

21世纪计算机专业系列精品教材

Java 大学教程

JAVA DAXUE JIAOCHENG

周 斌 石亮军/主 编
王维虎 冯春华 刘 忠/副主编

21 世纪计算机专业系列精品教材

Java 大学教程

主 编 周 斌 石亮军

副主编 王维虎 冯春华 刘 忠

 天津大学出版社
TIANJIN UNIVERSITY PRESS

内 容 提 要

本书是高校培养应用型人才的配套教材,主要介绍了 Java 语言及其编程技术。本书注重面向对象的思想与 Java 程序设计技术的结合,培养读者使用面向对象的思维方式思考问题,并使用 Java 语言解决问题。本书共分为 11 章:第 1、2 章主要介绍 Java 语言的基础知识,使读者初步了解 Java 语言;第 3、4 章是本书的核心,介绍 Java 中面向对象的概念及具体实现,重点讲述继承和多态;第 5、6 章主要介绍 Java 中各种类型的数组、字符串类、数据结构以及一些常用算法;第 7 章主要介绍 Java 中的各种流及相关应用;第 8 章介绍图形化用户界面的设计;第 9 章介绍异常的处理及多线程技术;第 10 章主要介绍 Java 中常用的几种网络通信模式;第 11 章介绍数据库的基础知识以及利用 JDBC 实现 Java 数据库编程。

本书主要作为高等院校计算机及相关专业 Java 课程的教材,也可作为 Java 入门的参考书,供面向对象编程爱好者和自学 Java 编程的读者使用。

图书在版编目(CIP)数据

Java 大学教程/周斌,石亮军主编. —天津:天津大学出版社,2011.12

21 世纪计算机专业系列精品教材

ISBN 978-7-5618-4221-8

I. ①J… II. ①周… ②石… III. ①JAVA 语言—程序设计—高等学校—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字(2011)第 238248 号

出版发行 天津大学出版社

出版人 杨欢

地 址 天津市卫津路 92 号天津大学内(邮编:300072)

电 话 发行部:022-27403647 邮购部:022-27402742

网 址 www.tjup.com

印 刷 廊坊市长虹印刷有限公司

经 销 全国各地新华书店

开 本 185mm×260mm

印 张 16

字 数 399 千

版 次 2012 年 1 月第 1 版

印 次 2012 年 1 月第 1 次

定 价 32.00 元

凡购本书,如有缺页、倒页、脱页等质量问题,请向我社发行部联系调换

版权所有 侵权必究

前 言

面向对象编程技术是当前乃至今后相当长的一段时间内的主流编程技术，它正在逐步代替传统的面向过程的程序设计技术。面向对象是当前计算机界关心的重点，它是 20 世纪 90 年代软件开发方法的主流。面向对象的概念和应用已超越了程序设计和软件开发，扩展范围很广，如数据库系统、交互式界面、应用结构、应用平台、分布式系统、网络管理结构、CAD 技术、人工智能等领域。

Java 语言则是面向对象技术应用最为成功的典范之一，它由 Sun Microsystems 公司于 1995 年推出，从诞生之日起便引起广泛关注，在短短几年的时间内便风靡全球。它以纯面向对象、多线程、平台无关性、高安全性、出色的可移植性和可扩展性受到了计算机界的欢迎，并得到了广泛的应用。随着当前 IT 业的迅猛发展，Java 在软件开发领域显得越来越重要。目前国内外对 Java 开发人员的需求量极大，特别是在最新的软件研发领域。

本书旨在培养读者使用面向对象的思维来分析和解决问题，掌握 Java 程序的编写方式和技巧，以适应现代社会对人才的需求。本书在内容取舍上作了精心选择，强调以简单、易懂的示例来讲述知识要点，使读者能够快速理解和上手，同时确保有一定的深度与广度。

下面简要介绍本书的主要内容。

第 1 章 Java 语言概述：主要介绍 Java 语言的发展历史、Java 运行环境的安装与配置，讨论了 Java 程序的分类，通过最简单的 Java 程序示例，介绍 Java 程序开发的基本步骤。

第 2 章 Java 语言基础：主要介绍 Java 语言的基础结构，例如，数据类型、常量、变量、表达式和流程控制语句等，为后续章节的程序设计打下基础。

第 3 章 类与对象：主要介绍 Java 语言的面向对象技术，包括面向对象的基本概念、面向对象的程序设计方法、Java 中的类与对象、类的继承、方法和域以及访问控制符。

