JAVA 语言基础教程

刘彦明 李鹏 编

西安电子科技大学出版社

Java 语言基础教程

刘彦明　李　鹏　编

西安电子科技大学出版社

1998

内 容 简 介

本书主要介绍了 Java 语言的基础知识和基本的程序设计方法。Java 语言是一种面向对象的程序设计语言，它特别适合于 Internet/Intranet 上的应用程序开发，同时具有普通计算机程序设计语言的特点。

本书从 Java 语言的基本语言要素开始，由浅入深、按部就班地介绍了 Java 语言的基础知识，特别对 Java 语言中新出现的概念(例如线程、异常等)作了详细的介绍，同时用较多的篇幅介绍了 Java 语言的输入、输出、图形用户界面、网络和 Applet 等的程序设计方法。

本书不仅是一本 Java 语言的基础教程，而且兼顾了具有一定 Java 语言基础的程序设计人员的需要，是一本比较系统的 Java 语言参考书。

本书是作者在从事 Java 开发的基础上编写出来的，具有通俗、易懂的特点和很高的参考价值。本书注重实用性，在书中给出了大量实例程序，使读者能够通过例子和上机实践很快掌握 Java 语言。

本书适合于电子类各专业的本科生、研究生、大学教师和工程技术人员使用，特别适合于从事网络和 Internet/ Intranet 开发的科技工作者使用。

Java 语言基础教程

刘彦明　李　鹏　编

责任编辑　杨　兵

出版发行　西安电子科技大学出版社

(西安市太白南路 2 号)

邮　编　710071

电　话　(029)8227828

经　销　新华书店

印　刷　西安电子科技大学印刷厂

版　次　1998 年 11 月第 1 版

1998 年 11 月第 1 次印刷

开　本　787 毫米×1092 毫米　1/16　印张 21.25

字　数　503 千字

印　数　1～4 000 册

定　价　21.50 元

ISBN 7-5606-0666-0/TP・0337

前　言

Java 是 Sun Microsystem 公司推出的新一代面向对象和面向网络的程序设计语言，特别适合于 Internet/ Intranet 上的应用软件开发，因此也把 Java 语言称为新一代网络程序设计语言。Java 语言将面向对象、多线程、安全和网络等特征集于一身，为软件开发人员提供了良好的程序设计环境，因此掌握 Java 语言对每一个软件开发人员都是至关重要的。为此，作者在编写《Java 语言及其程序设计》和《Java 类库及其实例大全》的基础上编写了本书，希望能够帮助广大软件开发工作者在 Java 应用方面有一个长足的进步。

本书主要包括三方面的内容：一是 Java 语言的基础知识，主要内容包括 Java 语言概述，Java 语言数据类型、运算符和表达式，Java 语言的流程控制语句，Java 语言的面向对象程序设计、数组和字符串，Java 程序的系统环境、异常处理和线程；二是 Java 语言的常用 API，主要介绍输入输出流、Java 网络和图形用户界面的程序设计方法；三是 Java 语言的 Applet，主要介绍如何设计 Applet 程序。

本书内容安排如下：

第 1 章　Java 语言概述，主要介绍 Java 语言的特点和 Java 语言的编程环境。

第 2 章　Java 数据类型、运算符和表达式，主要介绍 Java 语言的基本要素，包括 Java 语言的数据类型(整型、浮点型、字符型和布尔型等)、Java 语言的操作符和 Java 语言的表达式。

