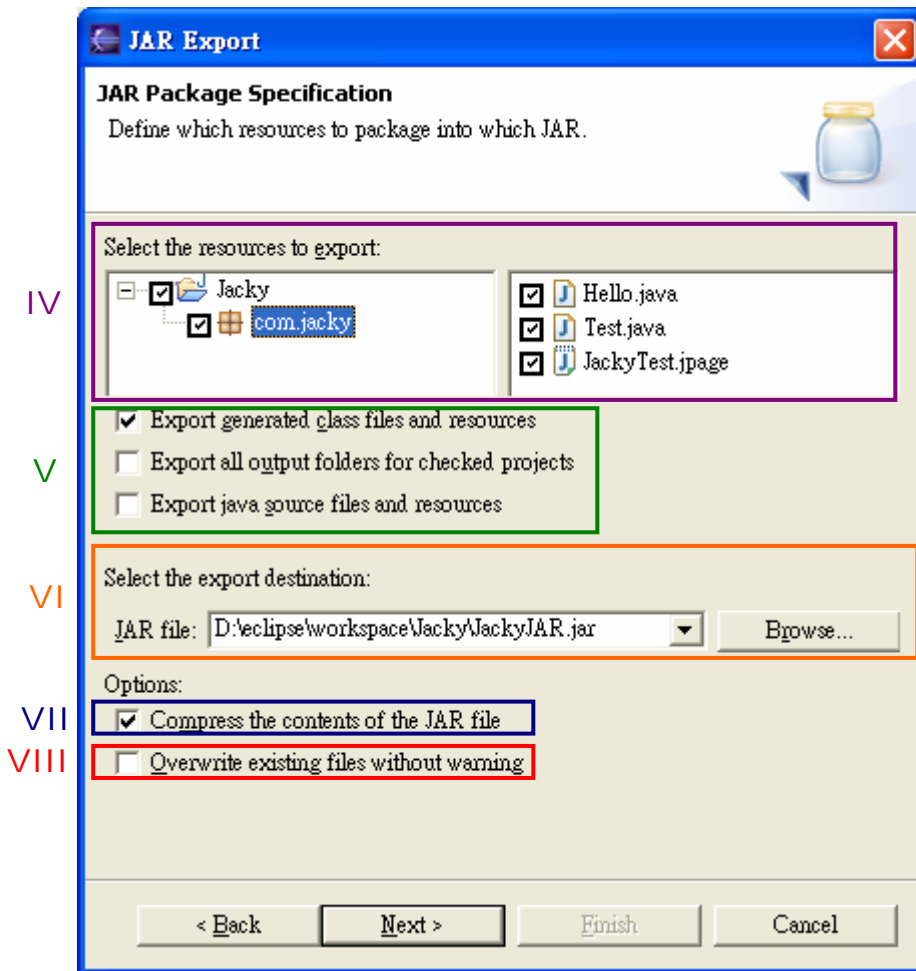


图 4.33

4.8.2 设定进阶选项

- I. 遵循建立 JAR 文件的程序进行，但在最后一个步骤中按一下 Next，以移至「JAR 套装选项」页面。
- II. 如果想储存 JAR 档说明，请选取 Save the description of this JAR in the workspace 勾选框。
- III. 编译器能产生 CLASS 文件，即使程序文件中有错误。可以选择排除内含编译错误的 CLASS 文件（但非程序文件）。如果有启用报告特性的话，则结束时将会报告这些档案。

