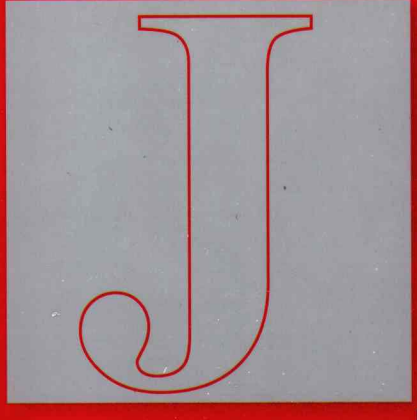
21世纪高等学校计算机专业实用规划教材

Java程序设计教程

江红 余青松 主编



清华大学出版社

21世纪高等学校计算机专业实用规划教材

Java程序设计教程

江红 余青松 主编

清华大学出版社
北京

内 容 简 介

本书主要基于 Java SE 6 SDK/Eclipse/NetBeans，讲述 Java SE 6 语言的基础知识，以及使用 Java SE 6 语言的实际开发应用实例。本书具体内容包括 Java 语言概述，数据类型、变量和常量、运算符、表达式和语句，程序流程和异常处理，数组，类和对象，继承和多态，枚举类型和注解类型，泛型，多线程编程技术，数值、日期和字符串处理，输入/输出流和文件，集合和数据结构，数据库访问技术，网络编程和通信，图形用户界面应用程序等。

本书作者结合多年的程序设计、开发及授课经验，精选大量的实例，由浅入深、循序渐进地介绍了 Java 程序设计语言，让读者能够较为系统、全面地掌握程序设计的理论和应用。

本书可作为高等学校各专业的计算机程序设计教程，也可作为广大程序设计开发者、爱好者的自学参考书。

本书配有实验和辅导教材《Java 程序设计实验指导与习题测试》，提供了大量的思考与实践练习，让读者从实践中巩固和应用所学的知识。

图书在版编目(CIP)数据

Java 程序设计教程/江红等主编. —北京：清华大学出版社，2012.11
(21 世纪高等学校计算机专业实用规划教材)
ISBN 978-7-302-28819-0

Ⅰ. ①J… Ⅱ. ①江… 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(2012)第 103094 号

责任编辑：魏江江　王冰飞
封面设计：傅瑞学
责任校对：梁　毅
责任印制：张雪娇

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课 件 下 载：http://www.tup.com.cn，010-62795954
印 刷 者：清华大学印刷厂
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：185mm×260mm　　印　　张：31.5　　字　　数：784 千字
版　　次：2012 年 11 月第 1 版　　印　　次：2012 年 11 月第 1 次印刷
印　　数：1～2500
定　　价：49.50 元

产品编号：045675-01

前言

目前，国内外众多的大学本科、专科、高职高专均采用 Java 作为程序设计课程的必修语言。Java 是较流行的编程语言之一，具有高度的安全性、可移植性和代码可重用性。

本书主要基于 Java SE 6 SDK/Eclipse/NetBeans，讲述 Java SE 6 语言的基础知识，以及使用 Java SE 6 语言的实际开发应用实例。

本书内容共分 16 章，分别讲述 Java 语言概述，数据类型、变量和常量，运算符、表达式和语句，程序流程和异常处理，数组，类和对象，继承和多态，枚举类型和注解类型，泛型，多线程编程技术，数值、日期和字符串处理，输入/输出流和文件，集合和数据结构，数据库访问技术，网络编程和通信，图形用户界面应用程序等。

本书配有实验和辅导教材《Java 程序设计实验指导与习题测试》，提供了本书的上机实验指导，以及本书各章节的习题测试和习题参考解答。

本书特点：

(1) 由浅入深、循序渐进、重点突出、通俗易学。

(2) 理论与实践相结合，通过大量的实例，讲述程序设计的基本原理，使读者不仅掌握理论知识，同时掌握大量程序设计的实用案例。

(3) 提供了大量的思考与实践练习，让读者从实践中巩固和应用所学的知识。

本书涉及的各章节所有的源程序代码和相关素材以及供教师参考的教学电子文稿和教学文档(包括 JDK 的安装与使用、Eclipse SDK 简介)均可以通过清华大学出版社的教学资料网站(www. tup. tsinghua. edu. cn)下载。