第 3 章　Java 语言的流程控制语句，主要介绍 Java 语言的条件控制语句、循环控制语句和转移控制语句，最后给出了几个简单的例子程序。

第 4 章　Java 语言的面向对象程序设计，主要介绍面向对象程序设计的基础知识，包括对象、类、包和接口以及它们的使用方法。

第 5 章　数组和字符串，主要介绍数组和字符串的使用方法。

第 6 章　Java 程序的系统环境，主要介绍 System 类中的标准输入输出、系统属性和命令行参数的使用方法。

第 7 章　异常处理，主要介绍异常处理的基本概念、异常的抛出、捕获和处理，以及构造自己的异常类等内容。

第 8 章　线程，主要介绍线程的基本概念、多线程程序设计、线程的控制

和线程的程序设计方法，以及线程同步、线程通信等。

第 9 章　输入、输出流，主要介绍文件的输入、输出和管道的输入、输出，以及顺序输入流、内存读写和过滤流等内容。

第 10 章　Java 网络程序设计，主要介绍网络程序设计中使用到的类和网络程序设计的基本方法以及网络安全等内容。

第 11 章　图形用户界面，主要介绍图形用户界面程序设计中用到的类和图形用户界面程序设计的基本方法。

第 12 章　Java applet 程序设计，主要介绍 Java applet 程序的设计方法，强调了 Java 程序和 Java applet 程序设计的不同，并给出了相应的例子程序。

本书不仅是一本 Java 语言的基础教程，而且兼顾了具有一定 Java 语言基础的程序设计人员的需要，是一本比较系统地介绍 Java 语言的书籍。

本书是作者在从事 Java 开发的基础上编写出来的，具有通俗、易懂的特点和很高的参考价值。本书注重实用性，在书中给出了大量实例程序，使读者能够通过例子和上机实践很快掌握 Java 语言。

本书适合于电子类各专业的本科生、研究生、大学教师和工程技术人员使用，特别适合于从事网络和 Internet/ Intranet 开发的科技工作者使用。

鉴于作者水平有限，书中难免有不妥之处，敬请广大读者批评指正。如果读者有好的建议、要求、批评等，请与作者联系，电子邮件地址是：ymliu @xidian. edu. cn.

刘彦明
谨识于西安电子科技大学
计算机通信教研室

目　录

Java 语言概述

第 1 章

Java 语言是由 Sun MicroSystem 公司开发的新一代面向对象的程序设计语言，是网络程序设计的最优秀工具，特别适合于建立分布式计算应用程序。它使 Internet 的作用从通信工具扩展到能够运行应用程序的网络，这一突破使企业可以在 Internet 上进行全方位的业务服务和实时信息交换。Java 语言还简化了软件代理(Software agent)的结构，软件代理是指在网络上交换信息或运行远程计算机上的程序。在不远的将来，用户也许可以在本地计算机上向 Internet 发送软件代理，以便到世界各地搜寻所需的信息或进行有限服务。

1.1 Java 语言的历史背景和发展过程

Java 语言是由 Sun 公司开发的，那么，Sun 为什么要开发这一语言呢？这要从 Sun 的 Green 开发项目讲起。

Sun 公司主要从事工作站及其软件的开发，到了 1990 年，由于受到 PC 机对工作站市场的巨大压力，Sun 公司计划在消费类电子产品方面开拓市场，因此成立了由 James Gossling 领导的 Green 开发小组，其首要目标是开发一个对家用电器进行集中控制的装置，为此就必须开发相应的软件。

开发软件的首要问题是选择开发环境——编程语言。由于考虑到现有语言不适合用于此项目，于是自行设计并开发了一种新的语言，并依此开发了一个称为 Oak 的软件。由于该项目的结果不令人满意，于是 Sun 公司几乎决定解散该项目，其中的许多人被转到其他开发项目中。

到了 1994 年，在 Internet 上出现了万维网(WWW)，James Gossling 等人立即意识到他们所开发的语言非常适合于在 WWW 上应用，为此，他们对该语言进行了改进，定名为 Java，并在此基础上实现了 HotJava 浏览器，一下子使得 Java 成为一种新的革命性的语言，得到了大多数著名计算机公司的支持，因此其后 Java 得以进一步完善和加强。

HotJava 的开发对 Java 语言的发展是至关重要的，它的意义不仅在于使 Java 语言更加成熟和完善，更为重要的是它向世界展示了 Java 的能力，是 Java 走向世界的里程碑。

1995 年，Sun 公司在 Sun World 会议上正式发布了 Java 技术。从此，Java 的知名度如日中天，甚至在 1996 年 1 月 Java 编译器第一版发行之前，Java 已经成为 Internet 开发的行业标准。