第 4 章 多态、包与接口：主要介绍面向对象程序设计中多态的概念及具体应用，以及包和接口。

第 5 章 数组与字符串类：首先介绍 Java 编程中经常用到的数组，包括一维数组、二维数组和对象数组，通过示例程序进一步来讨论它们的使用方式与技巧；然后介绍字符串类，包括两种具有不同操作方式的 String 类和 StringBuffer 类。

第 6 章 数据结构与常用算法：首先介绍了向量、哈希表的概念与使用方式，以及数据结构中的接口，然后讨论数据结构中常见的堆栈、队列、链表和二叉树结构以及排序与查找的算法。

第 7 章 流与文件：主要介绍流的概念、基本输入/输出流、常用输入/输出流、标准输入/输出流以及文件的处理。

第 8 章 GUI 设计：主要介绍 GUI 的概念、文字与图形的 GUI 设计、常用的 GUI 设计组件、布局设计以及 Java 中的事件处理机制。

第 9 章 异常处理与多线程：主要介绍 Java 中异常处理的机制、处理语句、多线程



程的实现与管理。

第 10 章 Java 网络编程：主要介绍 Java 网络通信中常见的 Socket 通信模式、UDP 通信模式以及 URL 通信模式。

第 11 章 Java 数据库应用：主要介绍数据库的基础知识、SQL 语言的基本语法以及使用 JDBC 实现 Java 访问数据库。

任何程序设计课程都是一门需要大量实践的课程。读者在学习本书内容的同时应注意辅以大量的实践练习，这样才能较好地理解和掌握书中所介绍的知识点。同时注意举一反三，增强自己编程时的灵活应变能力，为将来从事软件开发工作打下良好的基础。

在本书出版过程中，得到天津大学出版社的大力支持，在此表示感谢。

由于时间仓促，作者水平有限，书中难免存在疏漏之处，欢迎广大读者批评指正。

编 者

2011 年 7 月

目 录

第 1 章 Java 语言概述.....	1	习题.....	93
1.1 Java 语言的起源与发展.....	1	第 6 章 数据结构与常用算法.....	94
1.2 Java 运行环境的安装与配置.....	1	6.1 向量.....	94
1.3 Java 程序的分类.....	4	6.2 哈希表.....	100
1.4 简单的 Java 程序示例.....	4	6.3 数据结构中的接口.....	101
习题.....	7	6.4 链表.....	106
第 2 章 Java 语言基础.....	8	6.5 堆栈.....	108
2.1 Java 程序的基本结构.....	8	6.6 队列.....	109
2.2 Java 程序中的标志符、 关键字及分隔符.....	9	6.7 二叉树.....	111
2.3 数据类型.....	10	6.8 排序算法.....	113
2.4 常量与变量.....	13	6.9 查找算法.....	117
2.5 表达式.....	16	习题.....	120
2.6 流程控制语句.....	21	第 7 章 流与文件.....	121
习题.....	33	7.1 流的概念.....	121
第 3 章 类与对象.....	34	7.2 基本输入/输出流.....	122
3.1 面向对象的基本概念.....	34	7.3 常用输入/输出流.....	125
3.2 面向对象的基本特征.....	35	7.4 标准输入/输出.....	130
3.3 Java 中的类与对象.....	36	7.5 文件处理.....	132
3.4 类的继承.....	42	习题.....	138
3.5 抽象类与最终类.....	49	第 8 章 GUI 设计.....	139
3.6 域.....	52	8.1 GUI 设计概述.....	139
3.7 方法.....	55	8.2 Applet 小程序.....	140
3.8 访问控制符.....	59	8.3 文字、图形 GUI 设计.....	143
习题.....	61	8.4 常用组件 GUI 设计.....	150
第 4 章 多态、包与接口.....	62	8.5 容器.....	170
4.1 多态机制.....	62	8.6 布局设计.....	177
4.2 包与接口.....	65	8.7 菜单设计.....	183
4.3 程序举例.....	69	8.8 对话框.....	186
习题.....	75	8.9 Swing GUI 组件.....	189
第 5 章 数组与字符串类.....	77	习题.....	196
5.1 数组.....	77	第 9 章 异常处理与多线程.....	197
5.2 字符串类.....	88	9.1 异常.....	197
		9.2 异常处理语句.....	199

