第1章 Eclipse 简介

内容摘要 | Abstract

Eclipse 是一款非常优秀的开源 IDE（集成开发环境），基于 Java 的可扩展开发平台。除了可作为 Java 的集成开发环境外，还可以作为编写其他语言（如 C++和 Ruby）的集成开发环境。Eclipse 凭借其灵活的扩展能力、优良的性能与插件技术，受到了越来越多开发者的喜爱。

本章将介绍 Eclipse 开发工具，首先介绍其产生原因和体系结构，接下来介绍 JDK 安装和 Eclipse 汉化，然后再介绍 Eclipse 工具的使用，例如文件结构、代码编辑和文件查找等，最后介绍使用 Eclipse 开发 Java 应用程序。

学习目标 | Objective

- 了解 Eclipse 的产生原因
- 了解 Eclipse 的体系结构
- 掌握 JDK 的安装和配置
- 熟练掌握 Eclipse 的启动和汉化
- 掌握 Eclipse 窗口组成
- 熟练掌握文件结构、代码编辑和查找等功能
- 熟练掌握使用 Eclipse 开发 Java 程序

1.1 Eclipse 概述

很长时间以来，记事本加 JDK 的开发方式占据着 Java 开发的重要地位，并且被多数 Java 开发人员所接受，甚至成为衡量一个 Java 程序员水平高低的标准。然而，Eclipse 的出现，彻底改变了 Java 开发方式，越来越多的 Java 程序员开始将目光投向这个“神奇”的集成开发环境。

1.1.1 Eclipse 的产生和发展

在 1998 年左右，IBM 公司遇到了一些挑战，当时 IBM 的开发工具 Visual Age for Java 和 WebSphere StudioIE 难以集成到一起，并且底层的技术比较脆弱，这对于公司的进一步发展非常不利，因为以前的开发环境已经无法满足业界应用开发的需求。

IBM 意识到这个问题之后，成立了一个项目开发小组来开发新的 IDE，期望能解决上述矛盾。2000 年，项目小组决定将其新一代开发工具命名为 Eclipse。开发小组希望 Eclipse 项目

能够吸引开发人员，从而发展起一个强大而又充满活力的商业合作伙伴（独立软件供应商）社区。为了和 Microsoft Visual Studio 竞争，IBM 决定推出 Eclipse 的试用计划，并初步决定将 Eclipse 贡献给开源社区。

2001 年 12 月，IBM 成立了以支持和促进 Eclipse 开源项目为主要任务的 Eclipse 协会，并向开源社区捐赠了价值 4000 万美元的源代码，从此 Eclipse 走上了飞速发展的道路。2004 年初，为了促使 Eclipse 项目更好地发展，IBM 和其他成员公司一起成立了 Eclipse 基金会（Eclipse Foundation）。

Eclipse 目前主要由 4 个项目组成，分别是 Eclipse 项目、Eclipse 工具项目、Eclipse 技术项目以及 Eclipse Web 工具平台项目，每个项目有一个项目管理委员会监督，并使用 GPL 1.0 协议。

Eclipse 工具项目的主要任务是为 Eclipse 平台生产最好的构建工具；Eclipse 技术项目的主要任务是考虑到 Eclipse 今后的发展，为平台培养后续的接班人；Eclipse Web 工具平台项目的主要任务是提供更好的 Web 开发平台以及工具。

利用 Eclipse，读者可以将高级设计（例如 UML）与低级开发工具（如应用调试器等）结合在一起。如果这些相互补充的独立工具采用 Eclipse 扩展点彼此连接，那么当用户用调试器逐一检查应用时，UML 对话框可以突出显示用户正在关注的器件。事实上，由于 Eclipse 并不了解开发语言，所以无论 Java 语言调试器、C/C++调试器还是汇编调试器都是有效的，并可以在相同的框架内同时瞄准不同的进程或节点。

Eclipse 的最大特点是能接受由 Java 开发者自己编写的开放源代码插件，这类似于微软公司的 Visual Studio 和 Sun 公司的 NetBeans 平台。Eclipse 为工具开发商提供了更好的灵活性，使他们能更好地控制自己的软件技术。

最初的 Eclipse 开发人员大部分来自于 Visual Age for Java 项目组，尽管 Eclipse 目前由开源组织 Eclipse.org 管理，但是 Eclipse 的开发仍然由 IBM 的子公司 OTI（主要从事 Eclipse 开发）继续领导着。Eclipse 支持当前几乎所有的主流平台，包括 Windows、Linux、Solaris、HP-UXS 和 AIX，这就大大减低了开发跨平台软件的难度。Eclipse 版本更新比较迅速，本书完成时的最新版本是 3.4。

1.1.2 Eclipse 结构

随着 Java 的应用越来越广泛，围绕 Eclipse 的应用开发也越来越受到关注，各大主要软件工具提供商都参与到基于 Eclipse 架构的开发中，并且针对开放源代码 Eclipse 插件项目的数量正在与日俱增。

例如对 UML 建模的支持，能够对任何 Java 代码绘制 UML 图。这些都使得 Eclipse 逐渐发展成为一个功能强大、非常适合分布式计算的应用开发平台。

Eclipse 平台是一个开放的、通用的、可扩展的集成开发环境。Eclipse 项目提出的目标是：

- 提供应用开发工具的开放平台。能够在多个操作系统（如 Windows、Linux、Unix）上运行，支持基于 GUI 和非 GUI 的应用开发环境。
- 语言中立性。允许无限制的内容类型 HTML、Java、C、JSP、EJB、XML 和 GIF 等。
- 支持多种工具的无缝集成。
- 吸引工具开发者团体。

事实上，Eclipse的目标不仅仅是提供一个IDE。在Eclipse中还包括了插件开发环境（Plug-in Development Environment，PDE），该环境使得开发人员可以根据需要随时添加新的插件，从而能够轻松构建与 Eclipse 环境无缝集成的开发环境。

Eclipse 中最出色的部分莫过于其插件框架，正因为采用了插件机制，Eclipse 才得以被不断扩充，越来越强大。

整个 Eclipse 体系结构就像一个大拼图，可以不断地往其中添加插件，同时，现有插件上还可以再增加插件。可以说，在 Eclipse 平台中，几乎一切都是插件。从架构上来讲，Eclipse 基本采用的是“内核+核心插件+定制插件”的结构体系，除了内核部分外，其余部分均为插件，其体系结构如图 1-1 所示。

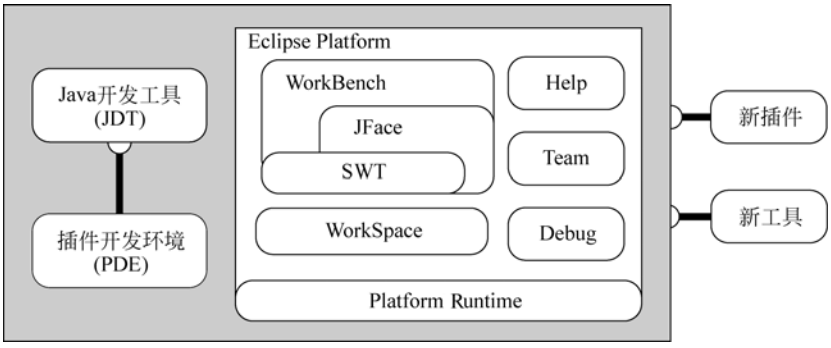

图 1-1　Eclipse 体系结构图

各组成部分功能为：

- Platform Runtime（平台运行库）是 Eclipse 平台核心，它是一个微内核，负责管理插件注册表和插件，其他所有的功能如 WorkSpace 和 WorkBench 等都是以插件的形式提供。Eclipse 启动时，平台运行库需要完成 2 个简单的动作：一是启动平台；二是查找系统的功能插件，并在内存中构建一个插件注册表。
- WorkSpace（工作空间）是负责管理用户资源的插件。用户资源包括用户创建的项目和项目中的文件等。WorkSpace 相当于 Model-View-Controller 结构中的 Model。
- WorkBench（工作台）插件为 Eclipse 提供用户界面。它为添加用户界面（UI）组件（例如视图、菜单）定义了扩展点（Extension Points），同时也提供了附加工具集（SWT 和 JFace）用于建造用户界面。工作台是在 JFace/SWT 和平台运行库的基础上构建。WorkBench 相当于 Model-View-Controller 结构中的 View。WorkBench 和 WorkSpace 是 Eclipse 平台的 2 个必备插件，它们提供了大多数插件使用的扩展点。
- SWT（Standard Widget Toolkit，标准小窗口工具箱）是 Eclipse 平台自己开发的图形 API 包，作用和 Java 的 AWT/Swing GUI API 一样，用来构建图形界面。
- JFace是使用SWT来实现的一个用户界面工具包，用于处理常见的用户界面编程任务，使工具开发者专注于实现插件的功能，而不必在界面设计上花过多的精力。
- Help（帮助系统）定义了插件扩展点，提供一个附加的导航结构，允许工具以 HTML 文件的形式添加文档。
- Team（团队）插件负责提供版本控制和配置管理支持。

1.2 安装 Eclipse

Eclipse 是一个开放源代码的项目，可以到其官方网站 www.eclipse.org 免费下载，也可以从其他网站下载。本书所用 Eclipse 为 Windows 平台下的 3.3 版本。虽然 Eclipse 本身是用 Java 语言编写，但下载的压缩包中并不包含 Java 运行环境，因此在使用 Eclipse 开发 Java 程序之前，需要先安装 JDK。

