第2章　Java程序设计环境

▲ 安装Java 开发工具箱
▲ 选择开发环境
▲ 使用命令行工具
▲ 使用集成开发环境
▲ 运行图形化应用程序
▲ 建立并运行applet

本章主要介绍如何安装Java开发工具箱（JDK）以及如何编译和运行各种类型的程序：控制台程序、图形化应用程序以及applet应用程序。运行JDK的方法是在shell窗口中键入命令行。然而，很多程序员更喜欢使用集成开发环境。为此，将在稍后介绍如何使用免费的开发环境编译和运行Java程序。尽管学起来很容易，但集成开发环境需要吞噬大量资源，在编写小型程序时会给人带来烦恼。作为折中方案，再介绍一下如何调用Java编译器并运行Java程序的文本编辑器。一旦掌握了本章的技术，并选定了自己的开发工具，就可以学习第3章，开始研究Java程序设计语言。

2.1　安装Java开发工具箱

Sun Microsystems公司为Solaris、Linux和Windows提供了Java开发工具箱（JDK）的最新、最完整的版本。用于Macintosh和一些其他平台的版本仍处于各种不同的开发状态中，不过，这些版本都是由相应平台的开发商授权并分发的。

注释：有些Linux的发行版已经预先打包了JDK。例如，在Ubuntu中，可以用apt-get或Synaptic GUI安装sun-java6-jdk包来安装JDK。

2.1.1　下载JDK

要想下载Java开发程序包，必须到Sun网站进行搜寻，并且在得到所需的软件之前必须弄清大量的专业术语。请看表2-1。

表2-1　Java术语

术 语 名	缩　写	解　释
Java Development Kit	JDK	编写Java程序的程序员使用的软件
Java Runtime Environment	JRE	运行Java程序的用户使用的软件
Standard Edition	SE	用于桌面或简单的服务器应用的Java平台
Enterprise Edition	EE	用于复杂的服务器应用的Java平台
Micro Edition	ME	用于微型手机cell phone和其他小型设备的Java平台
Java 2	J2	一个过时的术语，用于描述1998年~2006年之间的Java版本
Software Development Kit	SDK	一个过时的术语，用于描述1998年~2006年之间的JDK
Update	u	Sun的术语，用于发布修改的bug
NetBeans	—	Sun的集成开发环境

JDK是Java Development Kit的缩写。有点混乱的地方是：工具箱的版本1.2~版本1.4被称为Java SDK（Software Development Kit）。在某些场合下，还可以看到这些过时的术语。另外，还有Java 运行时环境（JRE），它包含虚拟机但不包含编译器。这并不是开发者所想要的环境，而是专门为不需要编译器的用户而设计。

还有，随处可见的Java SE，相对于Java EE（Enterprise Edition）和Java ME（Micro Edition），它是Java的标准版。

Java 2这种提法始于1998年。当时Sun公司的销售人员感觉通过增加小数点后面的数值改变版本号并没有反映出JDK1.2 的重大改进。但是，由于在发布之后才意识到这个问题，所以决定将开发工具箱的版本号仍然沿用1.2，接下来的版本就是1.3、1.4和5.0。但是，Java平台被重新命名为Java 2。因此，就有了Java 2 Standard Edition Software Development Kit 的5.0版，即J2SE SDK 5.0。

对于工程师来说，所有这一切都可能会引起困惑，这正是没有将其投入市场的原因。2006年，日趋完善的Java SE开始流行。无意义的Java 2被遗弃，Java当前的标准版本被称为Java SE 6。偶尔还会看到使用1.5版本和1.6版本，但这些只是5.0版本和6版本的同义词。

最后，当Sun为解决一些紧急问题做出了某些微小的版本改变时，将其称为更新。例如：对Java SE 6开发包做出的第一次更新，正式称为JDK 6u1，内部的版本号为1.6.0_01。

如果使用Solaris、Linux或Windows，就应该从http://java.sun.com/javase下载Java软件开发工具箱。找到版本6或后续的版本，并选择自己的平台。如果软件被称为“更新”也不必担心。更新将捆绑包含了整个JDK的最新版本。

Sun公司曾经制作过将Java开发工具箱和集成开发环境捆绑在一起的产品。其中的集成开发环境，在不同时期，被命名为不同的名字，例如，Forte、Sun ONE Studio、Sun Java Studio和NetBeans。我们无法知道每个人在登录Sun网站时，市场正在热销什么。这里，建议大家只安装Java开发工具箱。如果最终决定使用Sun的集成开发环境，就可以从http://netbeans.org下载。

下载JDK之后，随后的安装过程请参看不同平台的安装指南。在编写本书时，可以在网站http://java.sun.com/javase/6/webnotes/install/index.html上搜寻到。

只有Java的安装过程和编译命令与系统有关。一旦安装并运行了Java，本书阐述的所有内容都适用。与系统无关性是Java的一个主要优势。

注释：安装过程提供了包含JDK版本号（如jdk1.6.0）的默认的安装路径。这似乎有些烦人，但是，应该重视版本号，它会给安装新版JDK的测试带来便利。

在Windows环境下，强烈建议不要接受带空格的默认路径名，如：c:\ProgromFiles\jdk1.6.0。应该将Progrom Files部分删掉。

在本书中，使用的安装路径是jdk。例如：当引用jdk/bin目录时，意味着引用的是/usr/local/jdk1.6.0/bin或c:\ jdk1.6.0\bin。

