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

Oracle 公司为 Linux、Mac OS X、Solaris 和 Windows 提供了 Java 开发工具箱（JDK）的最新、最完整的版本。用于很多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。

2.1.1　下载 JDK

要想下载 Java 开发工具箱，必须访问 Oracle 网站，地址：www.oracle.com/technetwork/java/javase/downloads，并且在得到所需的软件之前必须弄清楚大量的专业术语。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	用于手机和其他小型设备的 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	Oracle 的术语，用于发布修改的 bug
NetBeans	—	Oracle 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JDK 1.2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Oracle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 7开发包做出的第一次更新，正式称为JDK 7u1，内部的版本号为1.7.0_01。更新不需要安装在前一个版本上，它将包含整个JDK的最新版本。

Oracle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下载网站时，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Oracle的集成开发环境，只需要从http://netbeans.org下载。

注释：安装过程提供了包含JDK版本号（如jdk1.7.0）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:\ProgramFiles\jdk1.7.0。应该将Program 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/jdk1.7.0/bin或c:\ jdk1.7.0\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Linux、Mac OS X和Solaris）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Bourne Again shell（Linux的默认选择），在~/.bashrc或

~/.bash_profile 文件的末尾就要添加：

```
export PATH=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 XP 中，立即会看到 System Properties 对话框。在 Vista 和 Windows 7 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 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：