1.2.1 安装 JDK 工具包

JDK 开发工具包是 Sun 公司针对 Java 开发人员提供的软件开发工具包。自从 Java 推出以来，JDK 已经成为使用最广泛的 Java SDK（Software development kit）。用户在编写 Java 程序时，必须用到类库和 Java 语言规范。

JDK 中包括完整的 JRE（Java Runtime Environment，Java 运行环境），也被称为 private runtime。还包括了用于产品环境的各种库类，以及给开发人员使用的补充库，如国际化的库和 IDL 库。JDK 中还包括各种例子程序，用以展示 Java API 中的各部分。

1. 下载 JDK

读者可以从 Sun 的官方网站 http://java.sun.com 上下载最新版本 JDK，进入到 Java SE 6.0 的下载页面。如图 1-2 所示。

图 1-2　下载 Java SE 6.0

在下载窗口中，单击 Download 按钮，就可以下载。这里下载的为 JDK 6 Update 7。

下载完毕后，会发现一个名称 jdk-6u7-windows-i586-p.exe 可执行文件。

2．安装 JDK

找到下载的 JDK 文件 jdk-6u7-windows-i586-p.exe，就可以开始安装了。具体步骤如下所示。

（1）双击 jdk-6u7-windows-i586-p.exe 文件开始进行安装，将打开【许可协议】窗口，单击【接受】按钮，打开【自定义安装】窗口。

（2）在自定义窗口中，可以更改文件的安装路径以及是否安装某些组件。这里将其安装到 C:\Java\jdk1.6.0_07 目录并安装所有组件，如图 1-3 所示。

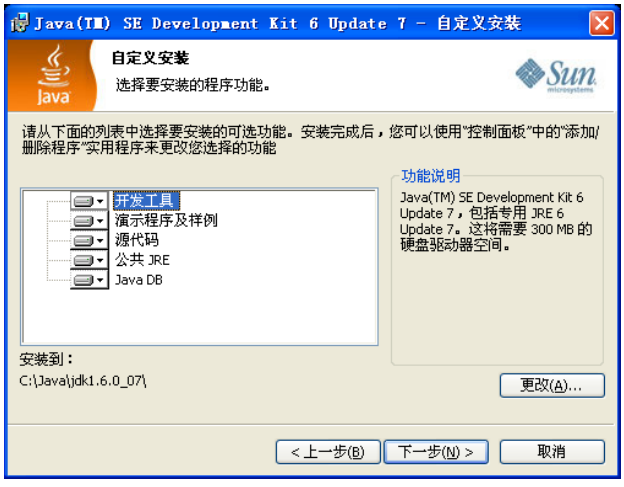

图 1-3　自定义安装窗口

（3）设置完成后，单击【下一步】按钮开始进行安装。

（4）JDK 类库安装完成后，开始安装 JRE，设置安装目录为 C:\Java\jre1.6.0_07，如图 1-4 所示。

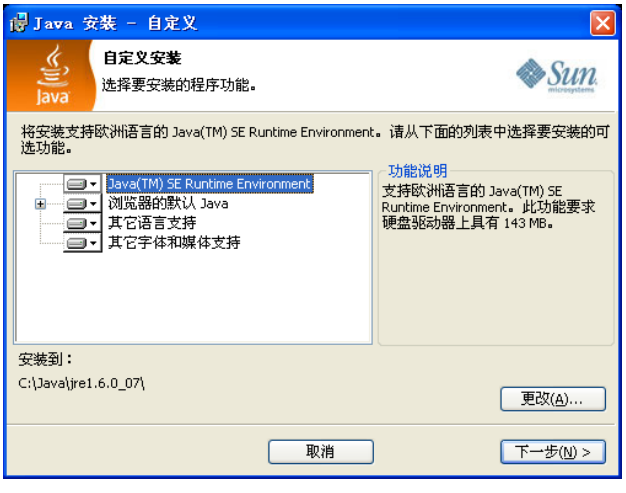

图 1-4　安装 JRE 运行环境

3. 配置 JDK

在安装好 JDK 之后，需要进行一些配置才能继续后面的应用程序开发。具体的配置步骤如下所示。

（1）在 Windows 桌面上，右击【我的电脑】图标，从快捷菜单中执行【属性】命令，将弹出【系统属性】对话框。

（2）在【系统属性】对话框中，选择【高级】选项卡。在该选项卡中，单击【环境变量】按钮，将弹出【环境变量】对话框。

（3）在【环境变量】对话框【系统变量】中，选中变量 path，然后单击【编辑】按钮，在显示的【编辑系统变量】对话框中，加入 C:\Java\jdk1.6.0_07\bin;（即 JDK bin 目录所在路径，注意路径后需要加;），如图 1-5 所示。

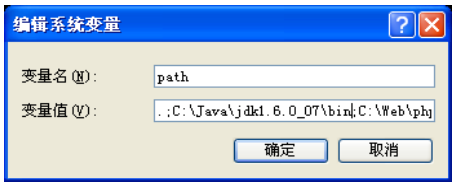

图 1-5　配置 path 变量

（4）按照同样的方式编辑系统变量 classpath，变量值为：

```
.;C:\Java\jdk1.6.0_07\lib\dt.jar;C:\Java\jdk1.6.0_07\lib\tools.jar;
```

（5）这样就完成了 JDK 在 Windows XP 操作系统的安装与配置。在 Windows 2000/NT 系统上的安装也是如此。path 变量必须要进行配置，classpath 环境变量一般情况下不需要配置，只有在计算机上安装了其他的 Java 开发工具时，才需要配置。

为了检测 JDK 6 配置是否成功，打开命令提示符窗口，输入 javac –version 命令，会出现当前 JDK 的版本号。其操作如图 1-6 所示。

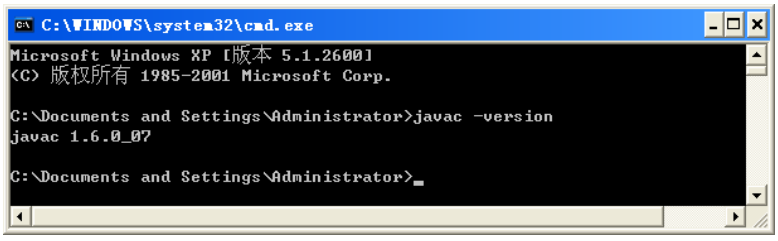

图 1-6　测试 JDK 是否成功

1.2.2　Eclipse 启动和汉化

安装 Eclipse 的操作非常简单，只需将下载的压缩包按原路径直接解压即可。如果有了更新的版本，要先删除旧版本重新安装，不能直接解压到原来的路径覆盖老版本。在解压缩之后可以到相应的安装路径去找 Eclipse.exe 运行。

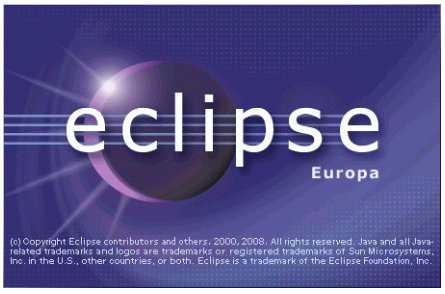

图 1-7　Eclipse 启动界面

1. 启动 Eclipse

启动 Eclipse，会出现一个由月蚀图片构成的闪屏（如图 1-7 所示）。随后出现一个如图 1-8 所示的选择工

作空间路径对话框，Eclipse 会将编辑的所有文件存放在工作空间指定的路径下。

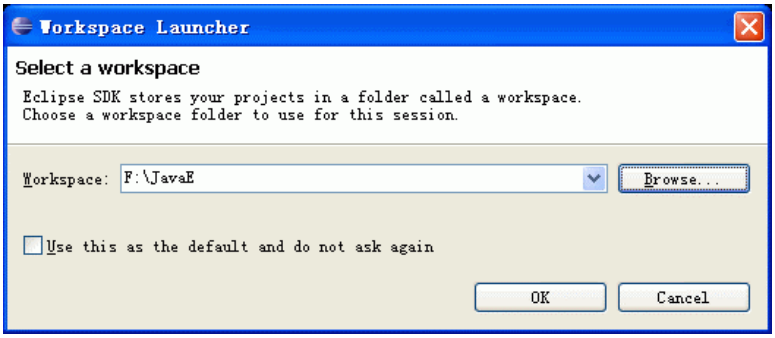

图 1-8　选择工作空间

如果在未安装 JDK 前启动 Eclipse，系统会给出相应的错误信息。提示用户先安装 JDK，并正确配置后再重新启动。

2．汉化 Eclipse

Eclipse 3.3 之后没有相应的汉化版本，因为原来的语言包不能够使用。如果要使用中文简体版本的 Eclipse，需要从其官方网站上自动更新。Eclipse 3.3.x 里都具有 software updates 选项，其升级地址为 http://download.eclipse.org/technology/babel/update-site/。

在 Eclipse 界面的菜单栏上，执行 Help | software updates | Find and Install…命令，打开如图 1-9 所示窗口。