9.3 多线程技术	206	第 11 章 Java 数据库应用	235
习题	219	11.1 数据库基础	235
第 10 章 Java 网络编程	220	11.2 利用 JDBC 访问数据库	238
10.1 Socket 通信模式	220	11.3 利用 JDBC 实现 Java 数据库应用实例	243
10.2 UDP 通信模式	226	习题	248
10.3 URL 通信模式	230	参考文献	249
习题	234		

第 1 章 Java 语言概述

本章首先介绍 Java 语言的发展历史，并对 Java 运行环境的安装与配置进行详细介绍。接着讨论 Java 程序的分类。本章将通过两个最简单的 Java 程序示例，详细讲述 Java 程序开发的基本步骤。

1.1 Java 语言的起源与发展

Java 语言是由 Sun 公司于 1995 年推出的一种新的编程语言。1991 年，Sun 公司预料嵌入式系统将在家电行业大展辉煌，于是试图为下一代智能家电编写一个通用的控制系统，其主要目的是为家用电子产品开发一个分布式代码系统，以便可以与电冰箱、电视机等家用电器进行信息交流，如向它们发送 E-mail，对它们进行控制。因此便成立了一个由 James Gosling 为领导的“Green”计划。

开发小组开始准备采用 C++ 语言，但由于 C++ 太复杂和安全性差，最后开发了一种基于 C++ 的新语言 Oak（一种橡树的名字），它是 Java 的前身。Oak 能够用于网络且安全性好，但当时的 Oak 并没有引起人们的注意。直到 1994 年，互联网和浏览器的出现给 Oak 语言带来了生机，随着互联网和 3W 的飞速发展，Gosling 将 Oak 语言进行了小规模修改，完成了一个网页浏览器 WebRunner（该浏览器后来被更名为 HotJava），得到了 Sun 公司首席执行官 Scott McNealy 的支持，才得以研发和发展。在准备注册该商标时，发现 Oak 已经被别人使用，于是更名为 Java。

Sun 公司在 1995 年 5 月正式推出 Java 语言，该语言具有安全、跨平台、面向对象、简单、适用于网络等显著特点，当时以 Web 为主要形式的互联网正在迅猛发展，Java 语言的出现迅速引起所有程序员和软件公司的极大关注，程序员们纷纷尝试用 Java 语言编写网络应用程序，并利用网络把程序发布到世界各地。IBM、Oracle、微软、Netscape、Apple、SGI 等大公司纷纷与 Sun Microsystems 公司签订合同，授权使用 Java 平台技术。

目前，Java 语言应用非常广泛，在应用领域占有较大优势，具体体现在以下几个方面。

- 1) 开发桌面应用程序，如银行软件、商场结算软件等。
- 2) 开发面向 Internet 的 Web 应用程序，如门户网站（工商银行）、网上商城、电子商务网站等。
- 3) 提供各行业的数据移动、数据安全等方面的解决方案，如金融、电信、电力等。

目前，Java 语言已发展成为最优秀的应用软件开发语言，它有着众多的开源工具。另外，Java 具有面向对象、跨平台、多线程、分布式处理等优点，因此其使用者越来越多，它在应用程序开发领域所占的份额也越来越大。

1.2 Java 运行环境的安装与配置

Java 开发环境大体上分为两种方式：一种是使用 JDK，这是一种命令行的使用方式；

另一种是使用集成开发环境，如 Borland 公司的 JBuilder、IBM 公司的 Eclipse、Sun 公司的 NetBeans 等。其中 JDK 是初学者首选的开发环境，也是构建其他开发工具的基础。

1.2.1 JDK 运行环境安装

JDK (Java Development Kits) 是 Sun 公司为 Java 编程人员提供的一套免费的 Java 开发和运行环境。自从 Java 推出以来，JDK 已经成为使用最广泛的 Java SDK。JDK 是整个 Java 的核心，包括 Java 运行环境、Java 工具和 Java 基础的类库。

从 Sun 公司的网站 <http://www.sun.com> 上可以免费下载适合于不同计算机操作系统的 JDK。本书使用的 JDK 是 Windows 操作系统下的开发工具 “jdk-6u22-windows-i586.exe”，下面详细介绍 JDK 的安装过程。