```
jdk\bin;other stuff
```

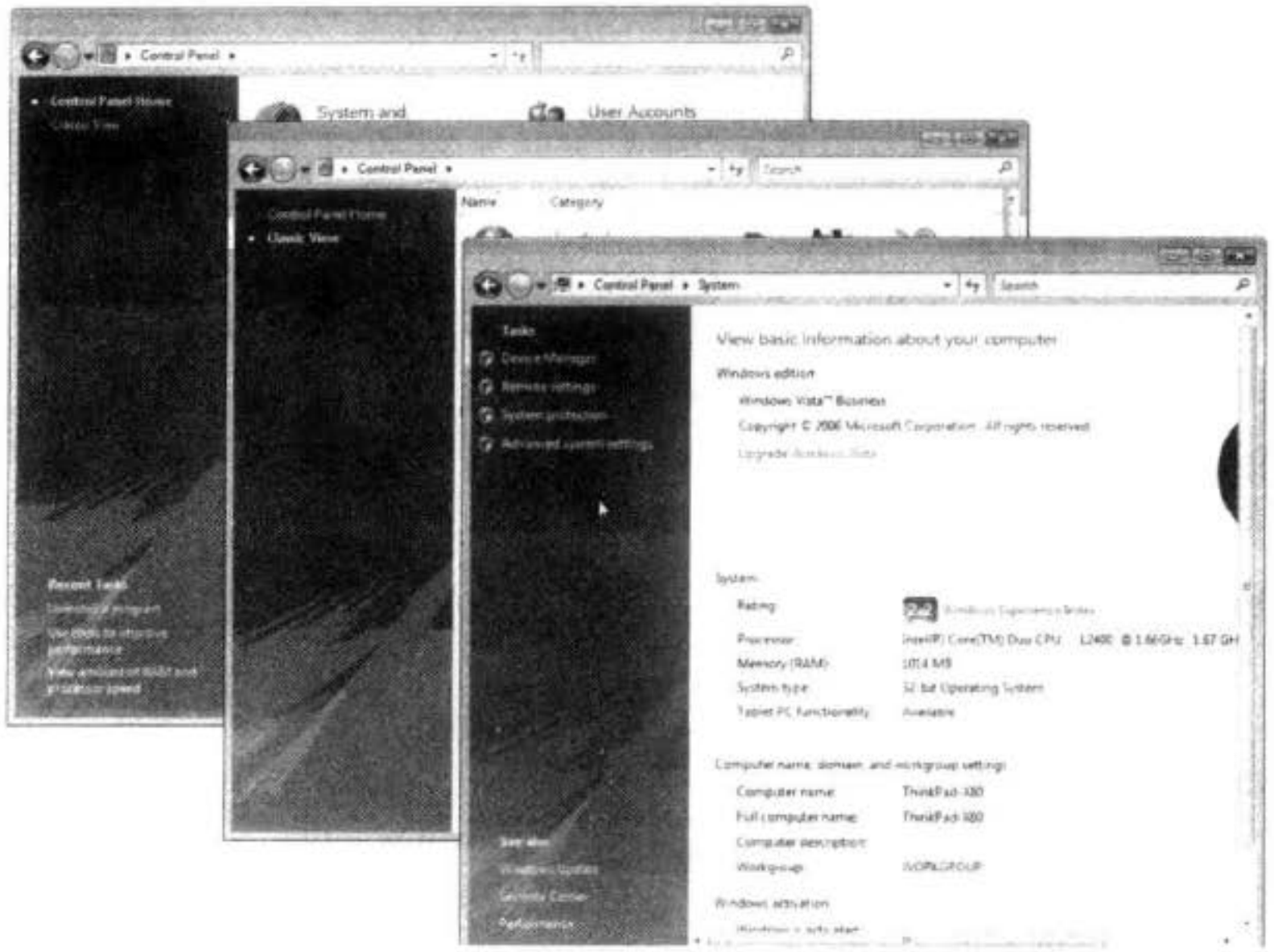

图 2-1 在 Windows Vista 中启动的 System Properties 对话框

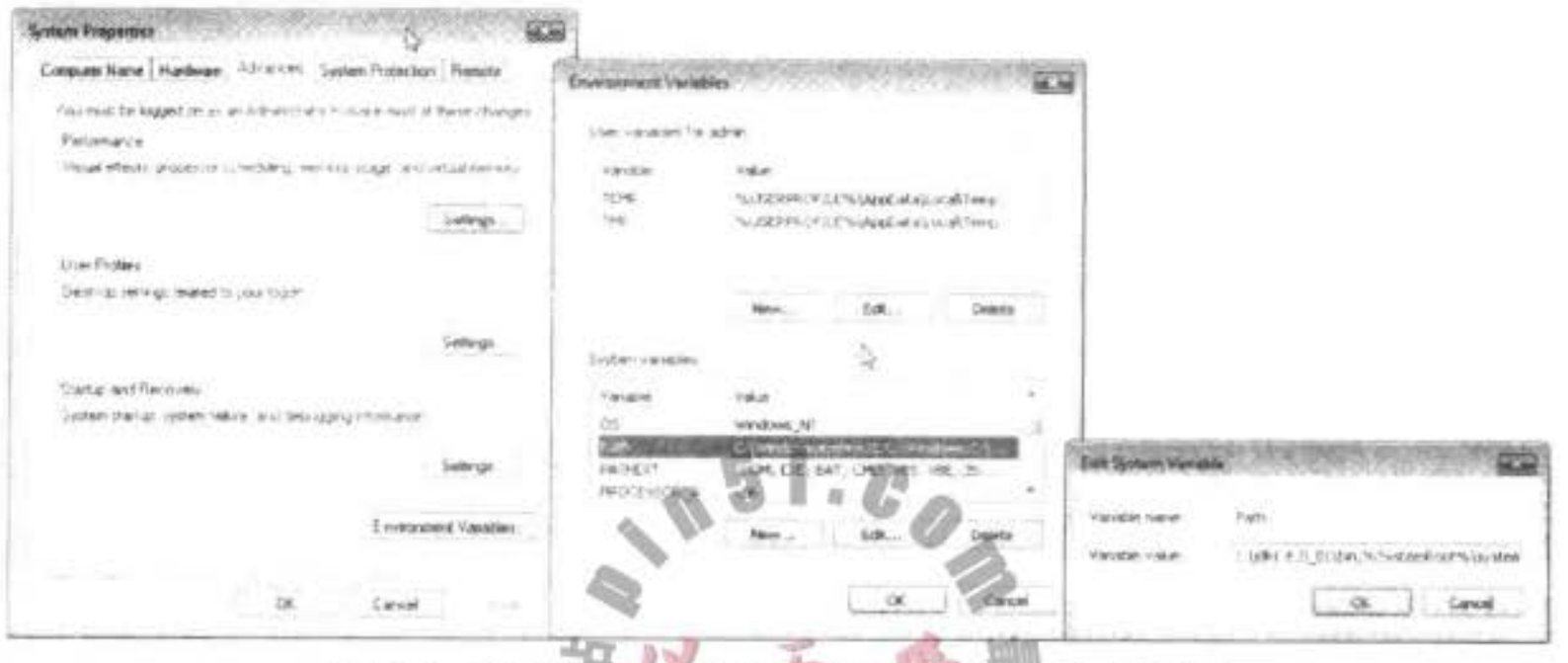

图 2-2 在 Windows Vista 中设置 Path 环境变量

要注意，需要将 *jdk* 替换为你的 Java 安装的实际路径，如 c:\jdk1.7.0_02。如果忽视我们的建议，想要保留 Program Files 目录，则要把整个路径用双引号引起来，如：“c:\Program Files\jdk 1.7.0_02\bin”;other stuff。

将这个设置保存起来，新打开的任何控制台窗口都会有正确的路径。

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：