1996 年，许多著名的计算机软、硬件公司从 Sun 得到了 Java 技术许可，并将其加入到自己的桌面产品、操作系统和开发工具中。

到此时，Java 已经形成，而且还在继续发展。在此过程中，Java Soft(Sun 公司为推动 Java 的进一步发展而专门成立的 Java 开发公司)先后推出了 Java Development Kit 1.0、1.1 和适合于不同应用领域的工具包。在本书中，以 JDK1.1 为原型介绍 Java 语言。

也许有的读者会问：本书中介绍的内容适合将来的 Java 版本吗？答案是：一定适合，因为本书主要介绍 Java 的基础部分，而这一部分内容在将来的发展中是不会有大的改变的。

1.2 Java 语言的特点

Java 语言发展如此迅速并能很快被众多计算机公司所承认，得益于 Java 语言的显著特点。

1.2.1 简单性

Java 语言是一种面向对象的程序设计语言，它继承了 C/C++语言的风格，但又抛弃了 C++中一些并非必要的功能，如头文件、预处理、指针、联合、结构、隐式类型转换、运算符重载和多重继承等语言成分，使 Java 语言比任何一种面向对象的语言都简单。

1.2.2 面向对象

Java 语言是一种完全的面向对象语言，除数字型、布尔型和字符型等三个基本数据类型外，其他数据类型都是用类来定义的。Java API 中的所有构件都是用类提供的。

Java 语言的程序代码是用类来组织的，由类来描述程序中所有对象的状态和行为。Java 语言抛弃了 C++中的 C 特征。Java 语言支持类的继承特性，只能是单继承。

因此，Java 语言是一种纯的面向对象程序设计语言。

1.2.3 安全性

安全性是网络环境、分布环境，特别是 Internet 环境中的最重要的问题。网上的用户最担心两件事：信息被窃取和计算机系统被黑客(hacker)破坏，而 Java 语言的内置式安全机制解决了这两个问题。

Java 语言的安全机制是通过下面三个基本组成部分实现的：

1. 字节代码验证器

Java 编译器虽然保证了 Java 源代码不违反安全规则，但是从其它地方引入浏览器的代码段就不一定满足 Java 语言的安全规则，必须对其进行字节码验证，因此在 Java 运行系统中引入了字节代码验证器，以便在运行前对字节码进行安全检验，确保其遵循 JVM(Java

虚拟机)的访问限制，这样引入的字节代码就不会访问非授权的数据或资源。

2. 安全管理器

安全管理器实现 JVM 的安全性策略，安全性策略确定 JVM 可以在什么条件下进行什么活动。例如，Java 语言有能力进行文件的输入和输出，但首先必须经过安全管理器的检查。这样就能确保 Java 语言的文件访问不会对文件系统造成损害。

3. 类装入器

从网络上获取类时，类装入器负责把来自不同服务器的类相互分开，并与本地类区别开来。通过这样的分离，类装入器就可以防止网上装入的类假装成标准的内置/本地类，或干扰从其他服务器装入的类的运行。

在专用网络中，需要文件访问和网络上的任意访问来解决业务要求。例如，在一个专用的网络上实现客户/服务器数据库访问应用程序时，可能需要 Java applet 程序与多个服务器建立连接。标准的安全管理器禁止这种行为，因为这种访问在公共网上会危及安全。因此，应用程序开发者可以对网络上的每台计算机改变安全管理器源代码(重载安全管理器)，并重新编译到 Web 浏览器中，这样就可以用户化专用网络的安全性策略。覆盖安全管理器要十分小心，否则公共网上的敌意小程序就会乘机入侵，造成损失。