本书由华东师范大学江红和余青松共同编写。在此衷心感谢本书的编辑魏江江老师，敬佩他的高瞻远瞩和敬业精神。由于时间和编者学识有限，书中不足之处在所难免，敬请诸位同行、专家和读者指正。

编　者

2012 年 3 月

目 录

第1章 Java语言概述

Java语言是一种通用的面向对象的编程语言，被广泛用于构建各种跨平台的、安全可靠的网络应用程序。

本章要点：

- Java语言及其特点
- Java语言的编译和运行环境
- Java程序的创建、编译和运行
- Java程序的基本结构

1.1 Java语言及其特点

1.1.1 Java语言简介

1991年6月，美国Sun Microsystems公司（该公司已于2009年被甲骨文公司收购）的James Gosling等人组成开发小组，旨在开发一种用于家用电器（如电视机、冰箱、烤箱、电话等）的控制和通信的逻辑控制系统。项目最初拟采用C语言，但发现C语言和可用的API存在许多问题，为此，Gosling等结合C语言和Mesa语言的特点，创建了一种全新的语言Oak，即Java语言的前身。由于这些智能化家电的市场需求不足，该项目最后被终止。

随着Internet的发展，Sun Microsystems公司看到了Oak比较适合编写Internet应用程序，于是结合WWW的需要对Oak进行了改进和完善，并在1995年5月以Java的名称正式发布。Java伴随着互联网的迅猛发展而发展，逐渐成为重要的网络编程语言。

Java编程语言的风格与C++语言近似。Java继承了C++语言面向对象技术的核心，但舍弃了C++语言中容易引起错误的指针、运算符重载、多重继承等特性，增加了垃圾回收器功能，用于回收不再被引用的对象所占据的内存空间。Java SE 1.5中又引入了泛型编程、类型安全的枚举、不定长参数和自动装/拆箱等语言特性。

与编译执行语言（例如C++）不同，Java是解释执行语言。Java源代码首先编译成字节码，然后依赖各种不同平台上的虚拟机来解释执行字节码，从而实现“一次编译、到处执行”的跨平台特性。Java语言的解释执行特点在一定程度上会降低Java程序的运行效率，但在J2SE 1.4.2发布后，Java的执行速度有了大幅提升。

Java作为广泛使用的语言，其主要发展历史如表1-1所示。

表1-1 Java主要发展历史

发布时间	Java版本	说　明
1995/05		Java语言诞生
1996/01	JDK 1.0	JDK 1.0发布

续表

发布时间	Java 版本	说　明
1997/02	JDK 1.1	JDK 1.1 发布
1998/12	J2SE 1.2	J2SE SDK 1.2 发布。自版本 1.2 至 1.5，通常称为 Java 2。JDK 被更名为 J2SE SDK。Sun 公司发布了 Java 的 3 个版本：J2SE（Java 2 Platform，Standard Edition）、J2EE（Java 2 Platform，Enterprise Edition）和 J2ME（Java 2 Platform，Micro Edition）
2000/05	J2SE 1.3	J2SE SDK 1.3 发布
2002/02	J2SE 1.4	J2SE SDK 1.4 发布
2004/09	J2SE 5.0	J2SE SDK 5.0 发布。J2SE 1.5 被更名为 J2SE 5.0。内部版本号为 1.5
2006/12	Java SE 6	JDK 6 发布。Java 2 被更名为 Java SE 6。内部版本号为 1.6
2011/7	Java SE 7	JDK 7 发布

本书主要基于目前使用最广泛的 Java SE 6，讲述 Java 语言的基础知识，以及使用 Java 语言的实际开发应用实例。

注：2009 年 4 月，Oracle 公司宣布收购 Sun 公司。

1.1.2 Java 的特点和开发应用范围

1. Java 语言的特点

Java 是一种现代的、面向对象的、类型安全的编程语言。Java 具有下列特点：

（1）简单。Java 语言的语法与 C/C++ 语言很接近，使得大多数程序员很容易学习和使用。Java 简化了 C/C++ 中许多复杂的特性，如运算符重载、多继承等。特别地，Java 语言不使用指针，并提供了自动的垃圾收集，从而避免了直接操作内存的复杂性和风险性。