```
javac -version
```

然后，按 ENTER 键。应该能够看到下面的显示信息：

```
javac 1.7.0_02
```

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

注释： 在 Windows 中，按照下面的命令打开 shell 窗口。如果使用 Windows XP 环境，那么在开始菜单中选择 Run 选项，并键入 cmd。在 Vista 和 Windows 7 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。

如果以前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。例如 http://www.horstmann.com/bigj/help/windows/tutorial. html。

2.1.3 安装库源文件和文档

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/jdk1.7.0 或 cd c:\jdk1.7.0）。

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://jdk7.java.net。

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://www.oracle.com/technetwork/java/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*-apidocs.zip，

其中的 *version* 表示版本号，例如 7。

3）打开一个 shell 窗口。

4）进入 jdk 目录。

5）执行命令：

```
jar xvf jdk-version-apidocs.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程序示例。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。图 2-3 显示了 JDK 目录树。

目录结构	描　述
jdk	（名字可能不同，例如：jdk1.7.0_02）
├─*bin*	编译器和工具
├─*demo*	演示
├─*docs*	HTML 格式的类库文档（解压 j2sdk*version*-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（解压 src.zip 之后）

图 2-3　JDK 目录树

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：Firefox）查看这些文档。

提示：在浏览器中设置一个指向 docs/api/index.html 的书签。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，如果倾向于使用其他的开发环境，同样也可以完成本书的所有程序。

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2-4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。虚拟机执行编译器放在class文件中的字节码。

图 2-4 编译并运行 Welcome.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。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。你可能想查看程序清单 2-1 的程序代码。（在下一章中，将解释它是如何工作的。）

程序清单 2-1 Welcome/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。

常常会出现很多错误，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，收到关于 java.lang.NoClassDefFoundError 的错误消息，就应该仔细地检查出问题的类名。