获得安装文件 “jdk-6u22-windows-i586.exe” 后直接双击启动安装，在安装过程中可以自定义安装目录等信息，安装设置界面如图 1-1 所示。

在安装设置界面中，可以通过“更改”按钮修改程序的安装路径，这里采用默认路径 “C:\Program Files\Java\jdk1.6.0_22”，单击“下一步”按钮继续安装。

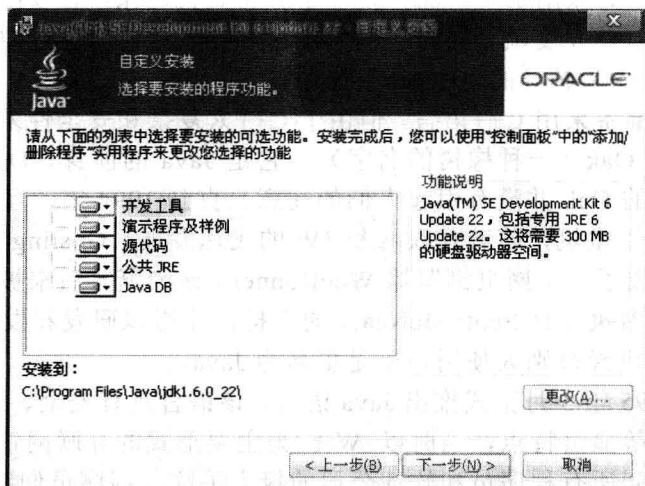


图 1-1 安装设置界面

安装 JDK 后产生的主要目录结构如下。

- \bin 目录：存放 JDK 开发工具的可执行文件，包括 Java 编译器、解释器等。
- \demo 目录：存放含有源代码的不同程序示例。
- \include 目录：包含 C 语言头文件，支持 Java 本地接口与 Java 虚拟机调试程序接口的本地编程技术。
- \jre 目录：是指 Java 运行时环境的根目录，包含 Java 虚拟机、运行时的类包和 Java 应用启动器，但不包含开发环境中的开发工具。
- \lib 目录：包含 Java 类库或库文件，是开发工具使用的归档包文件。

1.2.2 Java 运行环境配置

安装完 JDK 后，必须设置环境变量，JDK 才能正常工作，其中变量 PATH 和 CLASSPATH 是必须设置的。这里设置三个环境变量：JAVA_HOME、CLASSPATH 和 PATH。

下面详细介绍如何配置这三个环境变量。

选中“我的电脑”单击鼠标右键，选择“属性”，弹出“系统属性”对话框，再选择“高级”选项卡，如图 1-2 所示。

在图 1-2 所示的“系统属性”对话框中，单击“环境变量”按钮，进入“环境变量”对话框，如图 1-3 所示，分别设置如下变量。

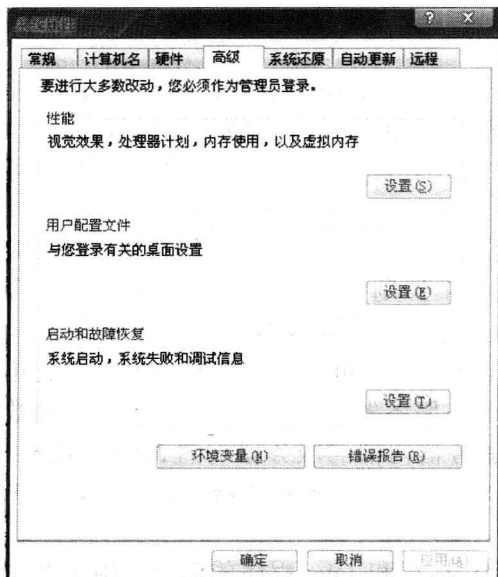


图 1-2 “系统属性”对话框



图 1-3 “环境变量”对话框

1) JAVA_HOME 环境变量：指定 JDK 安装路径，这里安装在 C:\Program Files\Java\jdk 1.6.0_22 目录下，设置 JAVA_HOME 为该目录路径，那么以后要使用该路径时，只需输入 %JAVA_HOME% 即可，避免每次引用都输入很长的路径串。

在图 1-3 所示的对话框中，单击系统变量部分的“新建”按钮，设置 JAVA_HOME 变量。

变量名：JAVA_HOME

变量值：C:\Program Files\Java\jdk1.6.0_22

2) CLASSPATH 环境变量：指定类的路径，Sun 公司提供了丰富的类包，一个是 dt.jar，另一个是 tools.jar，这两个 jar 包都位于 C:\Program Files\Java\jdk1.6.0_22\lib 目录下，通常都把这两个 jar 包加到 CLASSPATH 环境变量中。