（2）面向对象。Java 支持数据封装、继承、多态和接口。所有的变量和方法，包括 main 方法（应用程序的入口点），都封装在类定义中。类可以直接从一个父类继承（不支持多重继承），可以实现任意数量的接口。

（3）分布式。Java 语言提供了用于网络应用编程的类库，可以处理 TCP/IP 及其他网络协议，广泛用于开发各种分布式的网络应用程序。

（4）健壮性。Java 的强类型机制、异常处理、垃圾自动收集等是 Java 程序健壮性的重要保证。Java 的安全检查机制使得 Java 更具健壮性。

（5）安全性。Java 通常被用在网络环境中，为此，Java 提供了一个安全机制以防恶意代码的攻击。

（6）体系结构中立。Java 程序源文件（扩展名为.java 的文件）在 Java 平台上被编译为体系结构中立的字节码格式（扩展名为.class 的文件），然后可以在实现该 Java 平台的任何系统中运行，从而实现“一次编译、到处执行”的跨平台特性。

（7）可移植性。Java 的可移植性来源于体系结构中立性。Java 系统本身也具有很强的可移植性，Java 编译器是用 Java 实现的，Java 的运行环境是用 ANSI C 实现的，从而保证了整个 Java 系统的可移植性。

（8）解释型。Java 程序在 Java 平台上被编译为字节码格式，然后可以在实现该 Java 平台的任何系统中运行。在运行时，Java 平台中的 Java 解释器对这些字节码进行解释执行，执行过程中需要的类在连接阶段被载入到运行环境中。

（9）高性能。解释型语言的缺点是性能低，但随着 JIT（Just-In-Time）编译器技术的发展，Java 的运行速度越来越接近于 C++。

(10) 多线程机制。Java 语言支持多个线程的同时执行，并提供多线程之间的同步机制。Java 线程可以设定优先级。使用多线程机制，可以编写各种高性能的应用程序，例如，使用线程处理同时进行用户输入的监视、后台任务的执行，以及并发输入流的处理等。

(11) 动态性。Java 语言的设计目标之一是适应动态变化的环境。Java 程序需要的类能够动态地载入到运行环境，也可以通过网络来载入所需要的类，有利于软件的升级。另外，Java 中的类有一个运行时刻的表示，能进行运行时刻的类型检查。

2. Java 语言开发应用范围

Java 语言主要用来构建在 Java 的运行环境(Java Runtime Environment，JRE)上运行各种安全、可靠的应用程序。使用 Java，可以创建下列类型的应用程序和服务：

- 桌面应用程序；
- Java Applet 小应用程序；
- Web 应用程序；
- 企业应用程序；
- 移动应用程序。

1.2 Java 语言的编译和运行环境

1.2.1 Java 语言与 Java 平台

Java 语言的一个重要特性是跨平台特性，即“一次编译、到处执行”。要实现该目标，必须提供相应的 Java 运行平台，在实现相应 Java 平台的任何系统中运行 Java 程序。根据应用范围，Java 运行平台分为 3 个体系：

(1) Java SE(Java Platform，Standard Edition，Java 平台标准版)。Java SE(旧版本称为 J2SE)提供标准的 SDK(Software Development Kit)开发平台(旧版本称为 JDK)。Java SE 允许开发和部署 Java 桌面应用程序、Java Applet 程序、低端服务器应用程序。Java SE 程序开发环境提供了开发与运行 Java 软件的编译器等开发工具、类库及 Java 虚拟机。

(2) Java EE(Java Platform，Enterprise Edition)。Java EE(旧版本称为 J2EE)是在 Java SE 的基础上构建的。Java EE 用于开发和部署可移植、健壮、可伸缩且安全的服务器端应用程序。Java EE 提供 Web 服务、组件模型、管理和通信 API，可以用来实现企业级的面向服务体系结构(Service-Oriented Architecture，SOA)和 Web 2.0 应用程序。