2.1.2 设置执行路径

在完成了JDK的安装之后，还需要执行另外一个步骤：把jdk/bin目录添加到执行路径中。所谓执行路径是指操作系统搜索本地可执行文件的目录列表。对于不同的操作系统，这个步骤

的操作过程有所不同。

- 在UNIX（包括Solaris和Linux）环境下，编辑执行路径的过程与所使用的shell有关。如果使用的是C shell（Solaris的默认选择），在 ~/.cshrc文件的末尾就要添加：

```
set path=(/usr/local/jdk/bin $path)
```

如果使用的是Bourne Again shell（Linux的默认选择），在 ~/.bashrc 文件或~/.bash_profile文件的末尾就要添加：

```
export PATH=/usr/local/jdk/bin:$PATH
```

- 在Windows下，以管理员身份登录。启动Control Panel，切换到Classic View，并选择System图标。在Windows NT/2000/XP中，立即会看到System Properties对话框。在Vista中，需要选择Advanced系统设置（如图2-1所示）。在System Properties对话框中，点击Advanced标签，然后点击Environment Variables按钮。滚动System Variables窗口直到找到变量名path为止。点击Edit按钮（如图2-2所示）。将jdk\bin目录添加到路径的开始处，用分号将新条目隔开，如下所示：

```
c:\jdk\bin;other stuff
```

将这个设置保存起来，打开的任何控制窗口都会有正确的路径。

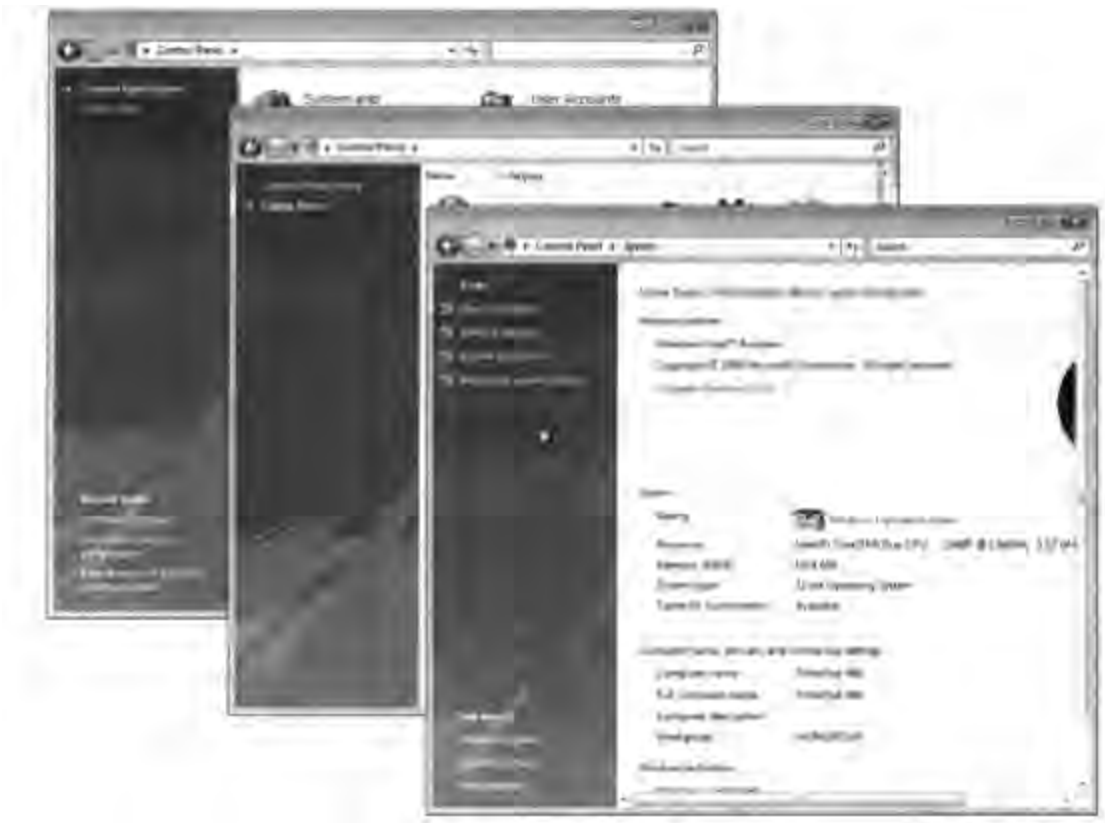

图2-1 在Windows Vista中启动的System Properties对话框

下面是测试上述设置是否正确的方法：打开一个shell窗口，键入：

```
java -version
```

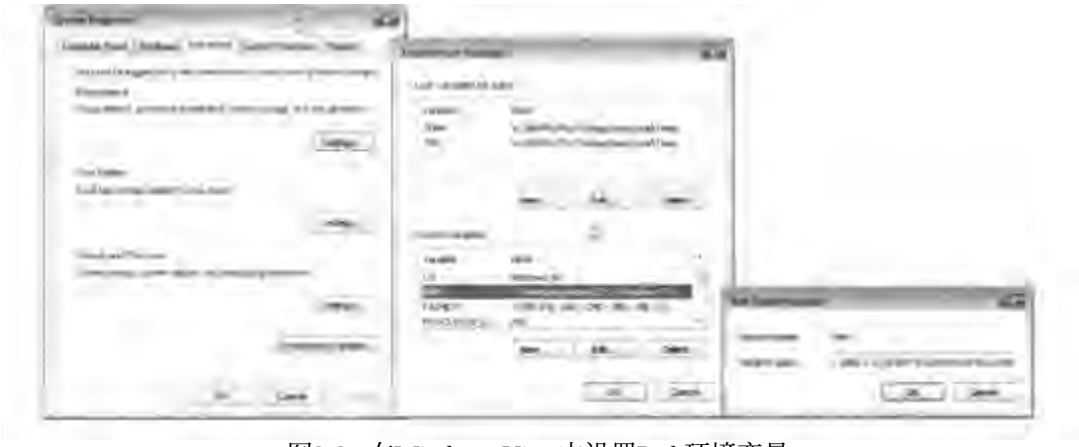

图2-2 在Windows Vista中设置Path环境变量

然后，按ENTER键。应该能够看到下面的显示信息：

```
java version "1.6.0_01"
Java(TM) SE Runtime Environment (build 1.6.0_01-b06)
Java HotSpot(TM) Client VM (build 1.6.0_01-b06, mixed mode, sharing)
```