  如果收到关于 welcome（w 为小写）的错误消息，就应该重新执行命令：java Welcome（W 为大写）。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，应该重新执行命令 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=
```

提示： 在 http://docs.oracle.com/javase/tutorial/getStarted/cupojava/ 上有一个很好的教程。其中提到了初学者经常容易犯的一些错误。

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 2-5 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。

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 2-6 所示。

图 2-5 Eclipse 中的“New Project”对话框

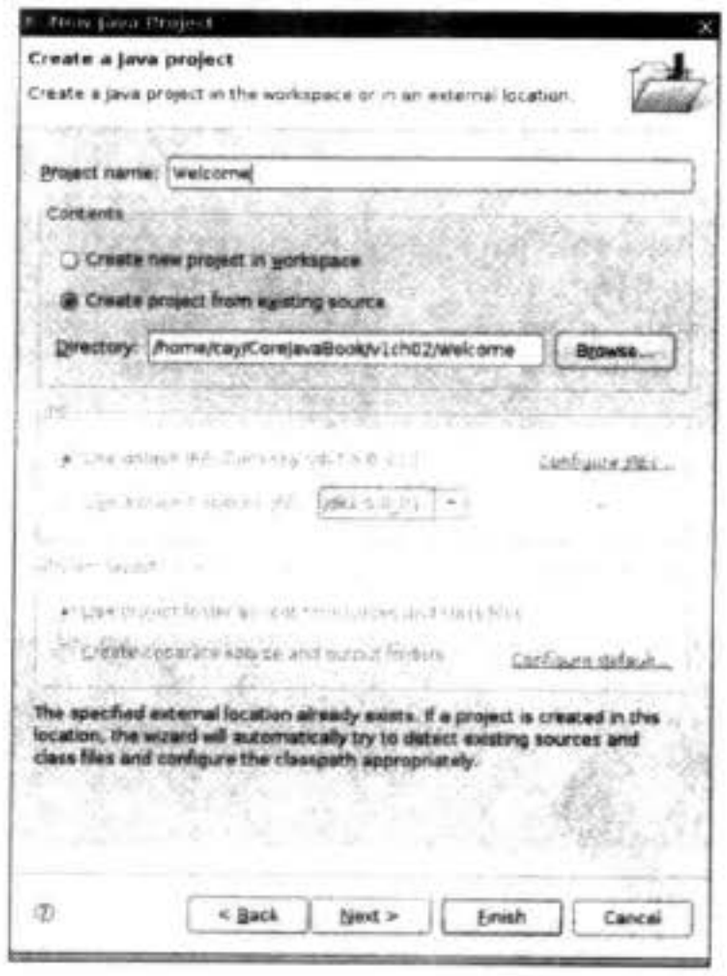

图 2-6 配置 Eclipse 工程

4）确保选中标有 Create project form existing source 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 Default package 旁边的三角，再双击 Welcome.java。现在应该看到带有程序代码的窗口了（如图 2-7 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 2-8 所示）。

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：