IV. 可以选择排除内含编译警告的 CLASS 文件（但非程序文件）。
在结束时将会报告这些档案。

附注：这个选项不会自动排除内含编译错误的类别档。

V. 可以选择包含来源数据夹路径，方法是选取 Create source folder structure 勾选框。

VI. 如果希望让汇出作业在建立 JAR 文件前先执行建置，请选取 Build projects if not built automatically 勾选框。

VII. 按一下 Finish，以立即建立 JAR 档；如果想变更预设 manifest，请按一下 Next。

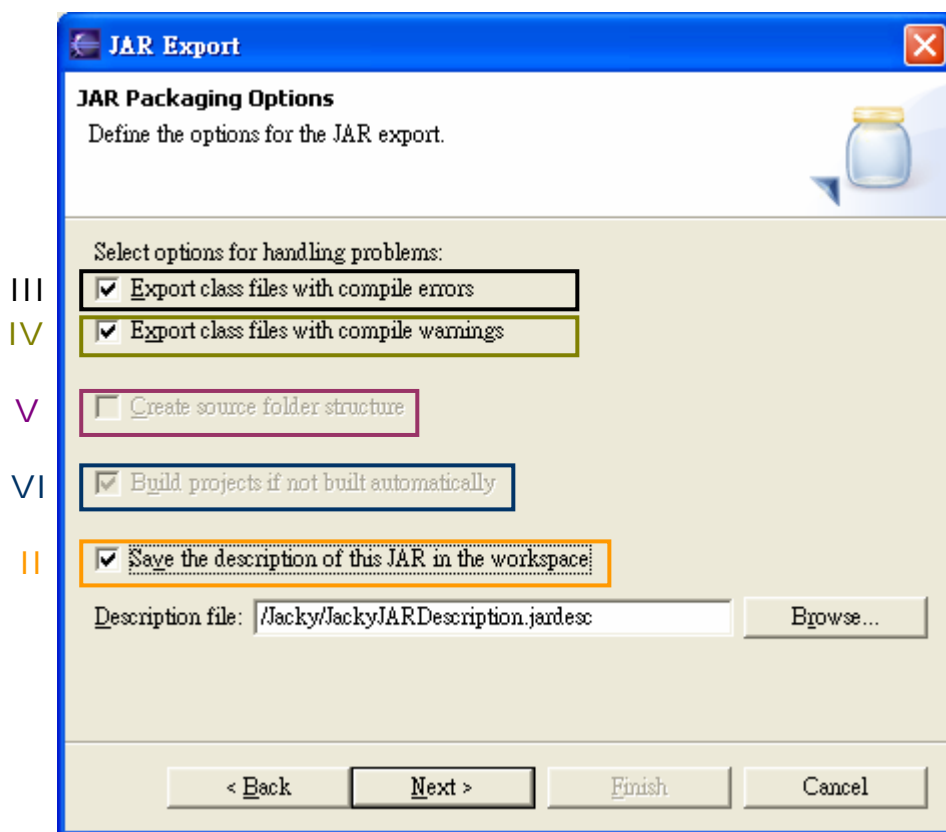


图 4.34

4.8.3 定义 JAR 档的 manifest

可以直接在精灵中定义 JAR 档 Manifest 的重要部分，或使用

已存在于工作台中的 Manifest 檔。

建立新 Manifest

- I. 遵循建立 JAR 文件的程序进行，但在最后一个步骤中按一下 Next，以移至「JAR 套装选项」页面。
- II. 设定任何要设定的进阶选项，再按一下 Next，移至「JAR manifest 规格」页面中。
- III. 如果尚未选取，请按一下 Generate the manifest file 按钮。
- IV. 这时可以选择将 Manifest 储存在工作台中。这会储存 Manifest 以便日后使用。按一下 Save the manifest in the workspace，然后按一下 Manifest file 字段旁的 Browse，以指定 Manifest 的路径与文件名。
- V. 如果在前一步骤中决定储存 Manifest 档，并在先前的精灵页面中选择储存 JAR 说明，可以选择在 JAR 说明中重复使用 Manifest（做法是选取 Reuse and save the manifest in the workspace 勾选框）。这表示当从 JAR 说明重建 JAR 档时，将会使用所储存的档案。如果想先修改或更换 Manifest 档，然后再从说明重建 JAR 档，请善用这个选项。
- VI. 可以选择密封 JAR，以及选择性地将某些套件排除在密封之外，或指定密封套件清单。依预设，不会进行任何密封。
- VII. 按一下 Main Class 字段旁的 Browse 按钮来指定应用程序的进入点。
附注：如果的类别不在清单中，表示在一开始时忘了选取它。
- VIII. 按一下 Finish。这会建立 JAR，并选择性地建立 JAR 说明与 Manifest 檔。