如果看到的是“ java:commond not found ”、或“ The name specified is not recogonized as an internel or external commond, operable program or batch file ”，则需要回到前面，重新检查整个的安装过程。

✔ 注释：在Windows中，按照下面的命令打开shell窗口。如果使用Windows NT/2000/XP环境，那么在开始菜单中选择Run选项，并键入cmd。在Vista中，只在开始菜单中的Start Search中输入cmd，再按下ENTER键就会出现一个shell窗口。

如果曾经没有接触过这些内容，建议学习一下有关命令行的基础教程。很多大学的计算机科学系在网上都有相关的教程，例如http://www.cs.sjsu.edu/facult/horstman/CS46A/windows/tutorial. html。

2.1.3 安装源代码库和文档

库源文件在JDK中以一个压缩文件src.zip 的形式发布，必须将其解压缩后才能够访问源代码。这里强烈地建议按照下面所述的步骤进行操作。很简单：

1）确保JDK已经安装，并且jdk/bin目录在执行路径中。

2）打开shell窗口。

3）进入jdk目录（例如：cd /usr/local/jdk1.6.0 或 cd c:\jdk1.6.0）。

4）建立一个子目录src

```
mkdir src
cd src
```

5）执行命令：

```
jar xvf ../src.zip
```

（或者在Windows中执行jar xvf ..\src.zip。）

提示：在src.zip文件中包含了所有公共类库的源代码。要想获得更多的源代码（例如：编译器、虚拟机、本地方法以及私有辅助类），请访问网站：http://download.java.net/jdk6。

文档包含在一个压缩文件中，它是一个独立于JDK的压缩文件。可以直接从网站http://java.sun.com/javase/downloads下载获得这个文档。操作步骤如下：

1）确认JDK已经安装，并且jdk/bin目录在执行路径上。

2）下载文档压缩文件并将其存放在jdk目录下。这个文件名为jdk-*version*-doc.zip，其中的*version*表示版本号，例如，6。

3）打开一个shell窗口。

4）进入jdk目录。

5）执行命令：

```
jar xvf jdk-version-doc.zip
```

其中*version*是相应的版本号。

2.1.4 安装本书中的示例

读者可以安装本书中的示例程序。这些程序可以从http://horstmann.com/corejava下载，它们都打包在corejava.zip文件中。应该将它们解压到一个单独的文件夹中，建议将文件夹命名为CoreJavaBook。需要执行下列步骤：

1）确保JDK已经安装，并且jdk/bin目录在执行路径中。

2）建立目录CoreJavaBook。

3）将corejava.zip下载到这个目录下。

4）打开一个shell窗口。

5）进入CoreJavaBook目录。

6）执行命令：

```
jar xvf corejava.zip
```

2.1.5 导航Java目录

在学习Java的过程中，经常需要查看Java源文件。当然，也会频繁地使用类库文档。表2-2显示了JDK目录树。

表2-2 JDK目录树

目录结构	描 述
jdk	（名字可能不同，例如：jdk5.0）
├── *bin*	编译器和工具
├── *demo*	演示
├── *docs*	HTML格式的类库文档（展开j2sdkversion-doc.zip之后）
├── *include*	用于编译本地方法的文件（参看卷II）
├── *jre*	Java运行环境文件
├── *lib*	类库文件
└── *src*	类库源文件（展开src.zip之后）

就学习Java而言，docs和src是两个最有用的子目录。docs目录包含了HTML格式的类库文档，可以使用任何浏览器（如：Netscape）查看这些文档。

提示：在浏览器中设置一个指向docs/api/index.html书签。使用Java平台时经常需要查看这一页的内容。

src目录包含了Java类库中公共部分的源代码。当对Java熟悉到一定程度时，可能会感到本书以及联机文档已经无法提供所需的信息了。那时，应该深入研究Java的源代码。请放心，只要深入地研究源代码就一定会搞清楚类库函数的真正功能。例如，如果对System类的内部工作感到好奇，可以查看src/java/lang/System.java。

2.2 选择开发环境

如果在此之前使用Microsoft Visual Studio 编写过程序，那么就会习惯于带有内嵌的文本编辑器、用于编译和运行程序的菜单，以及配有集成调试器的开发环境。基本的JDK全然没有这些功能。利用它完成每一项任务时都要在shell窗口中键入命令。这些听起来很麻烦，但只是一个基本的技能。第一次安装Java时，可能希望在安装开发环境之前检测一下安装的Java是否正确。此外，还可以通过执行一些基本的操作步骤，加深对开发环境幕后工作的理解。