```
String[] greeting = new string[3];
```

图 2-7 使用 Eclipse 编辑源文件

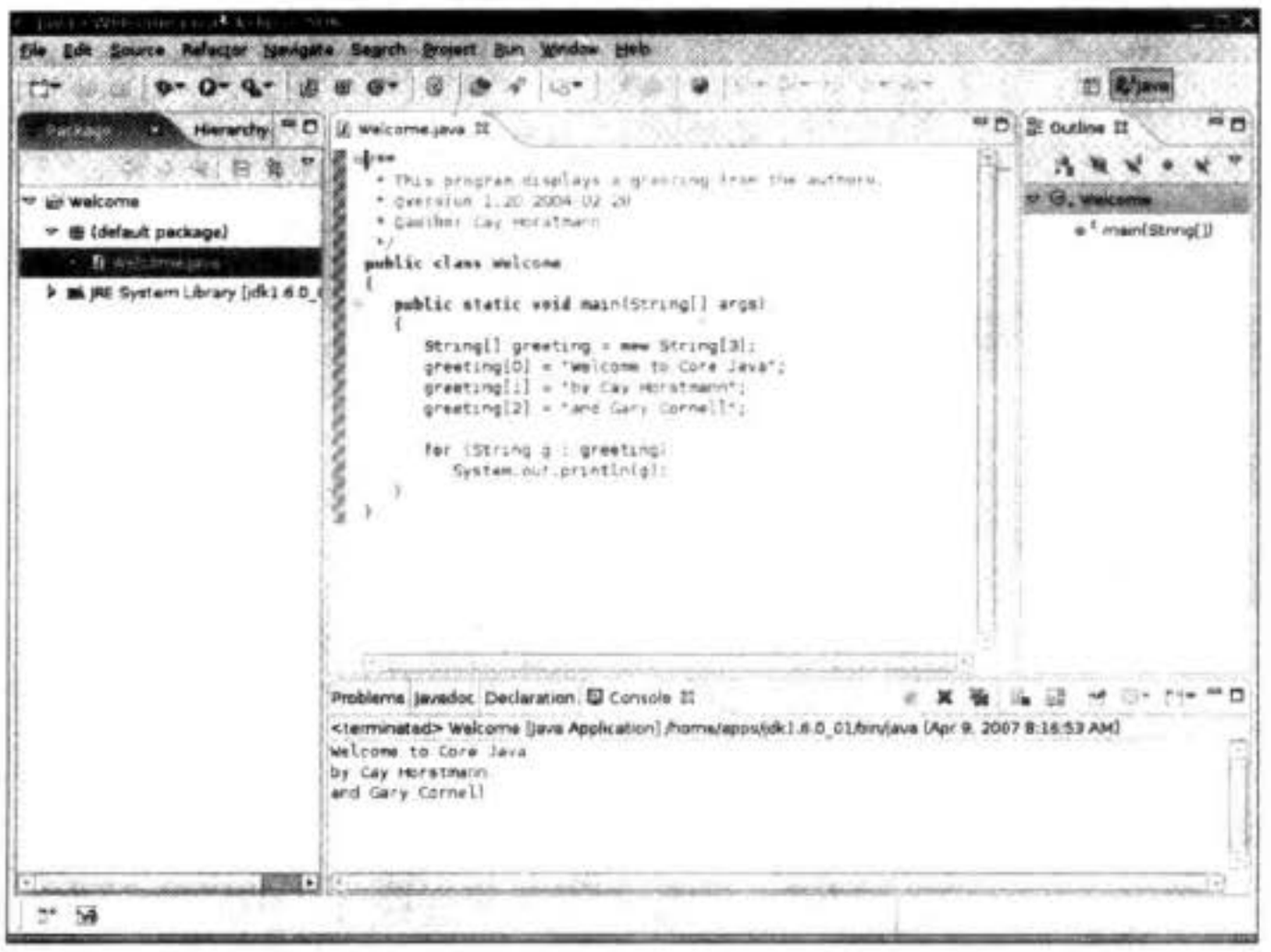

图 2-8 在 Eclipse 中运行程序

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 2-9 所示）。点击这个错误消息，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灯泡的图标。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。

通过这些介绍，你应该对集成环境已经有些认识了。我们将在第 11 章讨论 Eclipse 调试工具。

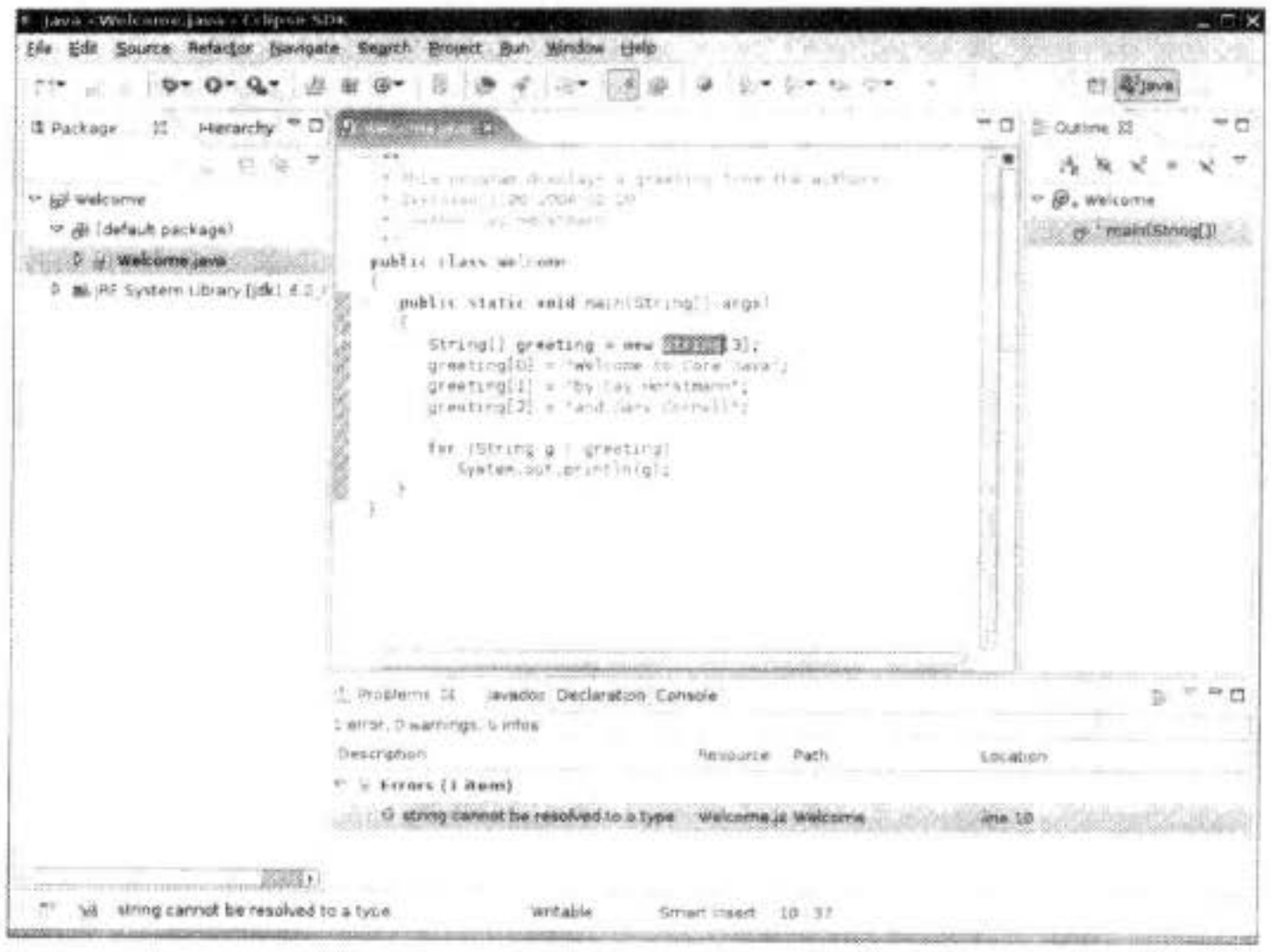

图 2-9 Eclipse 的错误消息

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 2-10 所示）。

现在，选择 File → Open，然后找到一个图像文件并打开它（我们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菜单中选择 File → Exit。

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（或者更确切地说“拖放”

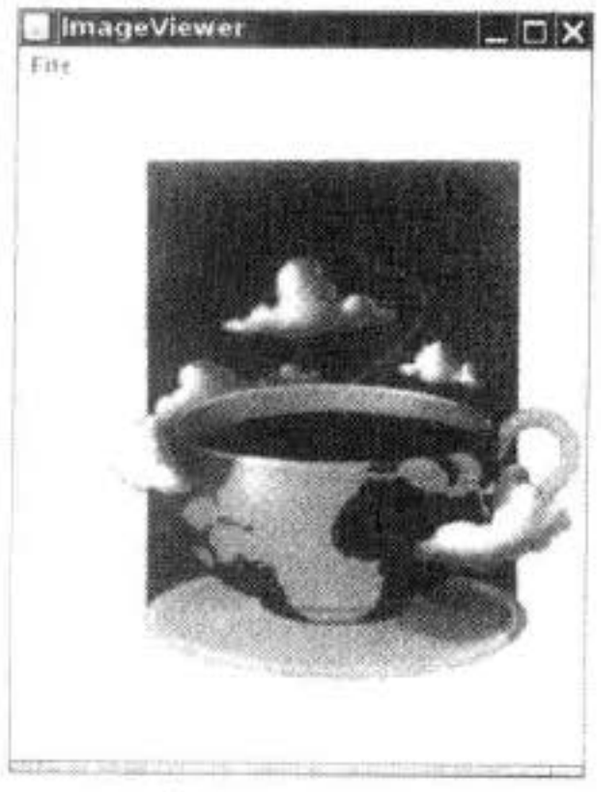

图 2-10 运行 ImageViewer 应用程序

出）这个程序相当简单。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，如程序清单 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。

程序清单 2-2 ImageViewer/ImageViewer.java