(3) Java ME(Java Platform，Micro Edition)。Java ME(旧版本称为 J2ME)是一种精简的 Java 运行环境，用于各种嵌入式设备，如移动电话、PDA、电视机顶盒等。Java ME 包括灵活的用户界面、健壮的安全模型、许多内置的网络协议，以及对可以动态下载的应用程序的支持。

1.2.2 Java SE

Java SE 程序开发环境包括 3 个部分：开发工具、类库及 Java 虚拟机。

1. Java SE 开发工具

Java SE 6 程序开发环境提供了开发与运行 Java 软件的编译器等开发工具，主要的组件包括以下内容。

(1) javac：编译器。将源程序转成字节码。

(2) jar：打包工具。将相关的类文件打包成一个文件。

(3) javadoc：文档生成器。从源码注释中提取文档。

(4) jdb：调试工具(Java debugger)。用于调试 Java 应用程序。

2. Java 类库

Java 提供了强大的应用程序编程接口(Application Programming Interface,API),即 Java 类库。Java 类库是系统提供的已实现的标准类的集合,包括大量类和接口,可以帮助程序员进行字符串处理、绘图、数学计算、网络应用等方面的工作。合理充分地利用 Java 类库,可以高效率地开发各种应用程序。

Java 类库中的类和接口大多封装在特定的包里,每个包具有自己的功能。Java 类库的主要包如表 1-2 所示。

表 1-2 Java 类库的主要包

包 名	主 要 功 能
java. applet	提供用于创建 Applet 所需的类
java. awt. *	提供用于创建用户界面以及绘制和管理图形、图像的类
java. beans. *	提供用于开发 beans 的类
java. io	提供通过数据流、序列化和文件系统实现系统输入和输出的类
java. lang	提供 Java 编程语言的基础类
java. math	提供用于执行任意精度整数、小数算法的类
java. net	提供用于实现网络应用程序的类
java. nio. *	提供用于高效输入和输出的非堵塞型 I/O 类
java. rmi. *	提供用于远程方法调用的类和接口
java. security. *	提供用于安全框架的类和接口
java. sql	提供处理 Java 数据源中的数据的类
java. text. *	提供以与自然语言无关的方式来处理文本、日期、数字和消息的类和接口
java. util. *	提供各种常用工具包(包括集合框架、国际化等)
javax. swing. *	提供一组“轻量级”组件构建图形用户界面应用程序
javax. xml. *	提供用于 XML 处理的类
javax. naming. *	提供用于访问命名服务的类

3. Java 虚拟机

Java 虚拟机(Java Virtual Machine,JVM)是可运行 Java 字节码的假想计算机,在实际的计算机上通过软件模拟来实现。Java 虚拟机包括一套字节码指令集、一组寄存器、一个栈、一个垃圾回收堆和一个存储方法域。Java 虚拟机是一种用于计算设备的规范,可用不同的方式(软件或硬件)加以实现。编译虚拟机的指令集与编译微处理器的指令集非常类似。

Java 程序源文件被编译为体系结构中立的字节码格式。执行 Java 程序时,Java 虚拟机将加载并解释执行字节码程序。只要根据 JVM 规格描述将解释器移植到特定的计算机上,就能保证经过编译的任何 Java 代码能够在该系统上运行,从而实现“一次编译、到处执行”的跨平台特性。

1.2.3 Java 的运行环境

Java 的运行环境(Java Run Environment,JRE)为 Java 应用程序提供运行环境,JRE 包括 Java 虚拟机(JVM)和标准类库(Class Library)。

注:Java SE SDK 包含一个 JRE,也可安装独立的 JRE。

1.2.4 Java 的开发环境

要开发 Java 应用程序,可以使用文本编辑器(如 Notepad)编写代码,并使用 Java SE 6 中的编译器进行编译、运行,也可以使用集成开发工具(如 Eclipse、NetBeans)。

1. JDK

JDK(Java Development Kit)包括开发人员编写、生成、测试和部署 Java 应用程序时所需要的一切，如编译器、运行调试工具、文档及示例等。有关 JDK 的下载和安装配置，请参见附录 A。