然而，在掌握了编译和运行Java程序的基本步骤之后，你就想要使用专业的开发环境了。在过去的10年中，这些环境变得功能非常强大，操作也非常方便，以至于不选用它们将是极其不理智的。使用Eclipse和NetBeans这两个免费的开发环境是一个很好的选择。尽管NetBeans 发展的速度比较快，但在本章中，还是打算介绍如何使用Eclipse，这是因为Eclipse要比NetBeans更加灵活一些。当然，如果青睐使用其他的开发环境，同样也可以完成本书的所有程序。

在过去，推荐使用文本编辑器编写简单的程序，如：Emacs、JEdit或者TextPad。现在不会再这样推荐了，因为集成开发环境非常快捷、方便。

总之，应当了解如何使用基本的JDK工具，这样才会感觉使用集成开发环境是一种享受。

2.3 使用命令行工具

首先介绍较难的一种方法：通过命令行编译并运行Java程序。

1）打开一个shell窗口。

2）进入CoreJavaBook/v1ch02/Welcome目录（CoreJavaBook是安装本书示例源代码的目录，请参看“安装本书中的示例”一节）。

3）键入下面的命令：

```
javac Welcome.java
java Welcome
```

然后，将会在shell窗口中看到图2-3所示的输出。

祝贺你！已经编译并运行了第一个Java程序。

那么，刚才都进行了哪些操作呢？javac程序是一个Java编译器。它将文件Welcom.java编译成Welcom.class，并发送到Java虚拟机。虚拟机执行编译器存放在class文件中的字节码。

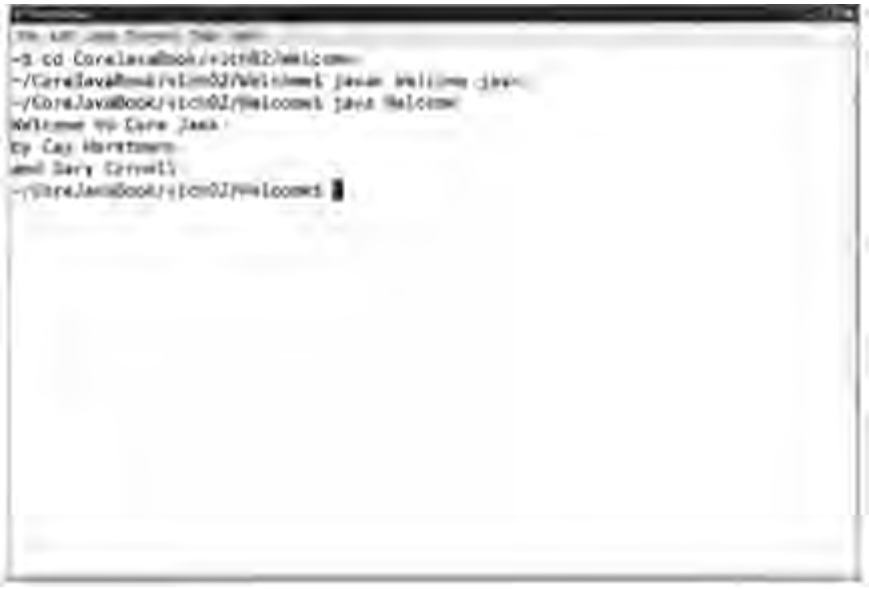

图2-3 编译并运行Welcom.java

注释：如果得到了下面这行语句的错误消息

```
for (String g : greeting)
```

就可能是使用了Java编译器的旧版本。Java SE 5.0为Java程序设计语言增加了一些非常令人欢快的新特性，在本书中，使用了这些特性。

如果使用的是Java旧版本，则需要将上面的循环语句改写成下列形式：

```
for (int i = 0; i < greeting.length; i++)
   System.out.println(greeting[i]);
```

Welcome程序非常简单。其功能只是向控制台输出了一条消息。看到例2-1的程序代码可能非常满意。(在下一章中，将解释它是如何工作的。)

例2-1 Welcome.java

```
1. /**
2.  * This program displays a greeting from the authors.
3.  * @version 1.20 2004-02-28
4.  * @author Cay Horstmann
5.  */
6. public class Welcome
7. {
8.    public static void main(String[] args)
9.    {
10.      String[] greeting = new String[3];
11.      greeting[0] = "Welcome to Core Java";
12.      greeting[1] = "by Cay Horstmann";
13.      greeting[2] = "and Gary Cornell";
14.
15.      for (String g : greeting)
16.         System.out.println(g);
17.   }
18. }
```

疑难解答提示

在使用可视化开发环境的年代，许多程序员对于在shell窗口中运行程序已经很生疏了。常常会出错很多错误，最终导致令人沮丧的结果。

一定要注意以下几点：

- 如果手工地输入源程序，一定要注意大小写。尤其是类名为Welcome，而不是welcome或WELCOME。
- 编译时需要提供一个文件名（Welcome.java），而运行时，只需要指定类名（Welcome），不要带扩展名.java或.class。
- 如果看到诸如Bad command or file name或javac:command not found这类消息，就要返回去检查一下安装是否出现了问题，特别是执行路径的设置。
- 如果javac报告了一个错误“ cannot read：Welcome.java ”，就应该检查一下目录中是否存在这个文件。

在UNIX环境下，检查Welcome.java是否以正确的大写字母开头。在Windows环境下，使用shell命令dir，而不要使用图形浏览器工具。有些文本编辑器（特别是Notepad）在每个文件名后面要添加扩展名.txt。如果使用Notepad编辑Welcome.java 就会存为Welcome.java.txt。对于默认的Windows设置，浏览器与Notepad都隐含.txt的扩展名，这是因为这个扩展名属于“ 已知的文件类型 ”。此时，需要重新命名这个文件，使用shell命令ren，或是另存一次，给文件名加一对双引号，如：“ Welcome.java ”。