```
1  import java.awt.EventQueue;
2  import java.awt.event.*;
3  import java.io.*;
4  import javax.swing.*;
5
6  /**
7   * A program for viewing images.
8   * @version 1.22 2007-05-21
9   * @author Cay Horstmann
10  */
11 public class ImageViewer
12 {
13    public static void main(String[] args)
14    {
15       EventQueue.invokeLater(new Runnable()
16          {
17             public void run()
18             {
19                JFrame frame = new ImageViewerFrame();
20                frame.setTitle("ImageViewer");
21                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
22                frame.setVisible(true);
23             }
24          });
25    }
26 }
27
28 /**
29  * A frame with a label to show an image.
30  */
31 class ImageViewerFrame extends JFrame
32 {
33    private JLabel label;
34    private JFileChooser chooser;
35    private static final int DEFAULT_WIDTH = 300;
36    private static final int DEFAULT_HEIGHT = 400;
37    public ImageViewerFrame()
38    {
39       setSize(DEFAULT_WIDTH, DEFAULT_HEIGHT);
40
41       // use a label to display the images
42       label = new JLabel();
43       add(label);
44
45       // set up the file chooser
46       chooser = new JFileChooser();
47       chooser.setCurrentDirectory(new File("."));
48
49       // set up the menu bar
```