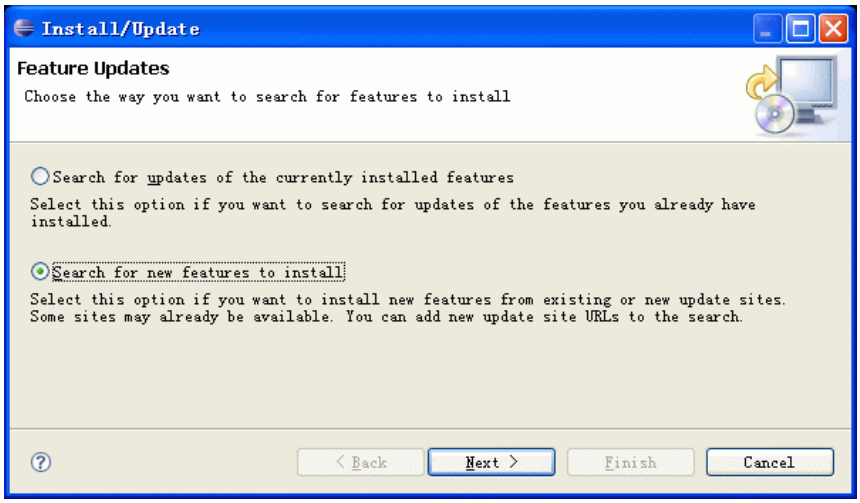

图 1-9　汉化下载包

单击 Next 按钮，打开如图 1-10 所示窗口。

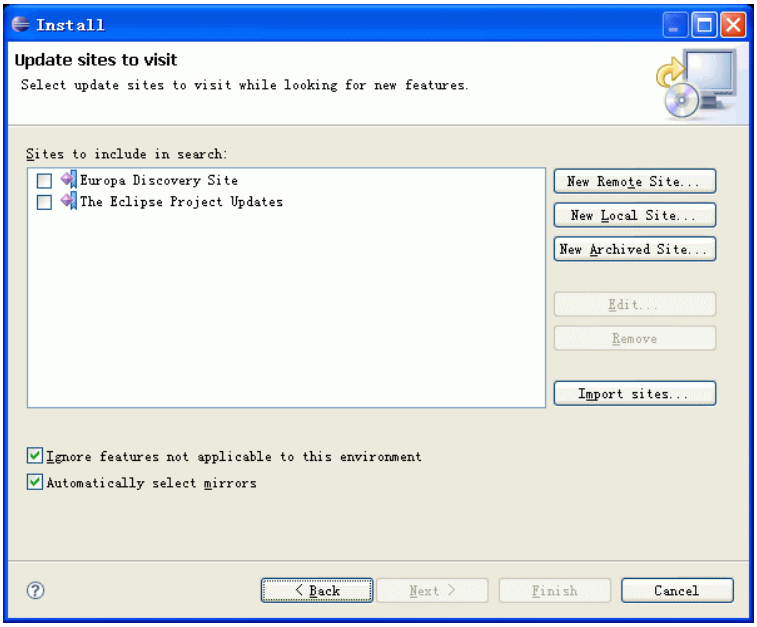

图 1-10　建立下载站点

在图 1-10 中，单击 New Remote Site 按钮，建立一个新的远程下载站点，如图 1-11 所示。在 Name 文本域中输入 AA 作为站点名称，在 URL 文本域中输入下载地址，单击 OK 按钮，打开如图 1-12 所示窗口。

图 1-11　输入下载地址

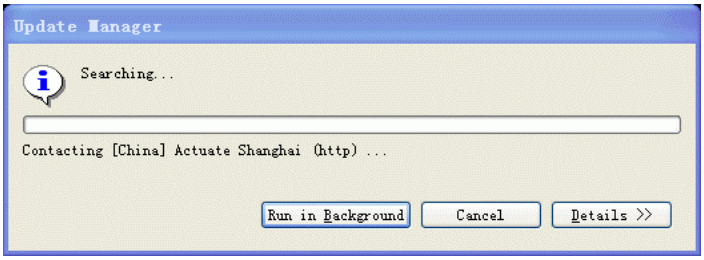

图 1-12　请求连接窗口

如果在图 1-12 中所连接的网站正确无误，则打开如图 1-13 所示窗口。

在图 1-13 中，选中站点 AA，在 Language Packs 子节点中选择 Simplified Chinese 选项，单击 Next 按钮。下面弹出的窗口，读者可以根据提示，一直单击【下一步】按钮即可。

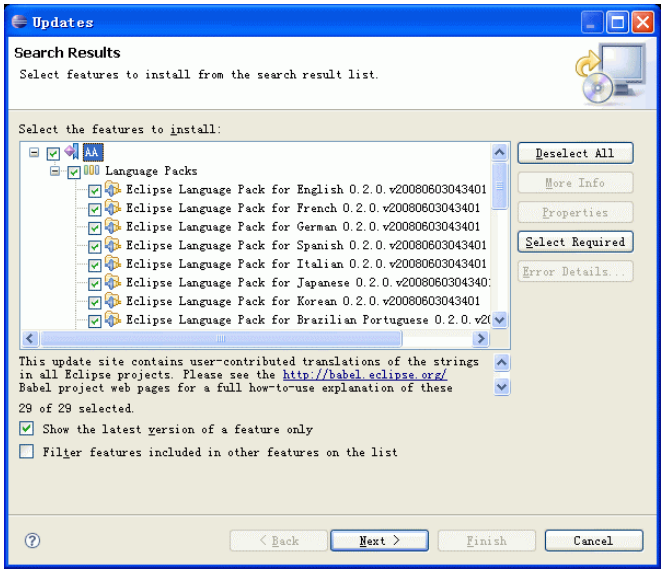

图 1-13　站点建立成功

在使用汉化后的 Eclipse 时，有的功能可能不支持，还是建议读者使用英文版。

1.3　Eclipse 窗口介绍

Eclipse 作为一种可视化开发工具，有自己独特的开发环境界面。下面来具体认识一下 Eclipse 界面的组成部分。图 1-14 所示为 Eclipse 的开发环境界面。

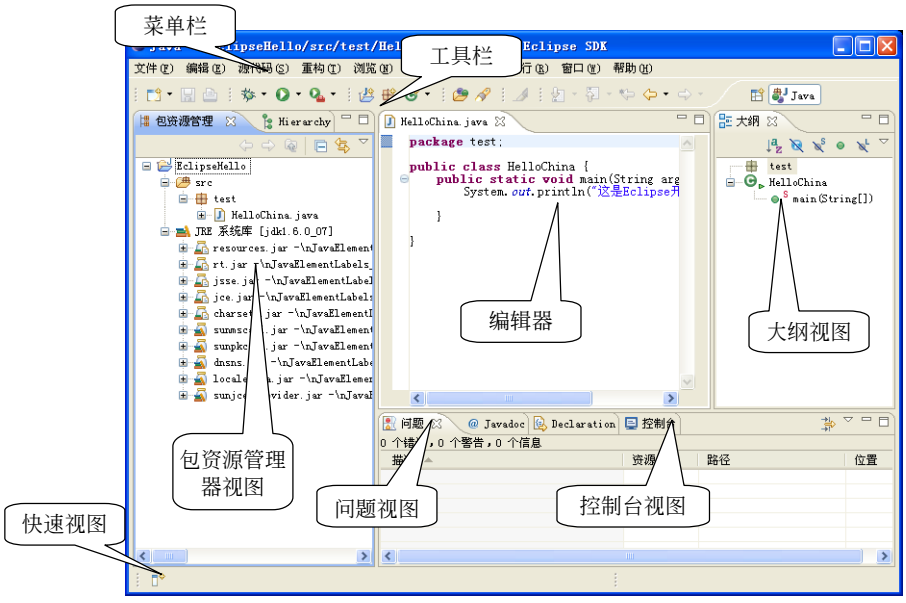

图 1-14　Eclipse 主窗口界面

要想理解 Eclipse 工作空间，需要了解以下 3 个概念：编辑器、视图和透视图。

- **编辑器（Editor）** 是用户打开文件后进行编辑的窗口。
- **视图（View）** 是方便浏览工作区信息的导航区。例如在图 1-15 中有【包资源管理器视图】，可以按包结构来浏览 Java 源文件。Eclipse 提供各种各样的视图，执行【窗口】|【视图】|【其他】命令来选择想要浏览的视图。