- 如果运行程序之后，收到错误消息java.lang.NoClassDefFoundError，就应该仔细地检查一下类名。

如果收到关于welcome（w为小写）的错误消息，就应该重新发出带有大写W的命令：java Welcome。记住，Java区分大小写。

如果收到有关Welcome/java的错误信息，就是错误地键入了java Welcome.java，应该重新输入命令java Welcome。

- 如果键入java Welcome，而虚拟机没有找到Welcome类，就应该检查一下是否系统的CLASSPATH环境变量被别人更改了（将这个变量设置为全局并不是一个提倡的做法，然而，Windows中有些编写很差的软件安装程序就是这样做的）。可以在当前的shell窗口中键入下列命令，临时地取消CLASSPATH环境变量的设置：

```
set CLASSPATH=
```

这个命令应用于使用C shell的 Windows和UNIX/Linux环境下。在使用Bourne/bash shell 的UNIX/Linux环境下需要使用：

```
export CLASSPATH=
```

- 如果收到的错误信息是有关新语言结构的问题，就需要确认一下当前使用的编译器是否支持Java SE 5.0。
- 如果程序中的错误太多，所有的错误消息就会飞快地闪过。编译器将会把这些错误消息发送到标准错误流上。如果消息超过一屏，就很难看清楚它们了。可以使用shell的操作符2>将这些错误重定向到一个文件中：

```
javac MyProg.java 2> errors.txt
```

提示：在http://java.sun.com/docs/books/tutorial/getStarted/cupojava/上有一个很好的教程。其中提到了一些初学者经常容易犯的错误。

2.4 使用集成开发环境

本节将介绍如何使用Eclipse编译一个程序。Eclipse是一个可以从网站http://eclipse.org上免费下载得到的集成开发环境。Eclipse是采用java编写的，然而，由于所使用的是非标准视窗类库，所以，不如Java那样具有可移植性。尽管如此，这个开发环境已经拥有可以应用在Linux、Mac OS X、Solaris以及Windows的版本了。

还有一些使用比较普遍的IDE，但就目前而言，Eclipse是最通用的。使用步骤如下所示：

1）启动Eclipse之后，从菜单选择File→New Project。

2）从向导对话框中选择Java Project（如图2-4所示）。这些是Eclipse 3.2的屏幕画面。如果使用的Eclipse版本稍有差异，不必介意。

图2-4 Eclipse中的“New Project”对话框

3）点击Next按钮，键入工程名Welcome，并输入包含Welcome.java的完整路径名。如图2-5所示。

4）确保没有选定标有Create project in workspace的选项。

5）点击Finish按钮。这个工程已经创建完成了。

6）点击在工程窗口左边窗格中的三角，并打开它。然后，点击旁边的Default package三角，再双击Welcome.java。现在应该看到带有程序代码的窗口了（如图2-6所示）。

7）用鼠标右键点击最左侧窗格中的工程名（Welcome），选择Run→Run As→Java Application。可以看到：这个窗口底部出现了一个输出窗口。在这个输出窗口中显示了程序的输出结构（如图2-7所示）。

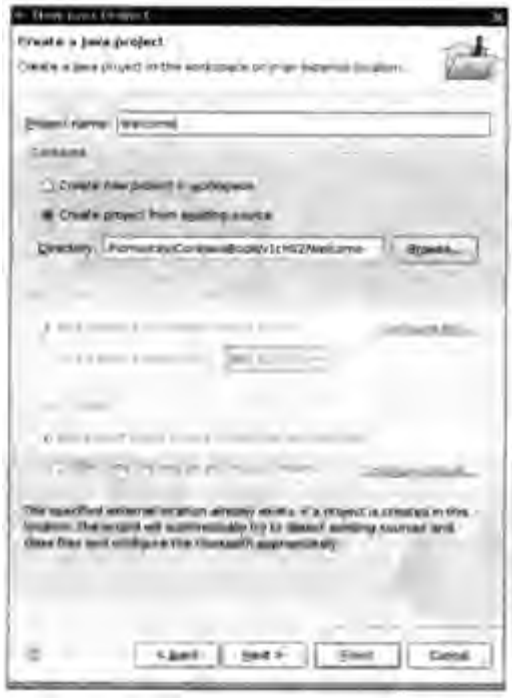

图2-5 配置Eclipse工程

定位编译错误

可以肯定，这个程序没有出现输入错误或bug（毕竟，这段代码只有几行）。为了说明问题，假定在代码中不小心出现了录入错误（或许语法错误）。试着将原来的程序修改一下，让它包含一些录入错误，例如，将String的大小写弄错：

```
public static void main(string[] args)
```