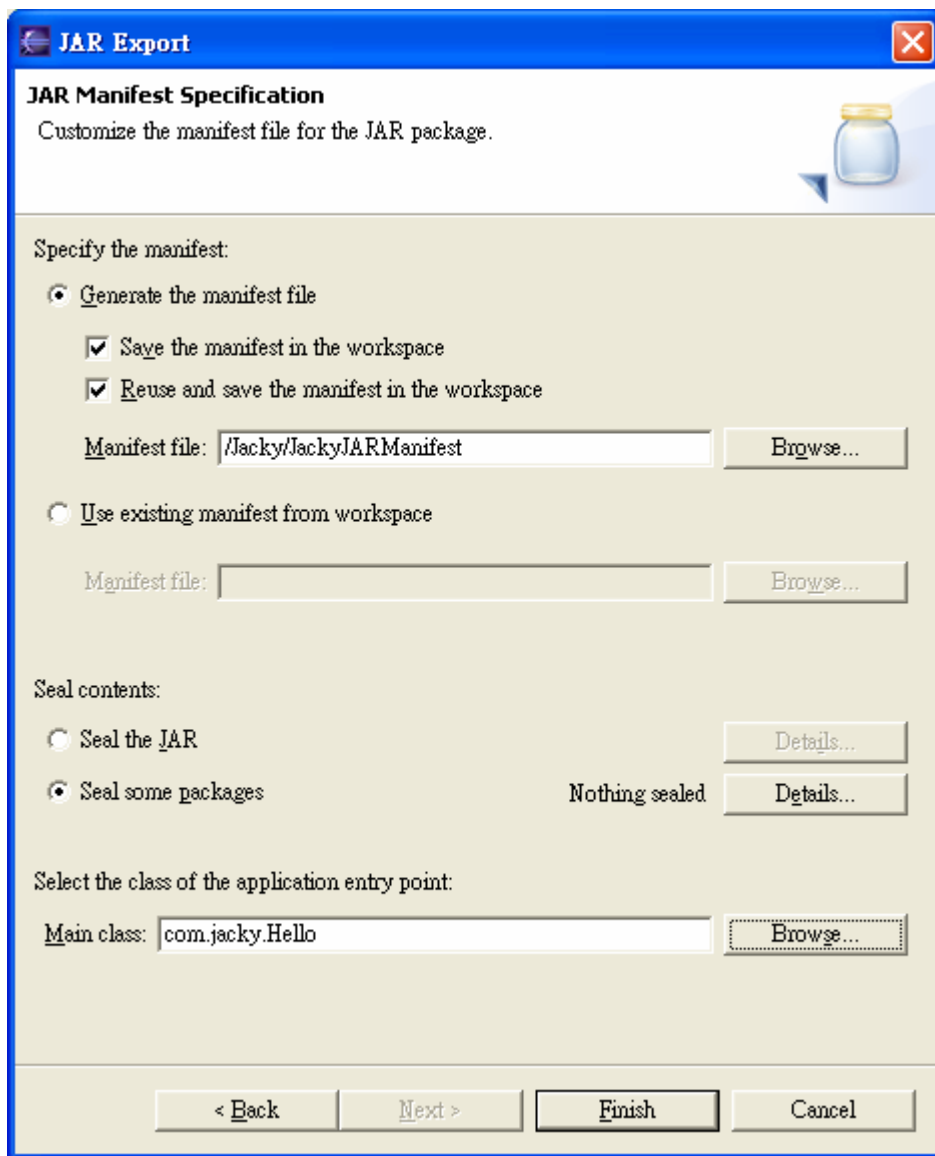


图 4.35

使用现有的 manifest

可以使用已存在于工作台中的现有 Manifest 檔。

- I. 遵循建立 JAR 文件的程序进行，但在最后一个步骤中按一下 Next，以移至「JAR 套装选项」页面。
- II. 设定任何要设定的进阶选项，再按一下 Next，移至「JAR manifest 规格」页面中。
- III. 按一下 Use existing manifest from workspace 圆钮。
- IV. 按一下 Browse 按钮来从工作台选取 Manifest 檔。
- V. 按一下 Finish。这会建立 JAR，并选择性地建立 JAR 说明。