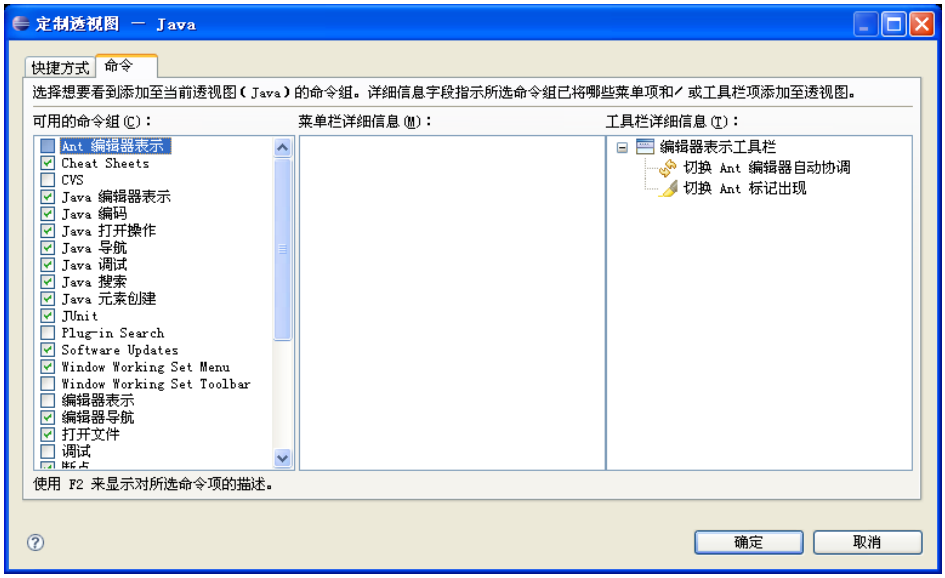

图 1-15　定制透视图

- **透视图（Perspective）** 用于管理视图和工作区布局。Eclipse 内置一些常用的透视图，执行【窗口】|【打开透视图】|【其他】命令来选择不同的透视图。另外，如果要想保存当前视图布局，通过执行【窗口】|【将透视图另存为】命令来保存。

在进行资源管理或者其他任务时，可能需要更多的功能支持。执行【窗口】|【定制透视图】命令，然后选择【命令】选项卡，如图 1-15 所示。在弹出的对话框中，可以看到很多的命令组，命令组提供了进行特定操作需要的功能。

如果用户想将定制的透视图恢复为最初的原始状态，也可以执行【窗口】|【定制透视图】命令，取消相关的命令组，或者可以使用【窗口】|【复位透视图】菜单。

1.4　文件结构

认识了 Eclipse 的工作环境，下面就来学习如何开发 Java 程序。包括如何创建类、如何编译类文件、如何导入编译时的类包以及如何运行程序等。

1.4.1　创建项目、包和类

一个 Java 项目工程包含用于构建 Java 程序的源代码和相关文件。可以用两种不同方法组

织 Java 项目，如下所示：

- **将项目用做源容器**　在这种组织方法中，所有 Java 包是直接在项目内创建的。默认情况下就选择这种组织方法。生成的 class 文件与 Java 源文件存储在一起。此种组织结构适用于简单的项目。
- **将源文件夹用做源容器**　在此项目组织方法中，包不是直接在项目内而是在源文件中创建，创建源文件作为项目的子目录，并在这些源文件中创建包。

对于比较复杂的项目，建议使用第 2 种方法。

1．创建 Java 项目

下面开始创建 Java 项目，步骤如下：

（1）打开 Eclipse 集成开发环境，执行【文件】|【新建】|【Java 项目】命令，打开如图 1-16 所示窗口。

图 1-16　创建 Java 项目

（2）在【项目名】文本域中，输入项目名称：HelloWorld；在【项目布局】选择框中可以选择项目的创建方式。

对于简单的项目，系统采用项目文件夹作为源文件或类文件的根目录，当然，用户也可以选择创建单独的源文件夹和输出文件夹。

（3）单击【完成】按钮，系统将自动打开 Java 透视图。在【包资源管理器视图】中，可以看到刚才创建的 HelloWorld 项目。另外，还可以看到层次结构视图和大纲视图，如图 1-17 所示。

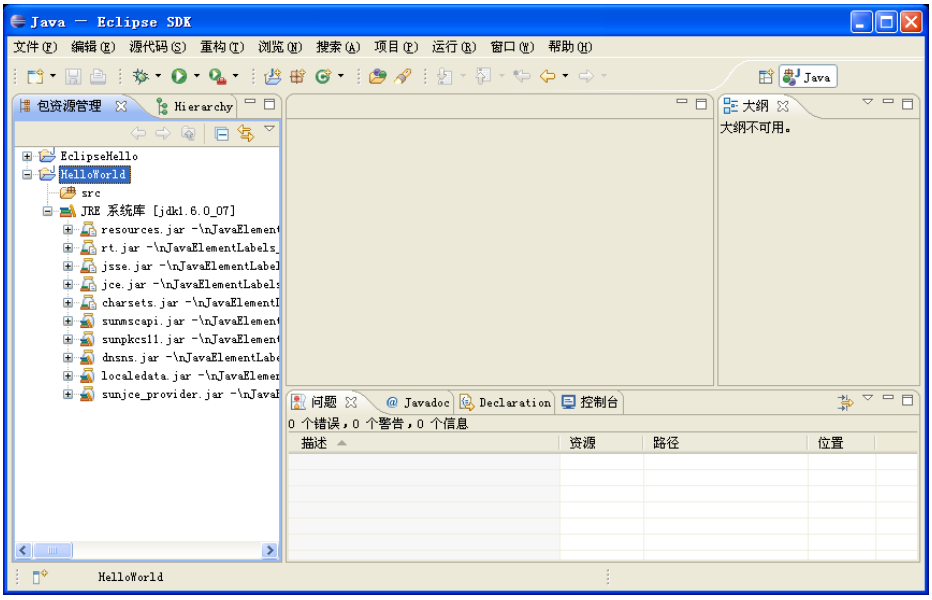

图 1-17　Java 项目透视图

默认条件下，Java 源文件会存储在 src 文件夹下面。

2．创建包

到目前为止，在工作空间中已经有了一个 Java 项目，当创建 Java 工具类或独立的 Java 应用程序时，可以在项目中为应用程序创建包和类，并且可以创建一个或者多个包，创建 Java 包的步骤如下所述。

（1）在【包资源管理】视图中，右击 HelloWorld 项目，在快捷菜单中执行【新建】|【包】命令，打开【新建 Java 包】对话框，如图 1-18 所示。

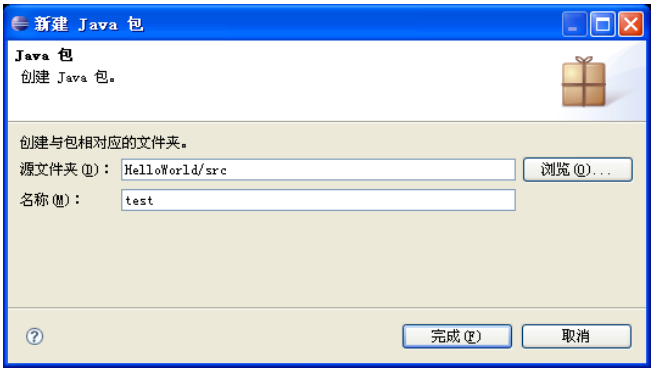

图 1-18　新建 Java 包对话框

（2）【源文件夹】文本域用来指定该包驻留在哪个容器中，可以输入路径或者单击【浏览】来查找容器。如果在选择创建新包时选择了文件夹，则该文件夹会作为新包的容器出现在【源文件夹】文本域，这里接受默认值。

（3）在【名称】文本域中，输入新包名称，这里输入 test。完成上述操作后，单击【完成】按钮。

3. 创建 Java 类

创建包之后，可以继续创建包中包含的 Java 类。创建 Java 类的步骤如下所述。

（1）在【包资源管理器】中，右击 test 包，在快捷菜单中执行【新建】|【类】命令，打开如图 1-19 所示窗口。

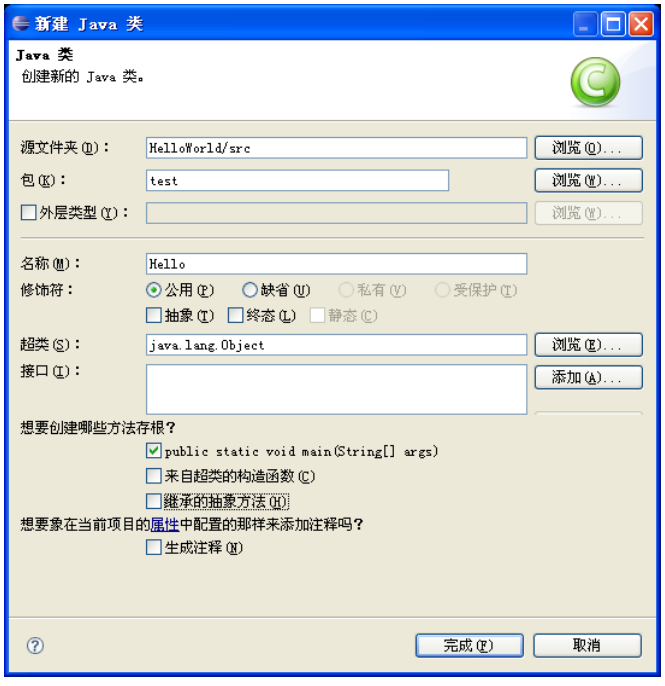

图 1-19　新建 Java 类对话框

上图中各个定义项的含义，如表 1-1 所示。

表 1-1　定义项含义

名称	说明
源文件夹	在上图中的源文件夹栏中，指定类所在项目对应的文件夹。通常与包所在的项目同名，但是可以是工作空间中的另一个 Java 源文件夹
包	在包的一栏中，可在工作空间的任何包中创建类。如果在创建类时选择了一个包，则包名就自动出现在栏中。也可以输入新的包名，在这种情况下，Eclipse 将创建该包
外层类型	这个复选框允许将这个类创建为嵌套类。如果启用，还需要提供外层类型名称
名称	在名称栏中输入 Java 类的名称。本例中为 Hello
修饰符	这些复选框和单选框按钮允许设置任意类型的修饰符：public、final、abstract 或者 default
超类	输入名字来指定超类，或者单击【浏览】按钮，从工作空间类的列表中选择一个类