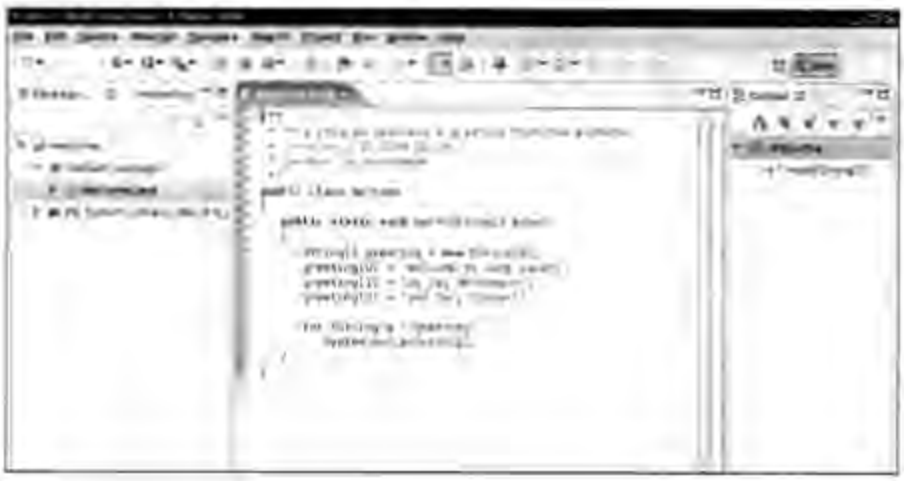

图2-6　使用Eclipse编辑源文件

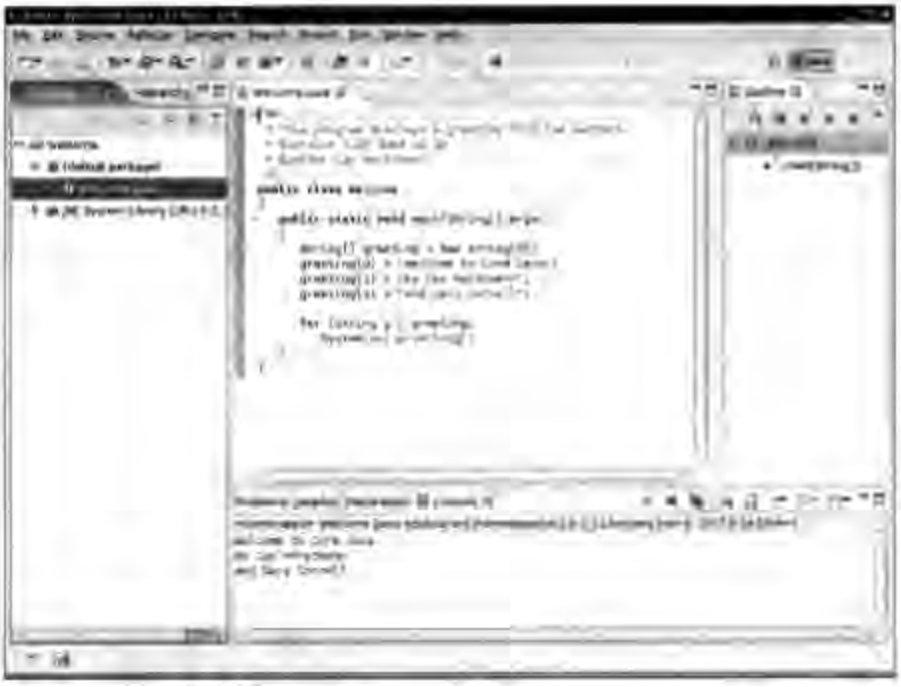

图2-7　在Eclipse 中运行程序

现在，重新编译这个程序，就会收到一条错误消息，其中指出了有一个不能识别的string类型（如图2-8所示）。点击一下错误消息，可以看到光标马上移到了编辑窗口中相应的行上，在这里将错误纠正过来。使用这种方式可以快速地纠正错误。

提示：通常，Eclipse错误报告会伴有一个加亮的图标。点击这个图标可以得到一个建议解决这个错误的方案列表。

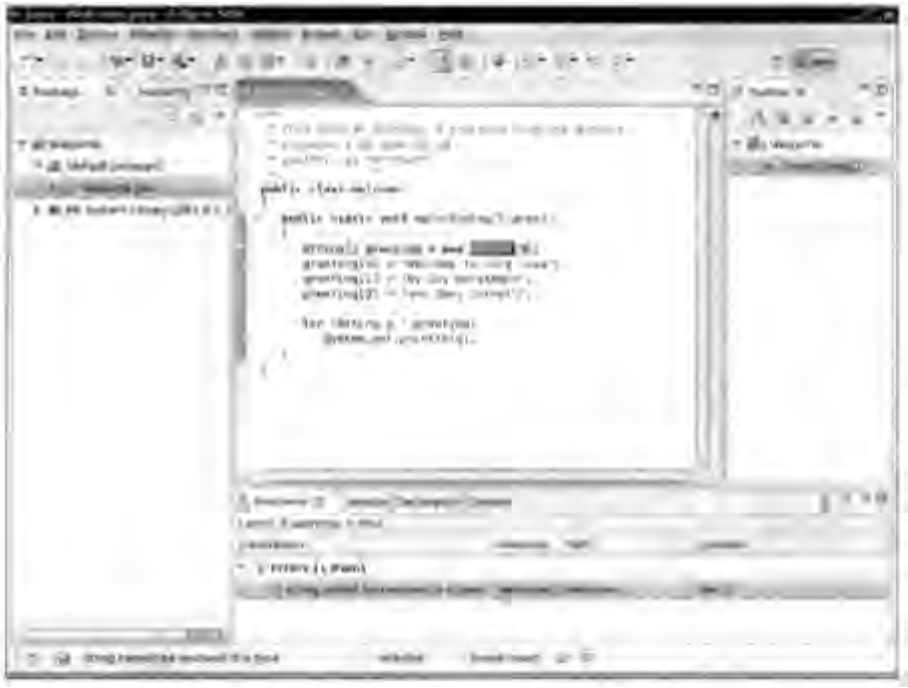

图2-8 Eclipse的错误消息

2.5 运行图形化应用程序

Welcome程序并不会引起人们的兴奋。接下来，给出一个图形化应用程序。这个程序是一个简单的图像文件查看器（viewer），它可以加载并显示一个图像。首先，由命令行编译并运行这个程序。