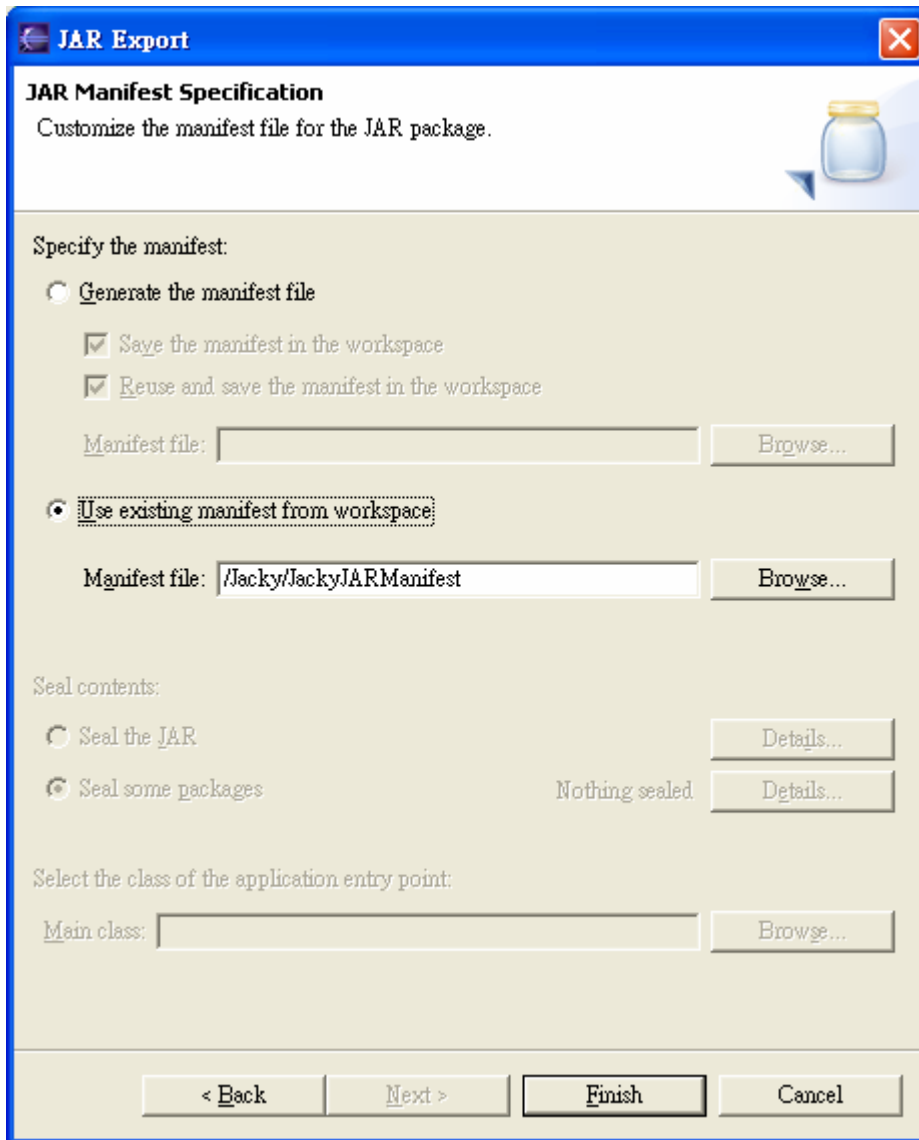


图 4.36

4.8.4 重新产生 JAR 檔

可以使用 JAR 档说明来重新产生先前所建立的 JAR 檔。

从选项的蹦现菜单中，选取 Create JAR。这时会重新产生 JAR 檔。

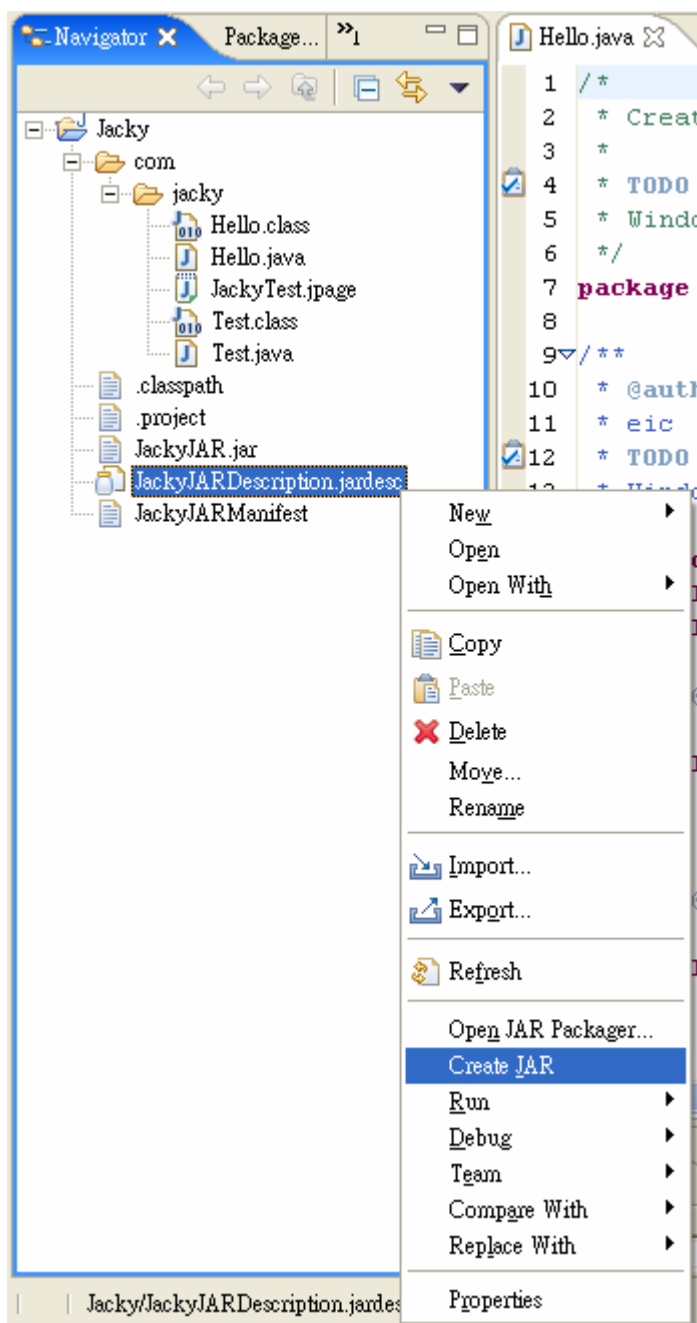


图 4.37

4.9. 建立 Javadoc 文件

选取所要的套件、来源资料夹或项目组（内含一或多个元素），以便为其产生 Javadoc 文件。

执行下列之一，以开启「汇出」精灵：

- 选取选择项之蹦现菜单中的汇出；或
- 从菜单列中选取「File」→「Export...」。

在出现的对话框中，从清单中选取 Javadoc，并按下下一步。

4.9.1 选取产生 Javadoc 用的类型

- I. 指定 Javadoc 指令的位置。
- II. 在树状结构控制中，选取要的元素，以为其产生 Javadoc。
- III. 使用为具有可见性的成员建立 Javadoc 下的圆钮，选取可见性。
- IV. 维持选取使用标准 doclet 圆钮。
(或是自定 doclet)
- V. 按下完成，为所选的元素产生 Javadoc，或按下下一步，指定其它选项。