续表

名称	说明
接口	单击【添加】按钮生成工作空间中接口的一个列表，从该列表中可以选择新建类必须实现的接口
Public staic void main(String[] args)	如果这是一个独立的类，则启用该复选框生成 main 方法
来自超类的构造函数	启用该复选框，向导生成与超类匹配的构造函数代码
继承的抽象方法	实现接口时，启用该复选框。向导将为接口上定义的方法生成存根（stub）
生成注释	表示在源文件中包含注释

（2）设置完毕以上选项，单击【完成】按钮。创建成功后的 Hello 类文件如图 1-20 所示。

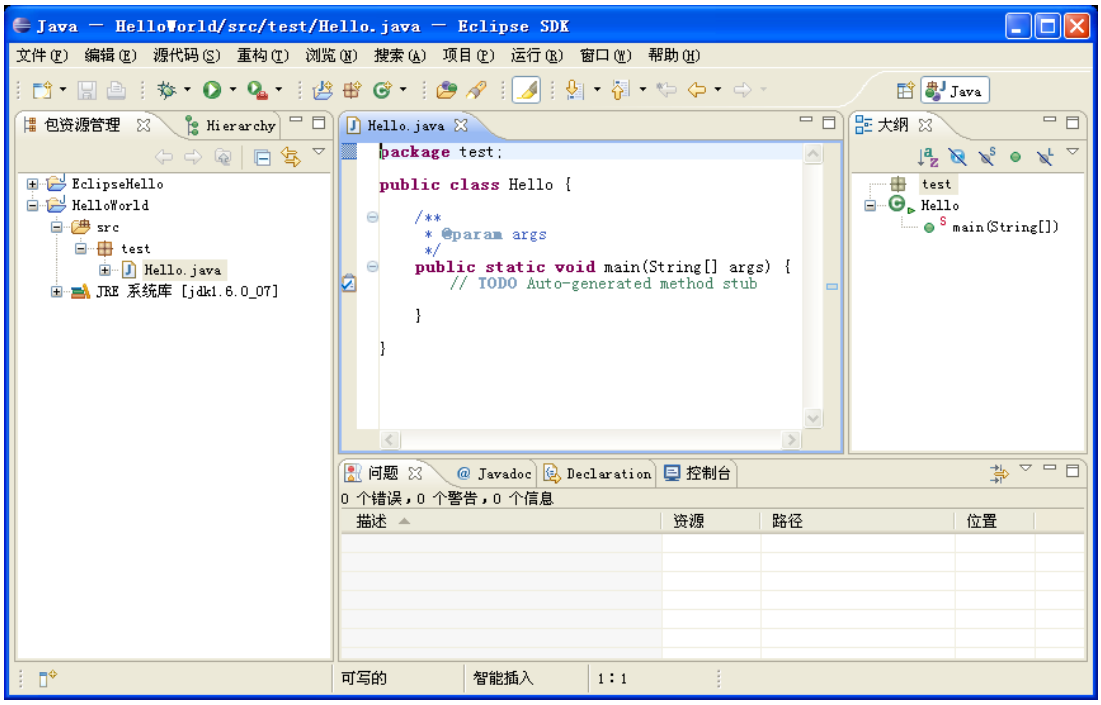

图 1-20　创建 Hello 类文件

1.4.2　导入项目使用的包

Eclipse 生成的项目文件会自动将 JRE 系统库包含进来，如图 1-17 所示。如果在项目中使用了其他 JAR 包的类，则需要先导入到该项目中。其步骤如下：执行【项目】|【属性】|【Java 构建路径】|【库】命令，打开如图 1-21 所示窗口。

在图 1-21 中，显示了该项目所使用的类库，例如这里已经包含了 JRE 系统库。导入类包通常是使用第三方提供的 JAR 包，例如常见的数据库连接的类包。

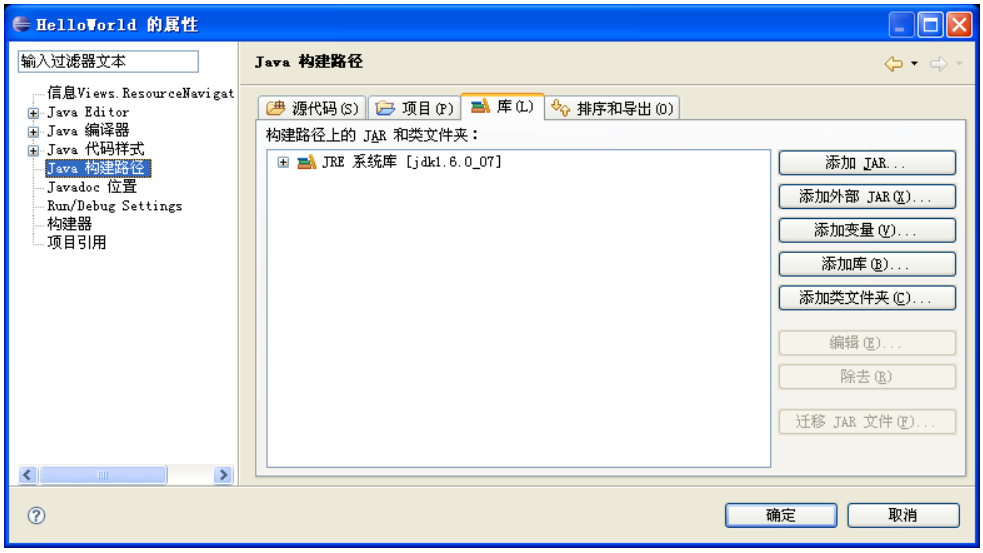

图 1-21　HelloWorld 项目库

例如项目在运行时需要连接 MySQL 数据库，此时就需要将 MySQL 数据库的 JDBC 驱动程序导入到项目中。假如该包存在于 E:下。此时单击图 1-21 中的【添加外部 JAR】按钮，打开如图 1-22 所示对话框。在对话框中选择要导入的 jar 包后，单击【打开】按钮即可。

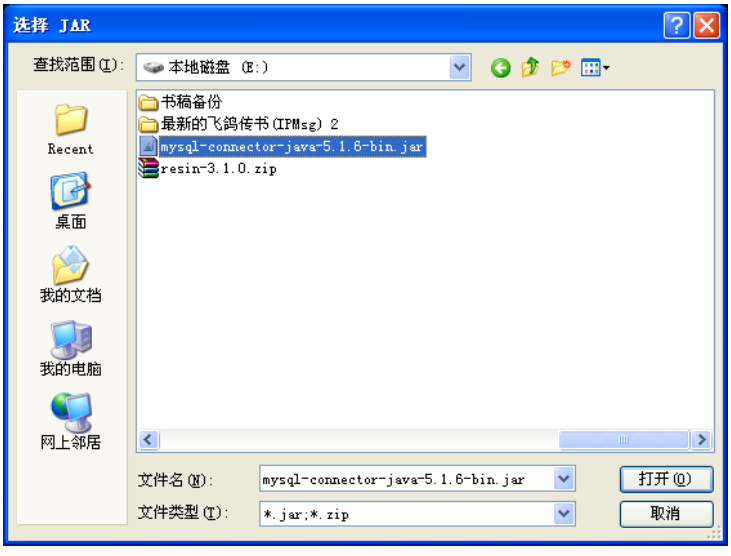

图 1-22　添加外部 JAR

添加完毕后，HelloWorld 属性对话框中会显示如图 1-23 所示情形。此时，单击【确定】按钮，在 HelloWorld 项目的【包资源管理器】中，显示添加的类库信息，如图 1-24 所示。

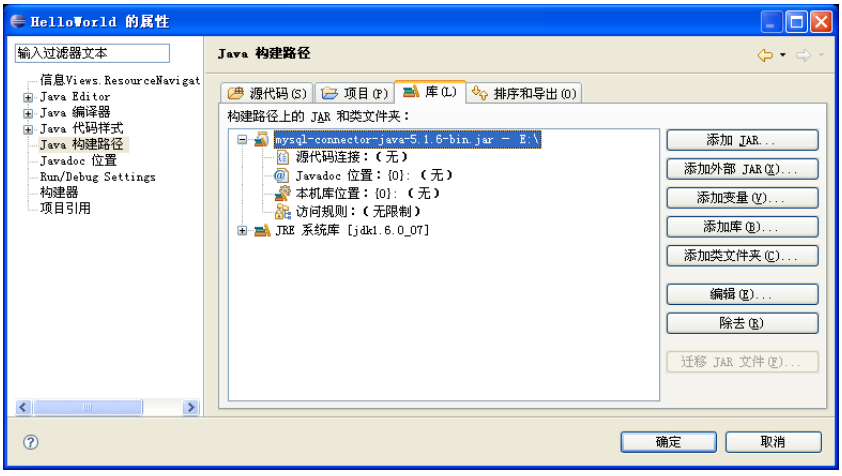

图 1-23　外部 JAR 显示

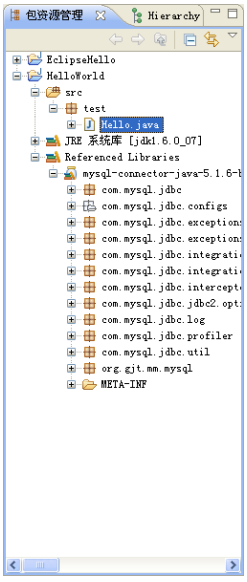

图 1-24　显示类库信息

1.4.3　设置编译方式和运行

Eclipse 提供一种很方便地编译文件的方式，即【自动构建】。也就是说，在保存.java 源文件时，直接将.java 文件编译成.class 文件，而不需要进行编译。