在图 1-3 所示的对话框中，单击系统变量部分的“新建”按钮，设置 CLASSPATH 环境变量。

变量名：CLASSPATH

变量值：;%JAVA_HOME%\lib\dt.jar;%JAVA_HOME%\lib\tools.jar

注意：CLASSPATH 变量值最前面有个“.”，表示当前目录，这样，当运行 Java 程序时，系统就会先在当前目录寻找类文件。

3) PATH 环境变量：指定 Java 工具的路径，使得系统可以在任何路径下识别 Java 命令。

在图 1-3 所示的对话框中，在系统变量部分找到 PATH 变量，单击“编辑”按钮，设置 PATH 变量。

变量名: PATH

变量值: %JAVA_HOME%\bin (PATH 变量里面已经有很多变量值, 因此要在变量值的最前面加上 %JAVA_HOME%\bin)

注意: 如果读者的安装路径不一样, 请根据具体的安装路径进行调整。

1.3 Java 程序的分类

Java 程序按其实现环境通常可分为三类: Java Application、Java Applet、Java Servlet。这三种类型程序的区别很少, 并且每种类型的程序都要在 Java 虚拟机上运行。

1) Java Application: 独立的 Java 应用程序, 只需要 Java 虚拟机就能够运行, 可在命令行单独执行。

2) Java Applet: 小应用程序, 不能单独运行, 必须依附于一个用 HTML 语言编写的网页并嵌入其中, 通过与 Java 兼容的浏览器来控制执行。当浏览器装入一个含有 Applet 的 Web 页时, Applet 会被下载到该浏览器中, 并开始执行。

3) Java Servlet: 运行于 Web 服务器端的 Java 程序, 可用来生成动态网页。它担当客户请求 (Web 浏览器或其他 HTTP 客户程序) 与服务器响应 (HTTP 服务器上的数据库或应用程序) 的中间层。Servlet 是位于 Web 服务器内部的服务器端的 Java 应用程序, 与传统的从命令行启动的 Java 应用程序不同, Servlet 由 Web 服务器进行加载, 该 Web 服务器必须包含支持 Servlet 的 Java 虚拟机。

其他还有 JSP 和 Java Bean 等。JSP 是 HTML 标记和 Java 程序的混合, 用于产生动态网页, 在支持 JSP 的 Web 服务器上运行, 严格地说, 它并不是独立的 Java 程序; Java Bean 则是可重用的、独立于平台的 Java 程序组件, 通常作为其他 Java 应用程序的一部分, 不单独运行。

1.4 简单的 Java 程序示例

Java 源程序可以用任何文本编辑器来编写代码, 如 Windows 中的记事本等, 然后用命令行工具进行编译和运行, 但是这样使用 JDK 比较麻烦。另外, 可以使用各种功能强大的 Java 集成开发环境, 如 Eclipse、JBuilder。下面以 Windows 中的记事本为例来说明 Java 程序是如何编写、编译和运行的。

1.4.1 Java Application 程序

打开 Windows 的记事本, 编写例 1-1 所示的 Java 程序源代码, 保存源程序, 修改文件名为 FirstApplication.java。后缀 “.java” 表示文件格式是 Java 类型的。

例 1-1 FirstApplication.java 源代码

```
public class FirstApplication
{
    public static void main(String args[ ])
    {
        System.out.println("Welcome to Java World!");
    }
}
```

本程序的运行结果是输出一行信息:

Welcome to Java World!

下面通过 Java 的编译器编译源文件 FirstApplication.java。在 MS-DOS 命令行中输入如下命令进行编译：

```
javac FirstApplication.java
```

上面命令要正确执行，需要确保以下两个条件：

- 1) JDK 已安装成功并配置环境变量；
- 2) 当前处于 FirstApplication.java 文件所在路径，假设该程序放在 D:\java 目录下。

上面命令将 Java 文件编译生成一个以类名字命名、以.class 为后缀的字节码文件，源代码中定义了几个类，编译结果就生成几个字节码文件。例 1-1 的源代码文件中只定义了一个类 FirstApplication，所以编译结果将生成一个名为 FirstApplication.class 的字节码文件，该文件和 FirstApplication.java 在相同的目录下。