2. Eclipse

Eclipse 是著名的免费且跨平台的集成开发环境(Integrated Development Environment，IDE)，集设计、编辑、运行、调试等多种功能于一体。使用 Eclipse，可以高效地开发各种 Java 应用程序。

Eclipse 本身只是一个框架平台，通过插件，可以实现不同的开发功能。Eclipse 主要用于 Java 语言开发，但通过插件使其作为 C++、Python、PHP 等其他语言的开发工具。许多软件开发商以 Eclipse 为框架开发自己的 IDE。Eclipse 是目前使用最广泛的 Java 集成开发环境。有关 Eclipse 的下载与安装配置，请参见附录 B。

3. NetBeans

NetBeans 是一款开源集成开发环境，可用于 Java、C/C++、PHP、Python、Ruby 等程序的开发。NetBeans 是一个开放框架，是可扩展的开发平台，可以通过扩展插件来扩展功能。NetBeans 也是目前使用十分广泛的 Java 集成开发环境，许多 Java 文档和示例使用的就是 NetBeans 集成开发环境。

1.3 创建简单的 Java 程序

1.3.1 Hello World 程序

使用任意编辑软件(如记事本 Notepad. exe)创建程序文件 HelloWorld. java(Java 源文件的扩展名通常是. java)。

【例 1.1】 创建 Hello World 程序。

```
01  //javabook\chapter01\HelloWorld.java A "Hello World!" program
02  public class HelloWorld {
03      public static void main(String[] args) {
04          System.out.println("Hello World!");          //显示字符串
05      }
06  }
```

1.3.2 代码分析

行 01 为注释。

行 02 定义了用户自定义类 HelloWorld。

行 02 和行 06 的花括号对定义了代码块，其中的代码为类 HelloWorld 的实现。

行 03 定义了类 HelloWorld 的一个成员，即名为 main 的方法。main()方法是使用 static 修饰符声明的静态方法，将作为程序的入口点。

行 03 和行 05 的花括号对定义了代码块，其中的代码为 main()方法的实现。

行 04 通过调用 System. out. println("Hello World!")，在控制台上输出字符串：Hello World!。System. out 是 System 类中 out 变量的全限定名称，代表标准输出设备，即控制台屏幕。System 是 Java 常用的类，本书范例也大量使用。

1.3.3 编译和运行结果

可以使用命令行“javac HelloWorld. java”调用 Java 编译器编译 HelloWorld 程序，如图 1-1

所示。编译后将产生一个名为 HelloWorld.class 的文件，即字节码的类文件。如果程序中产生错误(例如拼写错误。特别强调，Java 源代码大小写敏感，如果上例中的 public 误拼写为 Public，则会产生编译错误)，则编译器会产生错误信息。

可以使用命令行“java HelloWorld”调用 Java 解释器运行这个程序，如图 1-2 所示。当此应用程序运行时，输出结果为：Hello World!。

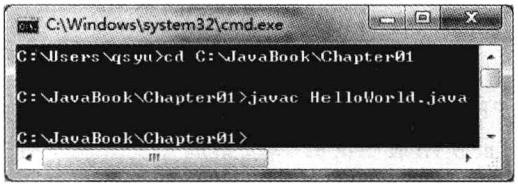


图 1-1　编译 HelloWorld 程序

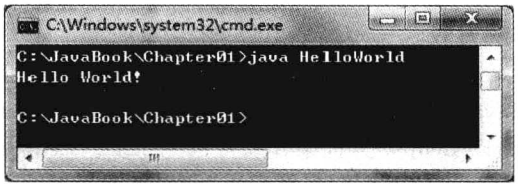


图 1-2　运行 HelloWorld 程序

使用 Java 解释器运行程序时，Java 解释器会在类路径(CLASSPATH 系统环境)中查找类的字节码文件。例如：执行命令“java HelloWorld”将在当前目录中查找 HelloWorld.class 字节码文件。如果找不到字节码文件，则会产生错误。

```
java HelloWorld.java
```

上述命令行会产生如下错误(Java 解释器会查找 HelloWorld.java.class)：

```
Exception in thread "main" java.lang.NoClassDefFoundError: HelloWorld/java
```