总之，Java 语言的内部安全机制确保 Java 程序在 JVM 规则下运行，防止未授权的程序访问包含信息的资源和危及客户机的完整性。

1.2.4 体系结构中立

Java 语言的体系结构中立(也称移植性)体现在两个方面：字节码编译和标准开放。

1. 字节码编译

Java 语言的编译器将 Java 程序编译为字节代码，该字节代码不是为某种特定的机器定义的，不能在某个具体的平台上执行，而需要 Java 运行系统中的解释器进行解释执行。因此，运行 Java 程序需要两个步骤：一是把 Java 源程序编译为字节代码；二是由解释器解释执行字节代码。

为了实现字节码的结构中立，还制定了一套完全统一的语言标准，如 Java 语言的基本数据类型不会随目标机的变化而改变，整型数总是 32 位，长整数总是 64 位等。为了做到图形用户界面的独立，提供了对等构件包，以支持不同硬件平台，这样就可以做到用户界面与平台无关。

2. 开放式标准

Java 语言的跨平台特征是指 Java 程序不需要针对每个平台进行编译，编译过的 Java 程序可以在不同的平台上运行。当然，要实现这样的跨平台，每个平台上必须装有支持 Java 语言的核心集，即 Java 运行系统。

1.2.5 多线程

Java 语言提供了语言级的多线程机制。Java 程序可以包含多个运行线程，而使每个线程完成不同任务。例如，可以让一个线程进行网上数据下载，而另一个线程对数据进行计算，第三个线程负责与用户进行交互。当然，多线程间必须实现必要的同步。

Java 语言的同步机制采用 C. A. R. Houre 提出的，并在许多新型操作系统中得到广泛应用的线程和临界区保护机制。

多线程通常用于完成下面的功能：

1. 维护用户响应

如果应用程序需完成一个费时的任务，则可以使用多线程防止任务进行时不对用户进行响应。

2. 多任务

采用多线程就可以很容易地在一个程序中实现多个任务。为了提高用户上网的效率，可以建立多个线程，使每个线程访问不同的数据源。

3. 多用户应用程序

多线程通常用于建立服务器应用程序，服务器应用程序可以为每个客户建立一个线程并为该客户服务。

1.2.6 内存管理

在 Java 语言中，开发人员不需关心内存管理问题。Java 运行系统会自动为应用程序分配必要的存储空间，并自动对不再使用的内存由内存收集器进行回收，这就使 Java 语言编程更加简单，且减少了因内存管理而出错的可能性。

1.2.7 分布式和动态特性

Java 语言扩展了动态库的概念，允许 Java 应用程序动态装入本地类和运程(网上)类，这使 Java 程序既是动态的又是分布式的。

一般情况下，可移植性、安全性和动态性总是以牺牲性能为代价的，但 Java 语言很好地实现了性能和以上特点间的权衡。

1.3 Java 简单应用程序

前面介绍了 Java 语言的特点，现在通过例子来说明 Java 程序的编写、编译、运行全过程，以期对 Java 软件开发过程有一个全面的了解。

1.3.1 Java 源程序

Java 源程序可以用简单的文本编辑器建立，也可以使用集成开发环境中的编辑器建立。下面给出 HelloWorld 的 Java 源程序：

```
class HelloWorld {
    public static void main(String argv[ ]){
        System. out. println("Hello World!");
    }
}
```

该程序编译运行后输出下面一行信息：

Hello World!

这个程序是最简单的 Java 程序，对于熟悉 C/C++的人来说，还是比较陌生的。下面对该程序进行必要的说明，但不要求读者完全搞懂它，所以不必关心每个关键字的含义，只需对 Java 程序结构有所了解就可以了。

一、类定义

在 Java 语言中，所有方法(函数)和变量都存在于类和对象(类的实例)中。Java 语言不支持全局变量，因此任何 Java 程序的基干都是类定义。

在上面所示的代码中，class HelloWorld 开始了一个类定义。类作为面向对象语言(例如 Java)的基本结构单元，是描述与类的实例相关联的数据与行为的模板。当你实例化一个类时，就创建了一个对象。与类或对象相关联的数据存储在变量中，与类或对象相关联的行为由方法执行。