调用 Java 解释器解释执行字节码文件 FirstApplication.class，输出程序运行结果。执行命令如下：

```
java FirstApplication
```

图 1-4 显示了程序的运行结果。

上面程序中有如下几个 Java 语法知识点。

1) 用关键词 `class` 声明一个新类 FirstApplication。类定义由 `{}` 括起来，在类中定义了类的变量和类的方法。任何一个 Java 程序都由若干个类定义组成，这些类只能有一个 `public` 类（公共类），且程序文件名应与公共类同名。

2) 在该类中定义了一个 `main` 方法，它是 Java Application 程序执行的入口点。`main` 方法所在的类叫做主类，一个 JavaApplication 程序只能有一个主类。任何一个 Java Application 类型的程序必须有且只能有一个 `main` 方法，而且这个 `main` 方法的开头必须按照下面的格式书写：

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

当执行 Java Application 时，整个程序将从这个 `main` 方法的方法体的第一个语句开始执行。

3) 在 `main` 方法中，只有一条语句：

```
System.out.println("Welcome to Java World!");
```

它用于将字符串输出到系统屏幕上。

4) Java 语言是严格区分大小写的，因此文件名必须与主类名称完全一致。

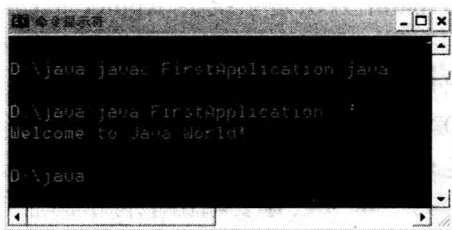


图 1-4 程序运行结果

1.4.2 Java Applet 程序

Applet 是嵌入到 HTML 页面并能够在浏览器中运行的 Java 小应用程序，它的源程序编辑与字节码生成过程与 Java Application 相同。

例 1-2 FirstApplet.java 源代码

```
import java.awt.Graphics;    //导入类 Graphics
import java.applet.Applet;   //导入类 Applet
public class FirstApplet extends Applet
```

```
{  
    public void paint(Graphics g)  
    {  
        g.drawString("Welcome to Java World!", 20, 20);  
    }  
}
```

程序说明：

1) 在程序里使用了行注释，在 Java 程序中，两道斜线 (//) 代表行注释的开始，跟在它后面的所有内容都将被编译器和解释器忽略而作为提高程序可读性的注释部分。

2) 用关键词 `import` 分别导入 `java.awt` 包中的 `Graphics` 类和 `java.applet` 包中的 `Applet` 类。`Graphics` 类使 Java Applet 能绘制各种图形（如直线、矩形、椭圆等）和字符串。类 `Applet` 规定了 Applet 程序如何与执行它的解释器——WWW 浏览器配合工作。

3) 用 `class` 关键词声明了一个名为 `FirstApplet` 的类，关键词 `extends` 表示继承，即 `FirstApplet` 是系统类 `Applet` 的子类，它拥有 `Applet` 类的所有属性和方法。

Java Applet 中不需要有 `main` 方法，但要求程序中有且必须有一个类是系统类 `Applet` 的子类。

4) `paint` 方法是系统类 `Applet` 中已经定义好的成员方法，它与其他一些 `Applet` 中的方法一样，能够被 WWW 浏览器识别和在恰当的時刻自动调用。利用 `paint` 方法可以绘制出用户想要的各种图形。

5) `drawString` 方法是 `paint` 方法的形式参数 `g` 的一个成员方法，其功能是在屏幕上的特定位置输出字符串。`drawString` 方法有三个参数，第一个参数是要输出的字符串，第二、第三个参数是要输出字符串的左下角坐标 (20,20)，它们是以像素为单位的。

使用如下命令在 JDK 中将源程序编译为 class 文件：

```
javac FirstApplet.java
```

编译后生成 Java 字节码文件 `FirstApplet.class`。`Applet` 中没有 `main` 方法作为 Java 解释器的入口，为了执行该代码，需要把它嵌入到 HTML 页面中，代码如下：

```
<HTML>  
<BODY>  
<APPLET CODE="FirstApplet.class" WIDTH=300 HEIGHT=150>  
</APPLET>  
</BODY>  
</HTML>
```