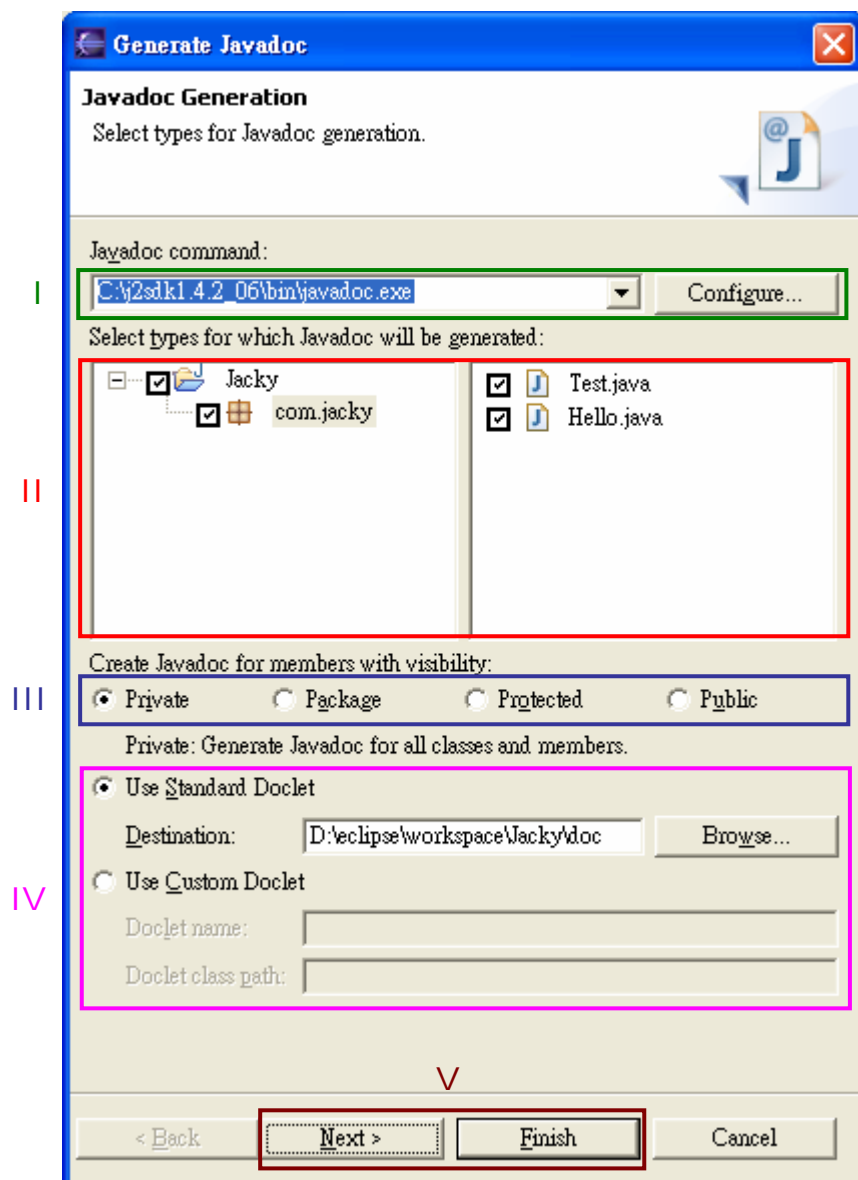


图 4.38

4.9.2 为标准 doclet 配置 Javadoc 自变量

I. 标题名称

II. 请使用基本选项下的勾选框，以指定 Javadoc 选项。

可以使用阐明这些标示群组中的勾选框，以变更所要阐明的标示。

III. 如果想让链接库中之类别的参照，链接至链接库的 Javadoc，

请在清单中选取该链接库，并按下配置，以指定链接库之 Javadoc 的位置。

IV. 选择 CSS 档案

V. 按下完成，以产生 Javadoc；或按下下一步，以指定其它的 Javadoc 产生选项。

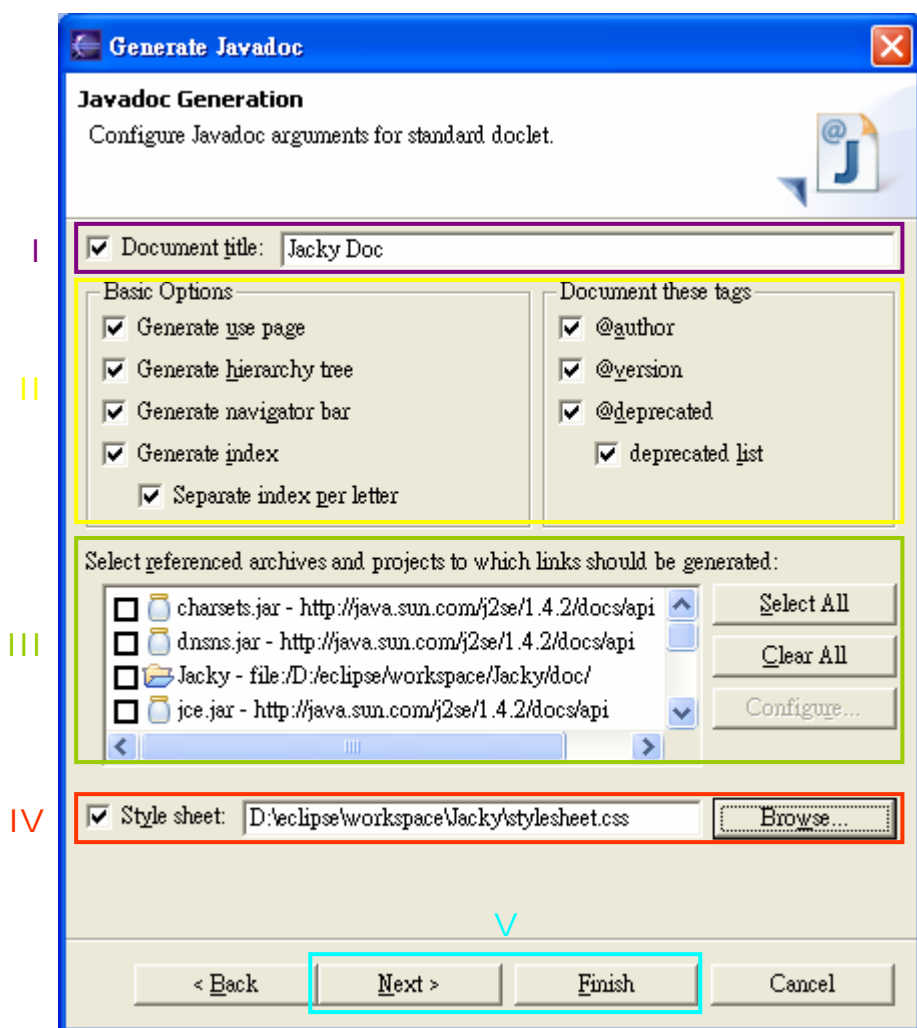