在 Java 语言中，类定义的最简单形式是：

```
class ClassName {
  //类体
}
```

关键字 class 为名为 ClassName 的类开始类定义的标志。这个类的变量和方法用开始和结束类定义块的花括号包含。"HelloWorld"应用程序没有变量，只有一个 main( )方法。

二、main( )方法

每个 Java 应用程序的首脑是 main( )方法。当 Java 解释器运行一个应用程序时，就确定了想运行的类名。解释器执行在类中定义的 main( )方法，mian( )方法控制程序流程，分配所需资源，运行其他任何为应用程序提供功能的方法。

每个 Java 应用程序必须包含一个 main( )方法，其声明方式如下所示：

```
public static void main(String args[ ]){
    // Java 语句
}
```

在 Java 语言中，main( )方法与 C 语言和 C++语言中的 main( )函数类似。在执行 C 或 C++程序时，系统通过首先调用它的 main( )函数来启动程序，然后 main( )函数调用程序中的其他函数。

类似地，当 Java 解释器执行应用程序时是从调用类的 main( )方法开始的。main( )方法能调用所有其他方法。

注意，在保存 Java 程序时应用类名 .java 来保存。HelloWorld 的 Java 程序应保存为 HelloWorld.java。

1.3.2 编译 Java 源程序

编译 Java 源程序的工具是 Java 编译器，它把 Java 源程序转换为 Java 字节代码。可以在 Dos Shell 提示符下输入如下命令来编译 HelloWorld 程序：

javac HelloWorld. java

该命令执行后，如果无错误提示信息，那么就会产生 HelloWorld. class 文件。该文件是经 Java 编译器 javac 编译得到的字节代码，它可以在任何支持 Java 的平台上运行，如果读者有兴趣可以亲自试一试。

1.3.3 执行 Java 程序

经编译后得到的 HelloWorld. class 就是可运行的字节代码，可以用 Java 解释器来执行它。在 Dos Shell 提示符下输入如下命令来解释执行 HelloWorld 程序：

java HelloWorld

其运行结果为.

Hello World!

1.3.4 Java 应用程序的一般结构

Java 应用程序一般由三部分组成：

• package 语句——定义文件中的类所属的包。

• import 语句——通过类名或包名引用已有的类(如 API 中的类)，使程序能够直接使用引入的类或包中的类。

• class 语句——定义各种类。

其中，package 和 import 语句是可选的，而 class 语句则是必需的。在 Java 源程序中，这三部分的出现顺序是 package 语句，import 语句，最后是 class 语句。

还有一点值得注意的是 class 语句中必须包括 main( )方法，否则 Java 应用程序就不能通过编译。

Java 应用程序的一般结构如下所示：

```
packagt Java. Myclass. example;      //包说明
import java. awt. Panel;             //引入 java. awt. penel 类
import java. net. * ;                //引入 java. net 包
class JavaExample extends Panel implements Runnable {
  //其他变量和方法说明
public static void meinl string argi[ ]){
  //main 方法体
  }
}
```

以后所看到的 Java 应用程序都类似于上述程序。

1.4 Java 开发工具和上机步骤

1.4.1 Java 开发工具

Java 开发工具是 Java Development Kit(JDK)的中文译名，是 Java 应用程序的开发环境。它由一个标准类库和一组建立、测试、运行和建立文档的 Java 实用程序组成，其核心是 Java 应用程序接口(API)。Java API 在后续章节中介绍。

Java 开发工具包括七种主要实用程序：

- javac——Java 编译器，将 Java 源代码转换成字节代码。
- java——Java 解释器，运行 Java 应用程序的字节代码。
- appletviewer——Java applet 程序观察器，用来执行 Java applet 程序的字节代码。
- javaDoc——根据 Java 程序及其注释生成 HTML 文档。
- jdb——Java 调试器，用来调试 Java 应用程序。
- javah——产生可以调用 Java 方法的 C 模块或建立被 Java 应用(小)程序使用的 C 模块的头文件。
- javap——Java 反汇编工具。