HTML 是一种简单的排版描述语言，称为“超文本标记语言”，它通过各种各样的标记来编排超文本信息。例如，`<HTML>`和`</HTML>`这一对标记标志 HTML 文件的开始和结束。在 HTML 文件中嵌入 Java Applet 同样需要使用一组约定好的特殊标记 `<APPLET>`和`</APPLET>`，其中`<APPLET>`标记必须包含三个参数。

- `CODE`：指明嵌入 HTML 文件中的 Applet 字节码文件的文件名。
- `WIDTH`：指明 Applet 程序在 Web 页面中占用区域的宽度。
- `HEIGHT`：指明 Applet 程序在 Web 页面中占用区域的高度。

将该 HTML 代码保存到 `FirstApplet.html` 文件中，然后运行该程序输出结果，运行该程序有两种方式。

1) 通过支持 Java 的浏览器，打开 FirstApplet.html 文件，可以看到程序运行结果，如图 1-5 所示。

2) 通过 JDK 提供的 AppletViewer 运行程序，例如，运行例 1-2 中的 Java Applet，可用如下命令：

appletviewer FirstApplet.html

运行结果如图 1-6 所示。



图 1-5 程序运行结果（方式 1）

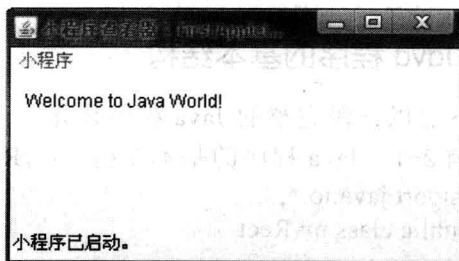


图 1-6 程序运行结果（方式 2）

习 题

1. 下载安装 JDK 软件包，并配置环境变量。
2. Java 程序按其实现环境可分为哪几种？各有什么特点？
3. 编写一个 Java Application 程序，在屏幕上输出“Hello!Java world!”。
4. 编写一个 Java Applet 程序，在窗口中显示“Hello!Java world!”，并编写相应的 HTML 文件。
5. 编写程序输出如下信息：

```
***
*****
*****
```

第 2 章 Java 语言基础

本章主要介绍编写 Java 语言程序所涉及的各种基本元素，包括数据类型、常量、变量、表达式和流程控制语句等。

2.1 Java 程序的基本结构

下面以一段完整的 Java 程序来介绍一下 Java 程序的基本结构。

例 2-1 Java 程序的基本结构 (myRect.java)

```
import java.io.*;
public class myRect                //主类
{
    public static void main(String args[]) //主方法
    {
        Rect rt=new Rect(8.0,2.0);    //创建 Rect 类的对象 rt
        System.out.println("矩形面积为: "+rt.area());
    }
}

class Rect                        //普通类
{
    double length;                //类中的数据成员
    double width;

    Rect(double l,double w) //类中的构造函数
    {
        length=l;
        width=w;
    }

    double area()                //类中的普通方法
    {
        return(length*width);
    }
}
```

程序初始执行结果如图 2-1 所示。

这段程序实现的是求矩形面积的功能。完整的 Java 程序一般由若干个类组成，但其中有且只有一个类被称为主类，它是整个程序的入口，例如，本程序中定义了两个类：myRect 和 Rect。其中 myRect 类是主类。类中成员通常由以下三部分组成。

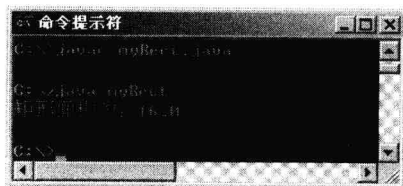


图 2-1 程序初始执行结果

- 1) 数据成员：它在类中用于存放数据。
- 2) 构造函数：它在创建类的对象时用于完成对该对象的初始化工作。
- 3) 普通方法：它是类中的动态成员，类似于其他语言中的函数或过程，是一种用于完成某种操作的程序片段，它主要描述类所具有的功能。

2.2 Java 程序中的标志符、关键字及分隔符

2.2.1 标志符

什么是标志符？在 Java 中变量以及类和方法都需要一定的名称，这种名称就叫做标志符。

标志符的命名基本规则如下：所有的标志符都必须以一个字母、下划线或美元符号“\$”开头，后面的字符可以包含字母、数字、下划线和美元符号。

虽然标志符是由程序员自定义的名称，可按上面的基本规则随意选取，但是为了提高程序的可读性，在定义标志符时，要尽量遵循“见其名知其意”的原则。另外，定义 Java 的标志符还有一些约定俗成的准则：