1）打开一个shell窗口。

2）进入CoreJavaBook/v1ch02/ImageViewer。

3）输入：

```
javac ImageViewer.java
java ImageViewer
```

运行后将弹出一个标题栏为ImageViewer的新程序窗口（如图2-9所示）。

图2-9 运行ImageViewer应用程序

现在，选择File→Open，然后找到一个GIF文件并打开它（在同一个目录下提供了两个示例文件）。

要想关闭这一程序，只需要点击标题栏中的关闭按钮或者从系统菜单中选择关闭程序（要在文本编辑器中或集成开发环境中编译并运行程序，按照前面的步骤做就行了。例如，在Emacs中选择JDE→Compile，然后，再选择JDE→Run App）。

现在读者一定会感觉这个程序既有趣又有用。下面快速地浏览一下源代码。这个程序比第一个程序要长很多，但是只要想一想用C 或C++编写同样功能的应用程序所需要的代码

量，就不会感到它太复杂了。当然，在Visual Basic中，编写这个程序相当简单，只需要简单的拖放几下，再加上两行代码就行了。JDK没有可视化的界面构造器，所以必须通过编写代码完成这一切工作，如例2-2所示。本书将在第7章~第9章介绍编写图形化应用程序的内容。

例2-2 ImageViewer.java

```
import java.awt.EventQueue;
import java.awt.event.*;
import java.io.*;
import javax.swing.*;

/**
 * A program for viewing images.
 * @version 1.22 2007-05-21
 * @author Cay Horstmann
 */
public class ImageViewer
{
   public static void main(String[] args)
   {
      EventQueue.invokeLater(new Runnable()
         {
            public void run()
            {
               JFrame frame = new ImageViewerFrame();
               frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
               frame.setVisible(true);
            }
         });
   }
}

/**
 * A frame with a label to show an image.
 */
class ImageViewerFrame extends JFrame
{
   public ImageViewerFrame()
   {
      setTitle("ImageViewer");
      setSize(DEFAULT_WIDTH, DEFAULT_HEIGHT);

      // use a label to display the images
      label = new JLabel();
      add(label);

      // set up the file chooser
      chooser = new JFileChooser();
      chooser.setCurrentDirectory(new File("."));

      // set up the menu bar
      JMenuBar menuBar = new JMenuBar();
      setJMenuBar(menuBar);

      JMenu menu = new JMenu("File");
```

```
50.       menuBar.add(menu);
51.
52.       JMenuItem openItem = new JMenuItem("Open");
53.       menu.add(openItem);
54.       openItem.addActionListener(new ActionListener()
55.          {
56.             public void actionPerformed(ActionEvent event)
57.             {
58.                // show file chooser dialog
59.                int result = chooser.showOpenDialog(null);
60.
61.                // if file selected, set it as icon of the label
62.                if (result == JFileChooser.APPROVE_OPTION)
63.                {
64.                   String name = chooser.getSelectedFile().getPath();
65.                   label.setIcon(new ImageIcon(name));
66.                }
67.             }
68.          });
69.       JMenuItem exitItem = new JMenuItem("Exit");
70.       menu.add(exitItem);
71.       exitItem.addActionListener(new ActionListener()
72.          {
73.             public void actionPerformed(ActionEvent event)
74.             {
75.                System.exit(0);
76.             }
77.          });
78.    }
79.
80.    private JLabel label;
81.    private JFileChooser chooser;
82.    private static final int DEFAULT_WIDTH = 300;
83.    private static final int DEFAULT_HEIGHT = 400;
84. }
```

2.6 建立并运行applet

本书给出的前两个程序是Java应用程序。它们与所有本地程序一样，可以独立地运行。然而，正如第1章提到的，有关Java的大量宣传都在炫耀Java能够在浏览器中运行applet的能力。下面介绍一下如何利用命令行建立并运行applet。然后，利用JDK自带的applet查看器加载applet。最后，在Web浏览器中显示。

首先，打开shell窗口并将目录转到CoreJavaBook/v1ch02/WelcomeApplet，然后，输入下面的命令：

```
javac WelcomeApplet.java
appletviewer WelcomeApplet.html
```

图2-10显示了在applet查看器窗口中显示的内容。

第一条命令是大家已经非常熟悉的调用Java编译器的命令。它将WelcomeApplet.java源文件编译成字节码文件WelcomeApplet.class。

在这里，不要运行Java程序，而调用appletviewer程序。这是JDK自带的一个特殊工具，它

可以帮助人们快速地测试applet。这里需要向这个程序提供一个HTML文件，而不是Java类文件名。WelcomeApplet.html的内容请参看下面的例2-3。

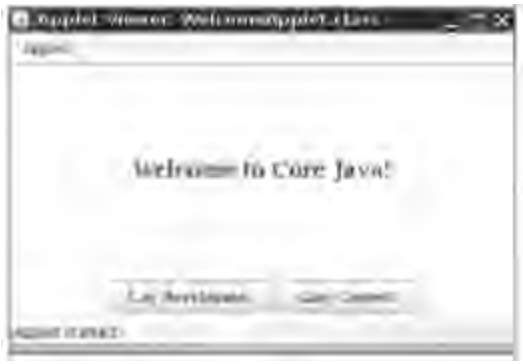

图2-10 在applet查看器窗口中显示的WelcomeApplet

例2-3 WelcomeApplet.html

```
<html>
   <head>
      <title>WelcomeApplet</title>
   </head>
   <body>
      <hr/>
      <p>
         This applet is from the book
         <a href="http://www.horstmann.com/corejava.html">Core Java</a>
         by <em>Cay Horstmann</em> and <em>Gary Cornell</em>,
         published by Sun Microsystems Press.
      </p>
      <applet code="WelcomeApplet.class" width="400" height="200">
         <param name="greeting" value ="Welcome to Core Java!"/>
      </applet>
      <hr/>
      <p><a href="WelcomeApplet.java">The source.</a></p>
   </body>
</html>
```