```
50          JMenuBar menuBar = new JMenuBar();
51          setJMenuBar(menuBar);
52
53          JMenu menu = new JMenu("File");
54          menuBar.add(menu);
55
56          JMenuItem openItem = new JMenuItem("Open");
57          menu.add(openItem);
58          openItem.addActionListener(new ActionListener()
59             {
60                public void actionPerformed(ActionEvent event)
61                {
62                   // show file chooser dialog
63                   int result = chooser.showOpenDialog(null);
64
65                   // if file selected, set it as icon of the label
66                   if (result == JFileChooser.APPROVE_OPTION)
67                   {
68                      String name = chooser.getSelectedFile().getPath();
69                      label.setIcon(new ImageIcon(name));
70                   }
71                }
72             });
73
74          JMenuItem exitItem = new JMenuItem("Exit");
75          menu.add(exitItem);
76          exitItem.addActionListener(new ActionListener()
77             {
78                public void actionPerformed(ActionEvent event)
79                {
80                   System.exit(0);
81                }
82             });
83       }
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 2-11 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。

图 2-11 在 applet 查看器窗口中显示的 WelcomeApplet

不过这一次不要运行 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 文件名，而不是 Java 类文件名。WelcomeApplet.html 的内容请参看下面的程序清单 2-3。

程序清单 2-3 WelcomeApplet/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>.
      </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>
```

当然，applet 就是要在浏览器中查看。遗憾的是，很多浏览器默认情况下并没有启用 Java 支持。http://java.com/en/download/help/enable_browser.xml 介绍了如何配置一些最常用的浏览器，使它们支持 Java。一旦完成浏览器的配置，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 2-12 所示。

图 2-12 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 的邮件地址。

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：也可以从集成开发环境内部运行 applet。在 Eclipse 中，使用 Run → Run as → Java Applet 菜单选项。

最后，在程序清单 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。

程序清单 2-4 WelcomeApplet/WelcomeApplet.java

```
1  import java.awt.*;
2  import java.awt.event.*;
3  import java.net.*;
4  import javax.swing.*;
5  /**
6   * This applet displays a greeting from the authors.
7   * @version 1.22 2007-04-08
8   * @author Cay Horstmann
9   */
10 public class WelcomeApplet extends JApplet
11 {
12    public void init()
13    {
14       EventQueue.invokeLater(new Runnable()
15          {
16             public void run()
17             {
18                setLayout(new BorderLayout());
19
```

```
20          JLabel label = new JLabel(getParameter("greeting"), SwingConstants.CENTER);
21          label.setFont(new Font("Serif", Font.BOLD, 18));
22          add(label, BorderLayout.CENTER);
23
24          JPanel panel = new JPanel();
25
26          JButton cayButton = new JButton("Cay Horstmann");
27          cayButton.addActionListener(makeAction("http://www.horstmann.com"));
28          panel.add(cayButton);
29
30          JButton garyButton = new JButton("Gary Cornell");
31          garyButton.addActionListener(makeAction("mailto:gary_cornell@apress.com"));
32          panel.add(garyButton);
33
34          add(panel, BorderLayout.SOUTH);
35       }
36    });
37 }
38
39 private ActionListener makeAction(final String urlString)
40 {
41    return new ActionListener()
42       {
43          public void actionPerformed(ActionEvent event)
44          {
45             try
46             {
47                getAppletContext().showDocument(new URL(urlString));
48             }
49             catch (MalformedURLException e)
50             {
51                e.printStackTrace();
52             }
53          }
54       };
55 }
56 }
```

在本章中，我们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。