- 1) 一个标志符可以由几个单词连接而成，以表明它的意思；
- 2) 对于类名，每个单词的首字母都要大写，其他字母则小写，如 `RecordInfo`；
- 3) 对于方法名和变量名，与类名有些相似，除了第一个单词的首字母小写外，其他单词的首字母都要大写，如 `getRecordName()`；
- 4) 对于常量名，每个单词的每个字母都要大写，如果由多个单词组成，通常情况下单词之间用下划线（`_`）分隔，如 `MAX_VALUE`；
- 5) 对于包名，每个单词的每个字母都要小写，如 `com.frame`。

需要注意的是，Java 严格区分字母大小写，标志符中的大小写字母被认为是不同的两个字符。例如，`ad`、`Ad`、`aD`、`Da` 是四个不同的合法标志符。

2.2.2 关键字

关键字是 Java 语言本身使用的标志符，它有其特定的语法含义，如 `public` 表示公有的，`static` 表示静态的。常用的关键字如表 2-1 所示。

表 2-1 常用的关键字

<code>abstract</code>	<code>const</code>	<code>finally</code>	<code>int</code>	<code>public</code>	<code>this</code>
<code>boolean</code>	<code>continue</code>	<code>float</code>	<code>interface</code>	<code>return</code>	<code>throw</code>
<code>break</code>	<code>default</code>	<code>for</code>	<code>long</code>	<code>short</code>	<code>throws</code>
<code>byte</code>	<code>do</code>	<code>goto</code>	<code>native</code>	<code>static</code>	<code>transient</code>
<code>case</code>	<code>double</code>	<code>if</code>	<code>new</code>	<code>strictfp</code>	<code>try</code>
<code>catch</code>	<code>else</code>	<code>implements</code>	<code>package</code>	<code>super</code>	<code>void</code>
<code>char</code>	<code>extends</code>	<code>import</code>	<code>private</code>	<code>switch</code>	<code>volatile</code>
<code>class</code>	<code>final</code>	<code>instanceof</code>	<code>protected</code>	<code>synchronized</code>	<code>while</code>

2.2.3 分隔符

分隔符用于区分源程序中的基本成分，可使编译器确认代码在何处分隔。常用分隔符有以下三种。

(1) 注释符

注释是程序员为了提高程序的可读性和可理解性,在源程序的开始或中间对程序的功能、作者、使用方法等所写的注解。注释仅用于阅读源程序,系统编译程序时,会忽略其中的所有注释。

注释有以下两种类型:

①//注释一行。以“//”开始,最后以回车结束。一般作单行注释使用,也可放在某个语句的后面;

②/*... */注释一行或多行。以“/*”开始,最后以“*/”结束,中间可写多行。

(2) 空白符

空白符包括空格、回车、换行和制表符(Tab 键)等符号,可作为程序中各种基本成分之间的分隔符。各基本成分之间可以有一个或多个空白符,其作用相同。和注释一样,系统编译程序时,只用空白符区分各种基本成分。

(3) 普通分隔符

普通分隔符和空白符的作用相同,用来区分程序中的各种基本成分,但它在程序中有确定的含义,不能忽略。

Java 有以下普通分隔符:

- . (句号),用于分隔包、类或分隔引用变量中的变量和方法;
- ; (分号),是 Java 语句结束的标志;
- , (逗号),分隔方法的参数和变量说明等;
- : (冒号),说明语句标号;
- {} (大括号),用于定义复合语句、方法体、类体及数组的初始化;
- [] (方括号),用于定义数组类型及引用数字的元素值;
- () (小括号),用于在方法定义和访问中将参数表括起来,或在表达式中定义运算的先后次序。

标志符、关键字和分隔符的使用示例如下。

```
public class Example
{
    public static void main(String args[] )
    {
        int i, c;
        ...
    }
}
```

2.3 数据类型

2.3.1 Java 数据类型简介

Java 语言中的数据类型划分为基本数据类型和引用数据类型两大类。其中基本数据类型由 Java 语言定义,不可以再进行划分。基本数据类型的数据占用内存的大小固定,在内存中存入的是数值本身。引用数据类型在内存中存入的是引用数据的存放地址,并不是数据本身。

Java 语言中的数据类型分类情况如图 2-2 所示。