如果熟悉HTML就会注意到一些标准的HTML指令和applet标记，它们用于告诉applet查看器需要加载一个applet，其代码存放于WelcomeApplet.class中。applet查看器将忽略applet标记之外的所有HTML标记。遗憾的是，浏览器的情况有点复杂。

- 在Windows、Linux 和Mac OS X环境下，Firefox支持Java 。要想实验一下applet，只需要下载这些浏览器的最新版本（访问http://java.com），并利用版本检查器查看一下是否需要安装Java插件。
- Internet Explorer的有些版本根本不支持Java。其他的也只支持Microsoft Java Virtual Machine的过时版本。如果运行Internet Explorer，就需要从http://java.com上下载并安装Java插件。
- 如果使用Macintosh运行OS X，Safari与 Macintosh的Java被集成在一起。在编写本书时支

持Java SE 5.0。

假定有一个支持目前流行的Java版本的浏览器，可以在浏览器中试着加载applet。

1）启动浏览器。

2）选择File→Open File (或等效的操作)。

3）进入CoreJavaBook/v1ch02/WelcomeApplet目录。在文件对话框中，将会看到WelcomeApplet.html文件。加载这个文件。

4）浏览器将加载applet，并显示文字。显示效果基本上如图2-11所示。

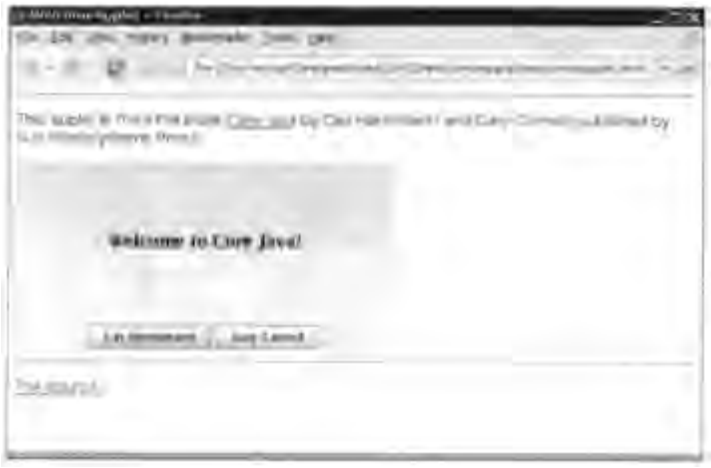

图2-11 在浏览器中运行WelcomeApplet

读者可能会发现这个应用程序是活动的，并且可以与Internet进行交互。点击Cay Horstmann按钮，applet会让浏览器显示Cay的网页。点击Gray Cornell 按钮，applet会让浏览器弹出一个邮件窗口，其中包含已经填写好的Gray邮件地址。

需要注意的是，在查看器中的两个按钮并不起作用。applet查看器没有能力发送邮件或显示一个网页，因此会忽略这里的操作请求。applet查看器适于用来单独地测试applet，但是，最终需要将applet放到浏览器中，以便检测与浏览器以及Internet的交互情况。

提示：也可以从编辑器或集成开发环境内部运行applet。在Emac中，从菜单中选择JDE→Run Applet。在Eclipse中，使用Run→Run as→Java Applet菜单选项。

最后，在例2-4中给出了这个applet的代码。现在，只要阅读一下就可以了。在第10章中，还会再次介绍applet的编写。

例2-4 WelcomeApplet.java

```
1. import java.awt.*;
2. import java.awt.event.*;
3. import java.net.*;
4. import javax.swing.*;
5.
6. /**
7.  * This applet displays a greeting from the authors.
8.  * @version 1.22 2007-04-08
```

```
 * @author Cay Horstmann
 */
public class WelcomeApplet extends JApplet
{
   public void init()
   {
      EventQueue.invokeLater(new Runnable()
         {
            public void run()
            {
               setLayout(new BorderLayout());

               JLabel label = new JLabel(getParameter("greeting"), SwingConstants.CENTER);
               label.setFont(new Font("Serif", Font.BOLD, 18));
               add(label, BorderLayout.CENTER);

               JPanel panel = new JPanel();

               JButton cayButton = new JButton("Cay Horstmann");
               cayButton.addActionListener(makeAction("http://www.horstmann.com"));
               panel.add(cayButton);

               JButton garyButton = new JButton("Gary Cornell");
               garyButton.addActionListener(makeAction("mailto:gary_cornell@apress.com"));
               panel.add(garyButton);

               add(panel, BorderLayout.SOUTH);
            }
         });
   }

   private ActionListener makeAction(final String urlString)
   {
      return new ActionListener()
         {
            public void actionPerformed(ActionEvent event)
            {
               try
               {
                  getAppletContext().showDocument(new URL(urlString));
               }
               catch (MalformedURLException e)
               {
                  e.printStackTrace();
               }
            }
         };
   }
}
```

在本章中，学习了有关编译和运行Java程序的机制。现在可以转到第3章开始学习Java语言了。