图 4.39

4.9.3 配置 Javadoc 自变量

I. 选取在浏览器中开启产生的索引文件勾选框。

II. 可以为 Javadoc 指令指定多个特定选项，方法是在文字区中输入

入这些选项。

III. 选取将这个 Javadoc 汇出的设定储存成 Ant Script 勾选框。

IV. 指定 Ant Script 的位置。

V. 按完成，以产生 Javadoc。

附注：Javadoc 指令产生的输出（包括错误与警告）会显示在「Console」视图中。

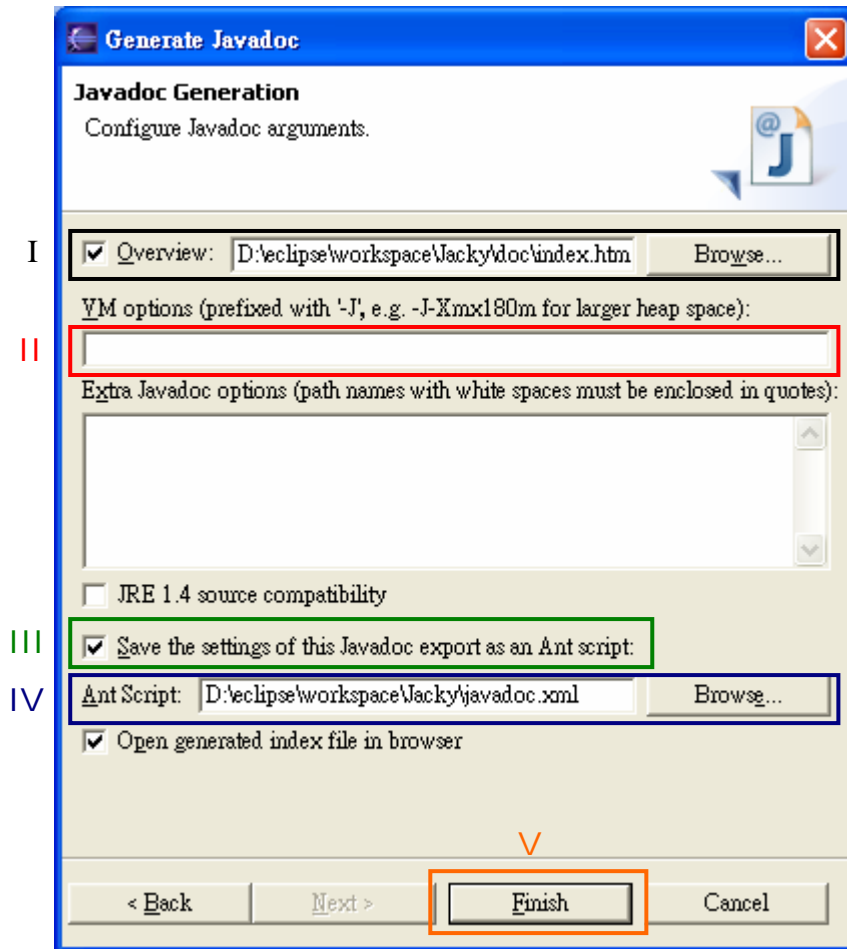


图 4.40


4.10 工作集(Working Sets)