（1）如果查看该功能是否启用，单击【项目】菜单，确保【自动构建】菜单项为选中状态。

（2）如果此时未选中【自动构建】，那么就需要手动进行编译文件了。用户可以执行【全部构建】命令来编译所有的源文件。也可以执行【清理】命令，删除掉已经编译的.class 文件。

设置好编译方式之后，接下来介绍如何对运行程序参数进行设置。右击所要运行的文件，在快捷菜单中执行【运行方式】|【Java 应用程序】命令，可以直接运行 Java 程序。如果执行【运行方式】|【open 运行 Dialog】，将打开如图 1-25 所示对话框。

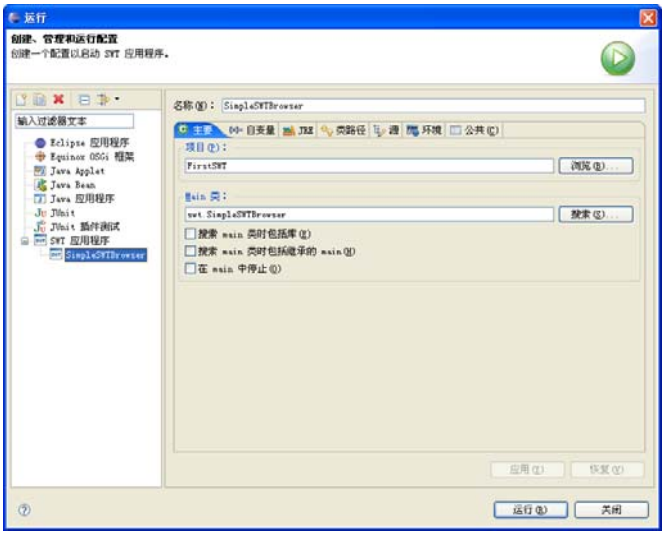

图 1-25　设置运行参数

在此界面可以为运行程序设置各种运行参数，包括自变量、类库、类路径以及环境变量等。

1.5 代码编辑功能

Eclipse 对编写 Java 代码提供很好的支持，即 Java 编辑器提供了语法高亮显示、格式化、折叠、内容辅助和代码模板等许多功能。它不断增长的可用重构和代码生成功能集合允许读者在更高的级别上操作代码，并自动化通常的代码密集型任务和易错任务。

1. 自定义格式代码

良好的组织代码是一个程序开发人员所要具备的最基本的素质，格式化的代码有益于其他开发人员的阅读，所以一定要养成良好的代码书写习惯。Eclipse 提供了格式化代码的功能，能够自动整理代码，使凌乱的代码变得整齐。

要想格式化一段代码，用户只需右击选中所需要格式化的代码，在快捷菜单中执行【源代码】|【格式】命令，此时会将选中的代码格式化。

如果用户想要按照自己的需求，设置格式化的方式，只要执行【窗口】|【首选项】命令，打开如图 1-26 所示的【首选项】窗口，在该窗口中选择【格式化程序】。

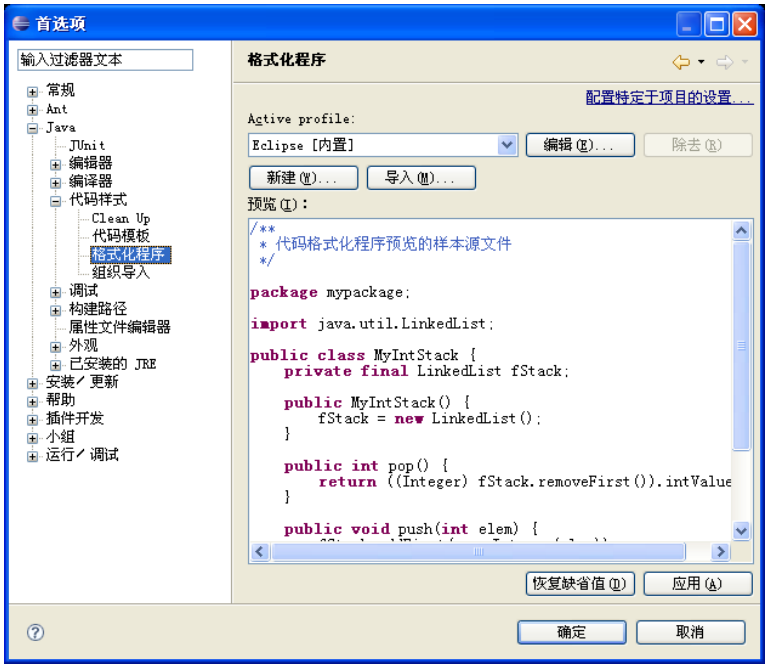

图 1-26　设置格式化程序

Eclipse 内置了 Eclipse、Java 约定和 Eclipse 2.1 三个格式化代码的模板，这三个模板不可以改变。用户如果要自定义代码模板，需要在这 3 个内置的模板上进行修改。自定义代码模板步骤如下。

（1）在图 1-26 中，单击【新建】按钮，在打开的窗口中，输入模板的名称并选择格式化设置文件。例如模板名称为 JavaStyle，选择 Eclipse，单击【确定】按钮，打开如图 1-27 所示窗口。

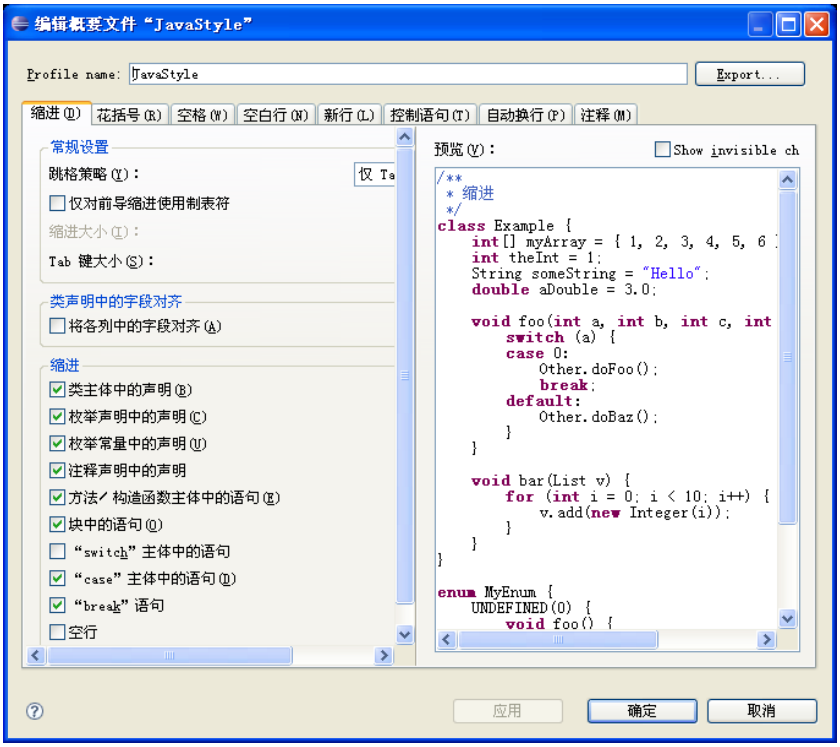

图 1-27　编辑概要文件 NewStyle 对话框

（2）用户可以根据自己的需要设置想要的样式，这里不再具体介绍。可以设置的选项很多，可以依照不同的喜好来设置。这样，如果以后再进行格式化代码的操作，就会按照刚才定义的代码格式进行格式化代码了。

如果在没有选中任何代码的情况下进行格式化，将会对该文件的所有代码进行格式化，也可以使用快捷键 Ctrl+Shift+F 进行相关操作。

2．自动生成 getter 和 setter 代码

JavaBean 是描述 Java 的软件组件模型，实质是 Java 类。在编写大型的 Java 项目中 JavaBean 组件的用处非常大，可以多次使用。Eclipse 对 JavaBean 提供了支持，JavaBean 类中有类的属性，也有设置和获取类属性的方法——getter 和 setter 方法。Eclipse 可以对类的属性自动生成 getter 和 setter 方法。例如，有这样一个类，代码如下：

```
package test;
public class BeanExample {
      private int bookid;
      private String bookname;
      private double bookprice;
}
```