以上的 JDK 实用程序都有相应的查错程序。查错程序的末尾加一个“_g”，例如 javac_g、java_g 等。

另外，与 C/C++集成开发环境类似，Java 也有集成开发环境，如 Javasoft 公司的 Workshop、Microsoft 公司的 Visual J++以及 Symetec 公司的 Visual Cafe 等，这些集成开发环境都集编辑、编译、调试、运行于一身，具有简单、直观、易用等优点。

1.4.2 上机步骤

对于初学者来说，Java 开发工具中的实用程序并不是都必须掌握的，但至少得掌握两个实用程序，即 javac 和 java，据此就可以上机进行 Java 实践了。

由于 Java 语言是 32 位的应用程序设计语言，因此要求用户的 PC 机上必须装有 32 位的操作系统，如 Windows 95 或 NT。为了更好地运行你的程序，你的 PC 机应至少有 8 MB 的内存，推荐使用 16 MB 内存，当然有更大的内存会更好。本书主要以 Windows 95/NT 为环境进行介绍，在此环境下你就可以开始 Java 之旅，其具体过程为：

(1) 用文本编辑器编辑你的 Java 应用程序。

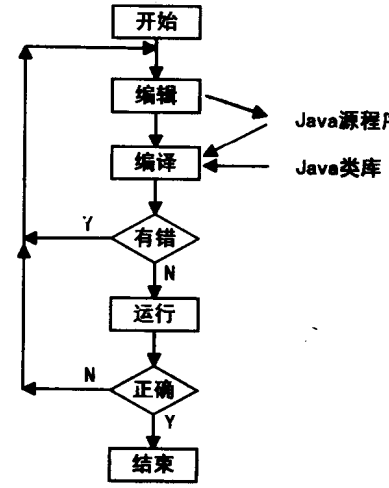

图 1-1

(2) 用 javac 编译你的 Java 应用程序。

(3) 如果有错，回到(1)改正错误，否则转(4)。

(4) 用 java 运行你的 Java 应用程序。

(5) 结果正确吗？若正确，则结束本次 Java 之旅；否则返回(1)。

除第一步必须在编辑器中进行外，其他各步都必须在 Dos Shell 下完成。上机过程如图 1-1 所示。

1.4.3 javac 和 java 解释器的进一步讨论

一、javac

Java 语言编译器的编译格式为：

javac [选项]文件名.java

javac_g [选项]文件名.java

javac 将扩展名为.java 的源程序编译成字节代码，传递给 javac 的文件中的每一个定义好的类，它们都产生各自的字节代码，并单独保存在各自的类名.class 文件中，javac 把它们都放在与.java 文件相同的子目录下。如果在编译的过程中，javac 找不到出现的某个类的定义，则按_classpath 选项提供的路径去查找。

javac_g 是 javac 的一个未优化版本，它适合于像 gdb 或 dbx 之类的调试器调试。

选项 -classpath <路径；路径；...>，定义 javac 去查找用户定义的某些类时的路径，各路径间以“;”分隔。

选项-d <目录；目录；...>，定义产生的 class 层次的根目录。

选项-g，产生调试信息，使字节代码包含行号和局部变量信息以供调试用，这是缺省方式。

选项-ng，不产生调试信息，字节代码不能用于调试。

选项-nowarn，不产生警告信息。

选项-O，代码优化。

选项-verbase，使编译和链接程序打印出源文件及字节代码文件的信息。

二、java 解释器

编译后的 Java 字节代码是无法在机器上直接运行的，需要由解释器对其解释执行。

格式：

java [选项] 类名 <参数>

java_g [选项] 类名 <参数>

类名是要运行的类的名字，它必须包含所在的组件(package)的名字(例如 java.lang.String)，只有在未定义组件名，即类缺省地放在一个无名的组件内时，才可直接写类的名字。

类名后的参数是传递给类而不是传递给解释器的。

java 解释器对扩展名为.class 的字节代码解释执行。类名.class 文件中必须包含一个 main( )方法，执行即从 main( )开始。由于 main( )在一个类内定义，但是却不能由这个类