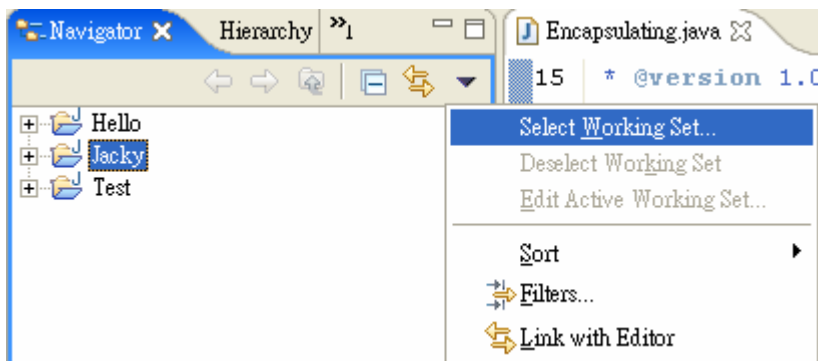
工作集将元素分组，以显示在视图中，或对元素集进行作业。

「Navigate」视图使用工作集来限制显示的资源集。如果在导览器中选取了工作集，只会显示资源、资源的子项，以及资源的母项。

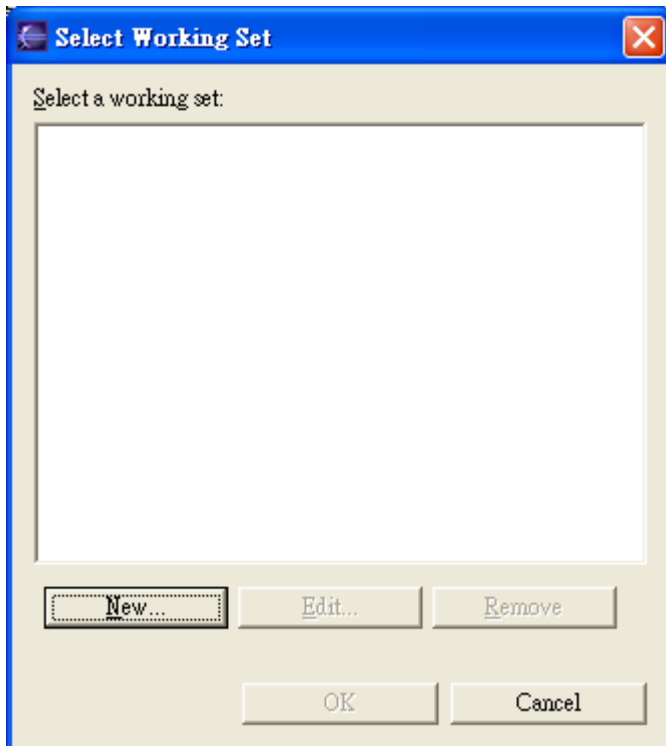
可以在「Tasks」视图中使用工作集来限制作业的显示，方法与「Navigate」视图类似。在使用工作台搜寻机能时，可以使用工作集来限制可搜寻的元素集。不同的视图提供不同的指定工作集方法。不过，它们通常使用下列工作集选项对话框来管理现有的工作集以及建立新的工作集。

4.10.1 新增工作集

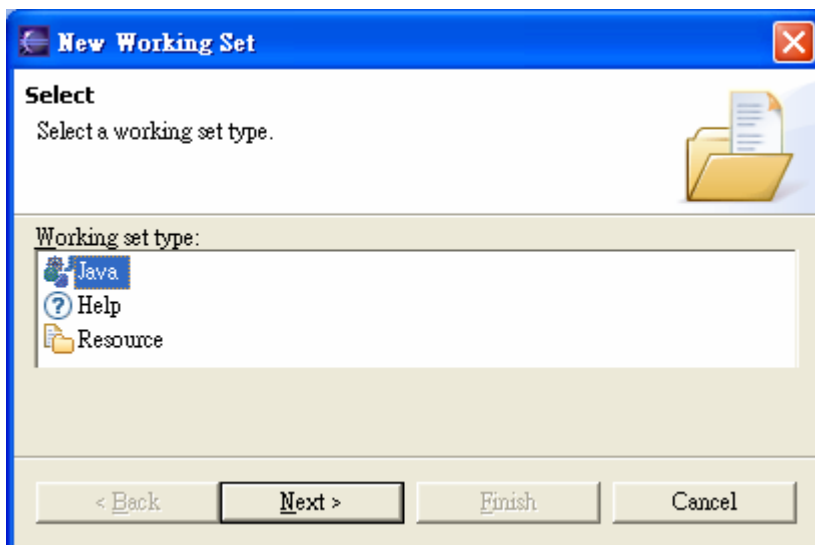
- I. 在「导览器」视图的工具列上，按一下菜单按钮 ，开启显示选项的下拉菜单。



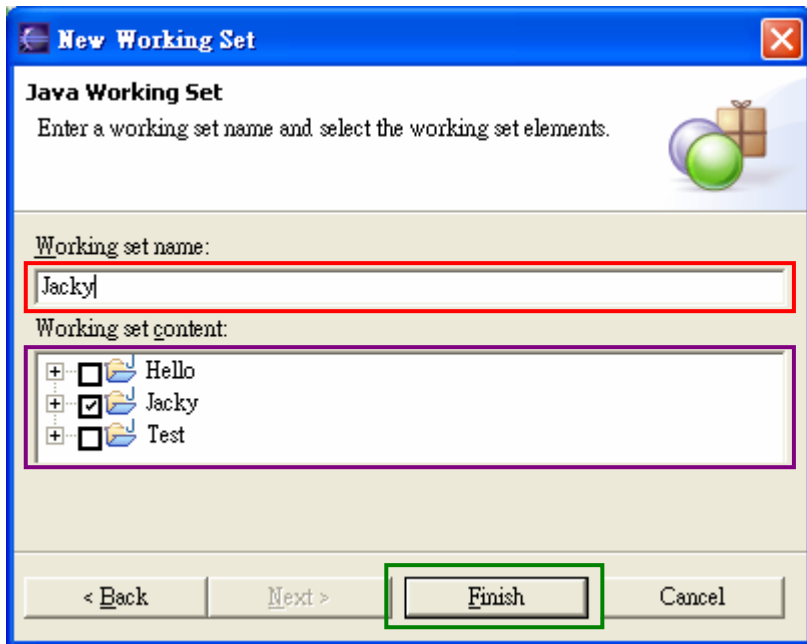
- II. 选 Select Working Set 后，出现 Select Working Set 的窗口