再如，执行命令“java HelloWorld.class”会产生如下错误信息(Java 解释器会查找 HelloWorld.class.class)：

```
Exception in thread "main" java.lang.NoClassDefFoundError: HelloWorld/class
```

有关 Java 程序的编译器的详细信息，请参见附录 A。

注：一个源文件(扩展名为.java)可以包含多个类，编译时每个类都将生成一个字节码的类文件。另外，各嵌套类也将分别生成对应的字节码的类文件(请参见 6.8 节)。

1.4　Java 程序的基本结构

1.4.1　程序结构

Java 程序由一个或多个源文件(扩展名为.java)组成，每个源文件又称为编译单元。在编译单元(源文件)中可声明类和接口，类包含成员，例如成员变量和方法等。通过 Java 编译器，编译单元(Java 源文件)被编译成字节码文件(扩展名为.class)，也可选择通过工具 JAR 把字节码文件打包为 JAR。

编译单元(源文件)中声明的类型按包组织成层次结构。Java 程序的基本结构如下所示：

```
//Java 程序的基本结构
import packagename;                          //导入包
package yourpackagename;                    //声明包
public class YourClass  {                   //声明类
//类体 声明字段、方法等
}
interface IYourInterface {                  //声明接口
//接口体
}
```

一般情况下，一个编译单元(源文件)声明一个类(或接口)，如果类的访问修饰符为 public，则源文件的文件名必须和文件中声明的 public 类的名称保持一致。

注：在一个编译单元(Java 源文件)中，只能声明一个 public 类(与 Java 源文件名相同)；如果在同一个源文件中声明了多个类，则其他类不能为 public。

1.4.2 包

Java 类库包含大量的类型，用户也可以自定义类型。为了有效地组织 Java 程序中的类型并保证其唯一性，Java 引入了包的概念，从而最大限度地避免类型重名错误。

在编译单元(Java 源文件)中定义类或接口时，可以声明其所在的包。Java 程序中，类型由指示逻辑层次结构的完全限定名(fully qualified name)描述。例如，java. util. Date 表示 java 包的子包 util 中的 Date 类。

1. 定义包

Java 程序中使用 package 关键字声明包。声明格式如下：

```
package 包名称;
```

其中，包名称的一般格式如下：

```
<company>.(<product>|<technology>)[.<feature>][.<subPackageName>]
```

包名称一般约定为小写字母，这样就可以与类的名称(一般约定为首字母大写)区分开来。例如，Acme 公司(假设其域名为 acme. com)的 ERP 项目中关于数据访问的类型可以组织到包 com. acme. erp. data 中。使用域名作为包命名的一部分，可以有效地组织项目中定义的类和接口，并与其他公司提供的类库区分开来。

package 语句必须是源程序文件中第一个非注释、非空白行的语句。

包的成员可以包含子包、类、接口。例如，java 包中包含子包 awt、applet、io、lang、net 和 util。java. util 包中包含 java. util. jar 等子包、java. util. Date 等类、java. util. Formattable 等接口。

一个包(子包)的成员不能重名。例如，如果包 javabook 定义了子包 test，则不能定义名称为 test 的类或接口。

注意：如果源代码中没有指定包，则使用默认包。默认包一般用于编写实验性质的临时程序，例如 Java 学习程序。在实际项目开发中，除非简单的小程序，一般不推荐使用默认包。

一般情况下，包对应于文件系统的目录，子包对应相应目录下的子目录，而类文件(. class)则对应目录下的字节码文件。

【例 1.2】 包示例 1：默认包。

```
//假设当前路径为：C:\javabook\chapter01
//源文件路径：C:\javabook\chapter01\HelloWorld1.java
//编译：javac HelloWorld1.java   -> 编译结果：HelloWorld1.class
//运行：java HelloWorld1
public class HelloWorld1 {
    public static void main(String[] args) {
        System.out.println("Hello World!");        //显示字符串
    }
}
```

【例 1.3】 包示例 2：简单包。

```
//假设当前路径为：C:\javabook\chapter01
//源文件路径：C:\javabook\chapter01\qsyu\HelloWorld.java
```