这时选中类属性的第 2 行，然后右击，在快捷菜单中执行【源代码】|【生成 getter 和 setter 方法】命令，打开如图 1-28 所示窗口。

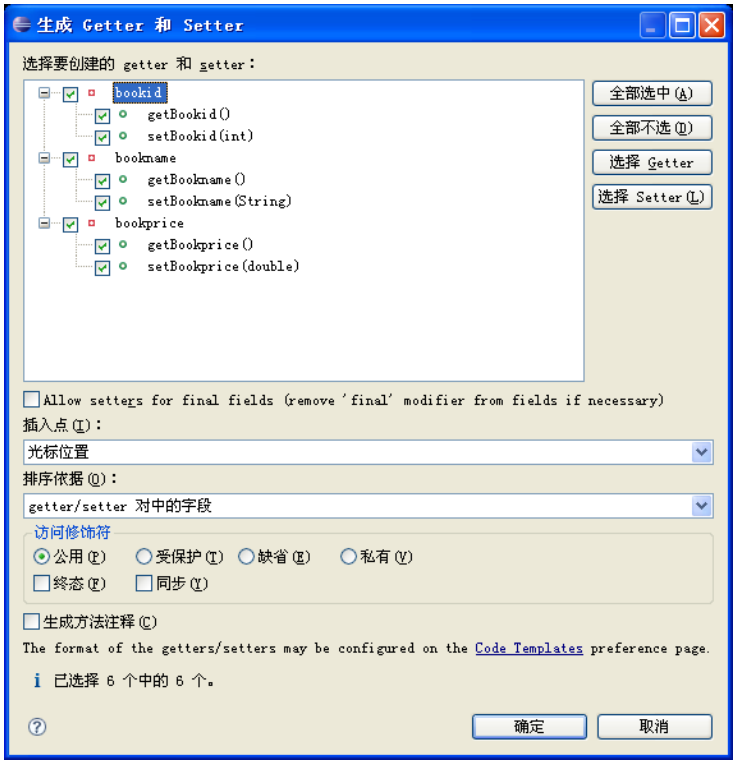

图 1-28　生成 Getter 和 Setter 对话框

单击【确定】按钮后，会自动生成如下代码：

```
package test;
public class BeanExample{
       private int bookid;
       private String bookname;
       private double bookprice;
       public int getBookid() {                    //自动生成的 bookid 的 get 方法
             return bookid;
       }
       public void setBookid(int bookid) {         //自动生成的 bookid 的 set 方法

             this.bookid = bookid;
       }
       public String getBookname() {               //自动生成的 bookname 的 get 方法
             return bookname;
       }
       public void setBookname(String bookname) {//自动生成的 bookname 的 set 方法
             this.bookname = bookname;
       }
       public double getBookprice() {              //自动生成的 bookprice 的 get 方法
             return bookprice;
       }
```

```
        public void setBookprice(double bookprice) { //自动生成的bookprice的set方法
            this.bookprice = bookprice;
        }
}
```


3. 添加注释

Eclipse 提供方便添加代码注释的方法，用户只需在代码处输入/**之后按 Enter 键，就自动完成了注释的标记。

同时，为了生成 JavaDoc 文件，通常会在注释中添加代码标记。这对 Eclipse 来说也是非常容易的，只需要输入@标记，Eclipse 会自动提示，此功能与编辑 Java 代码提示功能一样。如图 1-29 所示。

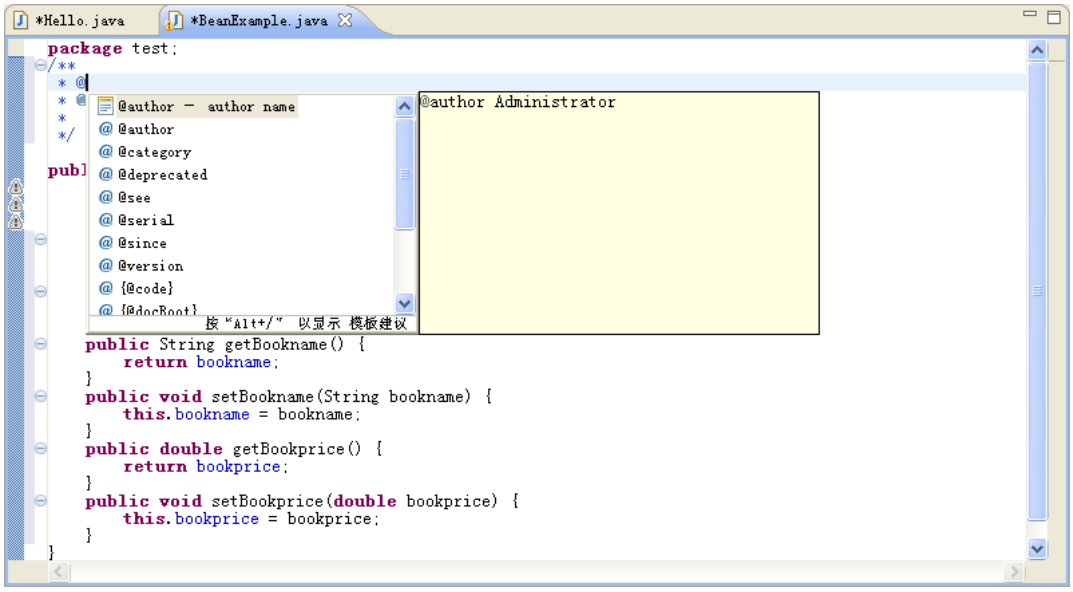

图 1-29　注释自动提示

1.6　文件查找

查找功能是一项基本功能，Eclipse 提供功能强大的查找功能。例如，在文件内部查找某个指定的字符串，此时，使用 Ctrl+F 快捷键，调出如图 1-30 所示的【查找/替换】对话框。

一般来说，许多软件都支持 Ctrl+F 快捷键进行查找。例如，如果在网页中查找某个字符串，也可使用该快捷键调出查找的对话框。

另外，Eclipse 还可以设定查找目录来进行搜索。执行【搜索】|【搜索】命令，打开如图 1-31 所示的【搜索】对话框，通过设置，就可以对不同类型的文件进行查找了。

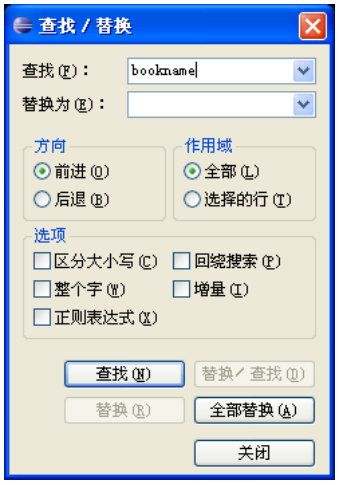

图 1-30 【查找/替换】对话框

图 1-31 【搜索】对话框

1.7 使用快捷键

快捷键是程序开发者所常用的，能大大提高开发效率。Eclipse 不仅可以使用内置的一些快捷键，也可以自定义快捷键。Eclipse 3.3 提供了快速显示快捷键功能，在编辑区使用快捷键 Ctrl+Shift+L 即可在编辑区上显示所有的快捷键及其说明提示，如图 1-32 所示。