III. 选择 Java 后，按 Next




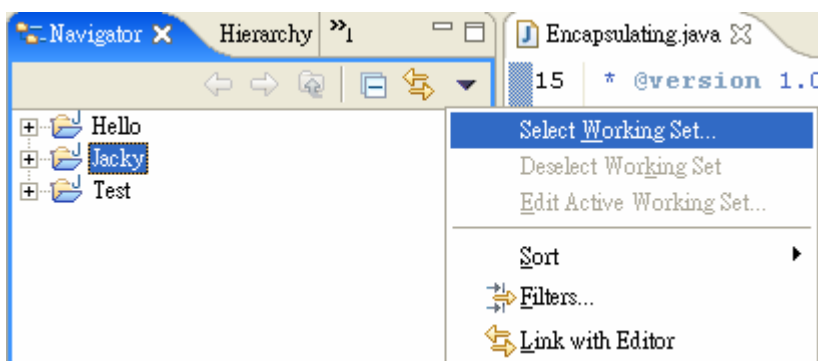
VI. 在 New Working Set 窗口输入名称和勾选所需的项目，最后按下 Finish 即可



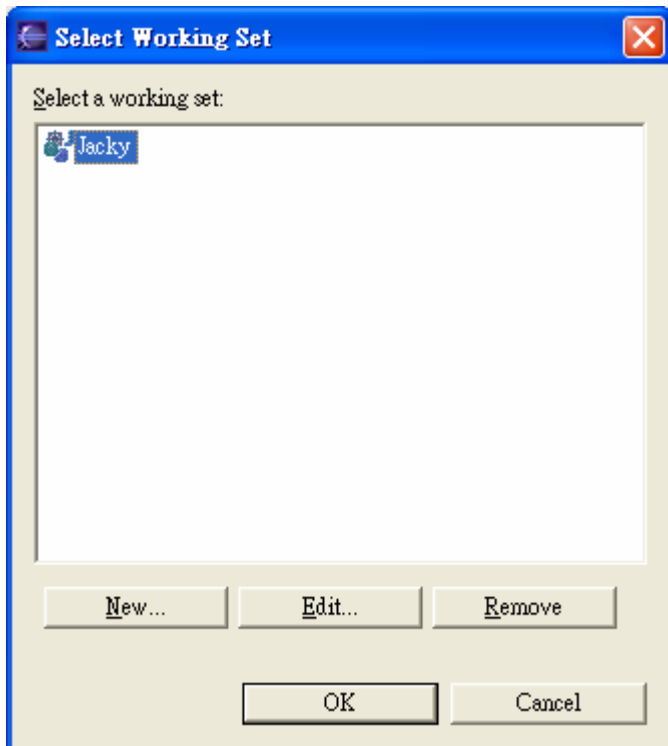
附注：新建的资源不会自动并入作用中的工作集。如果它们是现有的工作集元素的子项，则会被隐含地并入工作集中。如果要在建立其它资源之后并入它们，必须明确地将它们新增至工作集。

4.10.2 隐藏「导览器」视图中的档案

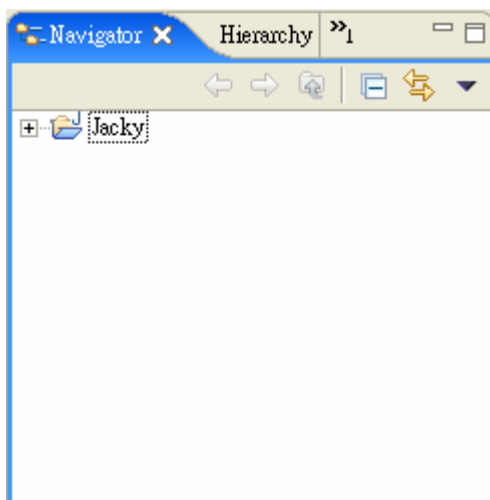
- I. 在「导览器」视图的工具列上，按一下菜单按钮，开启显示选项的下拉菜单。




- II. 选 Select Working Set 后，出现 Select Working Set 的窗口



111. 从清单中选取一个现有的工作集，来隐藏「导览器」视图中的档案



4.10.3 显示「导览器」视图中的档案

1. 在「导览器」视图的工具列上，按一下菜单按钮 ，开启显示选项的下拉菜单。