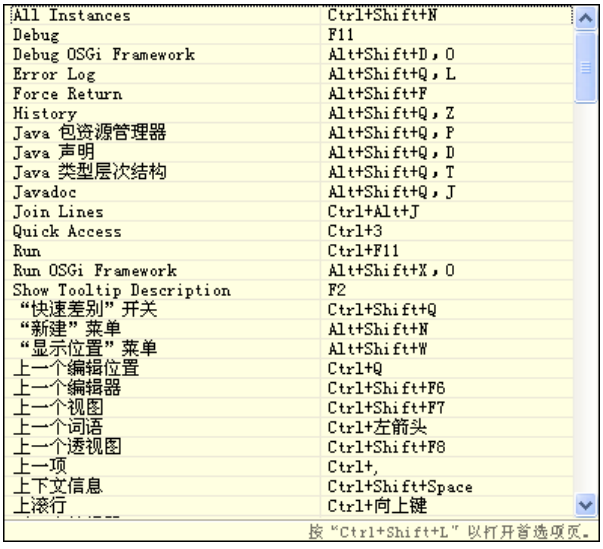

图 1-32 快捷键列表

使用系统内置的快捷键并不能满足用户的需求，用户有时还可按照自己的喜好来设置快捷键。Eclipse 也提供了自定义快捷键的功能。

执行【窗口】|【首选项】命令，在打开的窗口中选择【常规】|【键】选项，此时，更改快捷键的设置，如图 1-33 所示。使用快捷键也是开发软件中提高效率的一种方

式，用户应该记住一些常用功能的快捷键，例如格式化代码为 Ctrl+Shift+F、保存文件为 Ctrl+S 等。

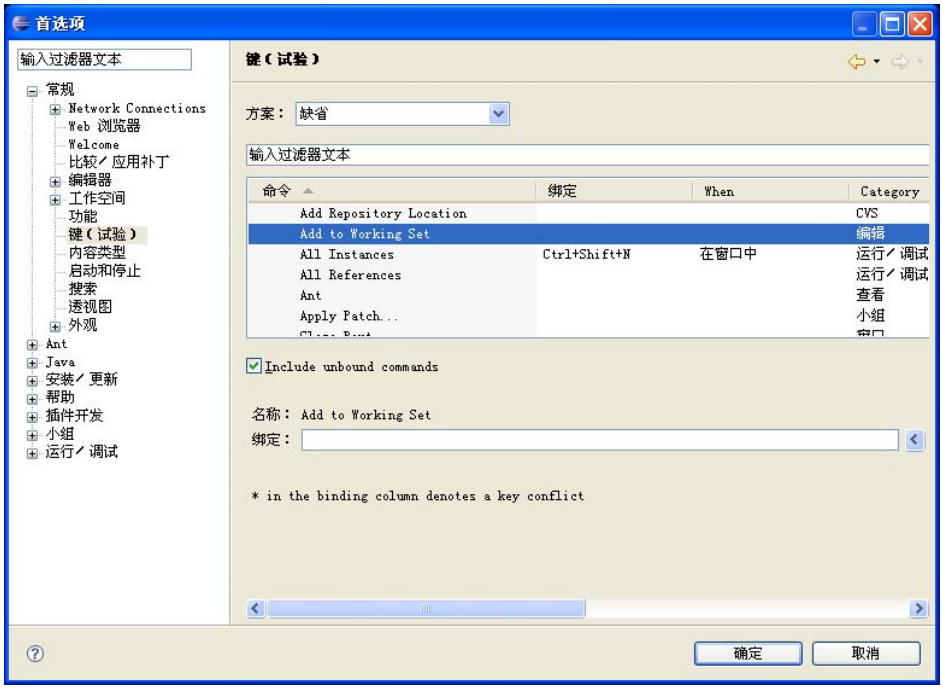

图 1-33　快捷键设置

1.8　创建 Eclipse 例子

Java 开发工具（JDT）是随 Eclipse 平台一起交付的插件。JDT 与其他插件一样是对工作台的一组扩展，它允许用户编辑、编译和运行 Java 程序。在本节将使用 JDT 创建一个简单的 Java 程序。在 Eclipse 工作区中，对任何程序都是以项目为单位进行管理的。项目也是工作区使用的最大结构化单元，项目中包含文件夹和文件，并且可以打开、关闭或构造文件夹和文件。

1. 创建 Java 项目

新建 Java 项目的操作步骤如下所述。

（1）从菜单栏中执行【文件】|【新建】|【Java 项目】命令，打开如图 1-34 所示窗口。在该对话框中输入创建的项目名 EclipseExample。关于项目所用的 JRE 和项目布局可以采用默认设置。

（2）单击【完成】按钮，在【包资源管理器】中，可以看到名为 EclipseExample 的 Java 项目，如图 1-35 所示。在此处用户可以浏览该项目中的 Java 元素和为该项目所配置的 JRE 文件。

图 1-34　新建 Java 项目

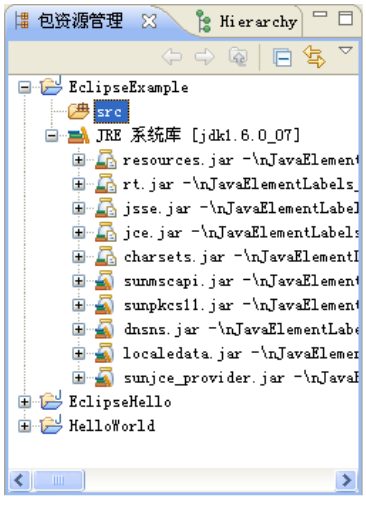

图 1-35　Java 项目所使用的 JRE

注意

Java 元素包括 Java 项目、编译单元、类文件、类型、方法和字段。到目前为止，新创建的项目仍然为一个空项目，内部不包含任何文件。

2. 创建 Java 类文件

下面创建一个 Java 类文件，新建 Java 类文件的操作步骤如下所述。

（1）右击选中创建的 EclipseExample 项目，在快捷菜单中执行【新建】|【Java 类】命令，打开如图 1-36 所示的窗口。

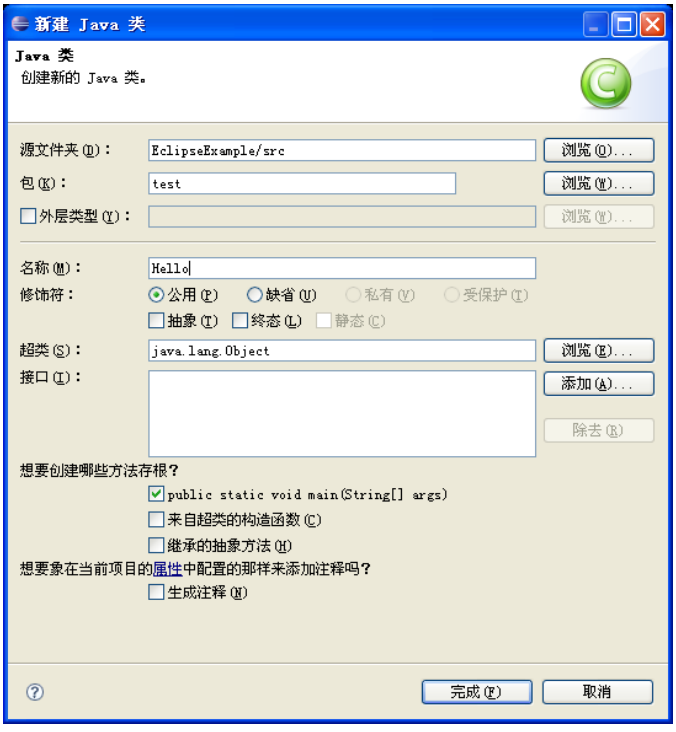

图 1-36　新建 Java 类对话框

（2）在图 1-36 中输入类的名称：Hello，设置完毕，单击【完成】按钮，打开如图 1-37 所示窗口，在该窗口中会自动打开一个 Java 编辑窗口。

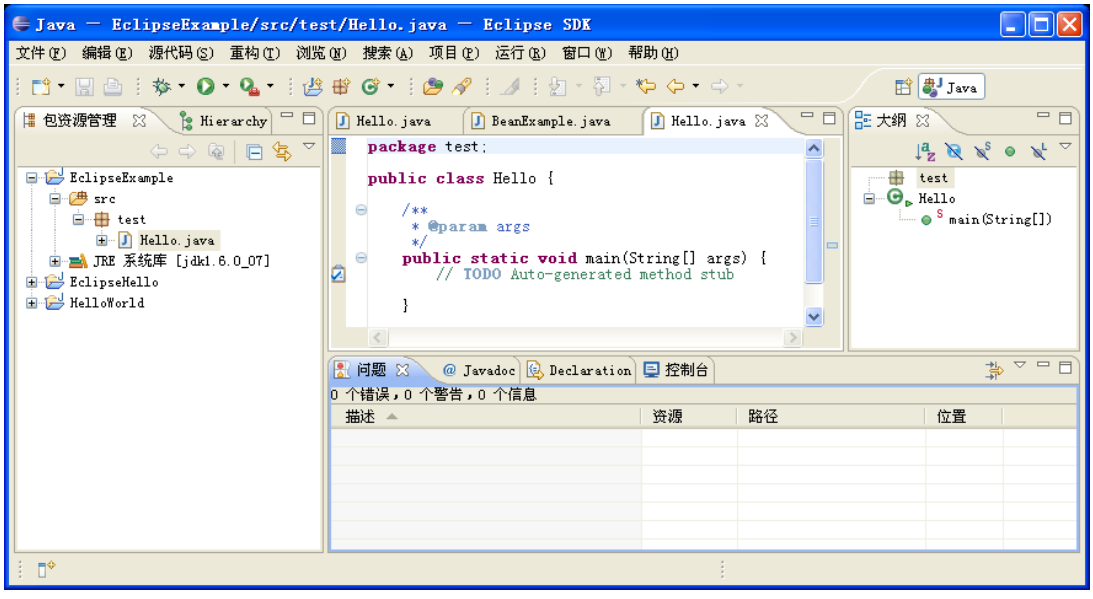

图 1-37　Java 编辑窗口

现在就可以在 Java 编辑器中编辑程序。代码编写完成后，Hello 类的完整代码如下所示：

代码 1.1　第一个 Eclipse 程序：Hello.java

```
package test;
public class Hello {
     public static void main(String[]args) {
System.out.println("Eclipse 是一个功能强大的开发工具");     //输出一个字符串
      }
}
```

这样完成了第一个项目的编程工作。单击工具栏上的保存按钮，将新创建 Hello 类保存到磁盘，保存时也会编译该类。现在就可以运行该程序了。

3. 运行

单击工具栏上的运行按钮 右边的箭头会弹出下拉菜单，在下拉菜单中执行【运行方式】|【Java 应用程序】命令来运行程序。在 Java 编辑器下方的【控制台】标签页中将显示输出的文本信息，如图 1-38 所示。

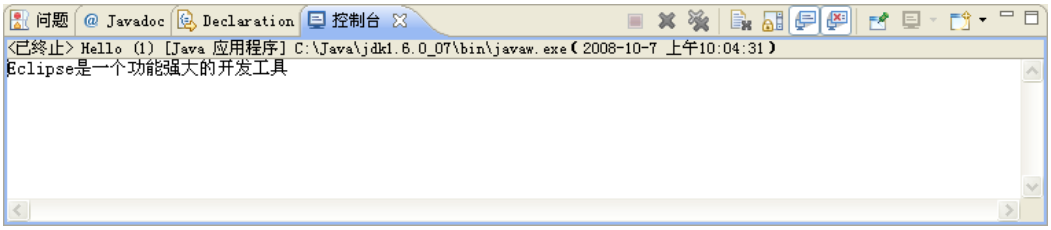

图 1-38　Java 程序运行结果