面向对象的 Java 语言程序设计

孟祥武　张玉洁　编著

北京邮电大学出版社
·北京·

内 容 提 要

本书介绍了 Java 语言的基本语法和程序设计方法，全书共 12 章，分别介绍了 Java 语言的背景、数据类型、运算符、控制流、数组、面向对象程序设计、异常处理、线程、Java Applet、图形用户界面、输入/输出处理和网络通信。

本书可作为高等学校理工科学生教材或教学参考书，也可作为工程技术人员的培训教材或自学参考书。

图书在版编目(CIP)数据

面向对象的 Java 语言程序设计/孟祥武，张玉洁编著. —北京：北京邮电大学出版社，2001.2
ISBN 7-5635-0496-6

Ⅰ.面… Ⅱ.①孟…②张… Ⅲ.JAVA 语言－程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2001）第 07363 号

书　　名：面向对象的 Java 语言程序设计
责任编辑：蒋　亮
出 版 者：北京邮电大学出版社(北京市海淀区西土城路 10 号)
　　邮编：100876　电话：62282185　62283578
　　网址：www.buptpress.com
经　　销：各地新华书店
印　　刷：北京源海印刷厂
开　　本：787 mm×1 092 mm　1/16　印张：9　字数：230 千字
版　　次：2001 年 3 月第 1 版　2001 年 3 月第 1 次印刷
书　　号：ISBN 7-5635-0496-6／TP·47
定　　价：16.00 元

前　言

计算机的主要用途将从单机应用转向网络应用，这是一大趋势，不可逆转。网络就是计算机，Java 是网络上的世界语。它已成为 Internet 网络编程语言事实上的标准。它是一种跨平台的、适合于网络计算环境的面向对象程序设计语言。Java 语言的基础是 C++ 语言，但去除了其中易出错部分。

对于 Java 语言程序，首先由 Java 语言编译器将源程序编译为字节码，字节码是一种中间码，这种中间码由 Java 虚拟机解释和执行。

Java 语言具有简单性、面向对象、分布性、编译和解释性、鲁棒性、安全性、中性的体系结构、可移植性、高性能、多线程和动态性等特点。

Java 语言源于失败的消费电子类技术，然后在一个流产的系统中为计算机提供交互视频信号，最终在 1994 年再次革新，并于 1995 年 5 月推出。

Java 语言自从 1995 年问世以来，很快就流行于全世界，并获得了极大地成功，甚至改变了计算模式，在世界范围内，引起了一股 Java 热。

Java 语言是一种面向对象的程序设计语言，特别适合于 Internet/Intranet 上的应用软件开发。

目前，越来越多的软件开发项目选择 Java 语言作为编程语言，特别是有关计算机网络方面的软件，所以越来越多的人开始学习并使用 Java 语言，因此我们编写了这本教材。

本书共 12 章，第 1 章简要介绍了 Java 语言产生的背景、发展与现状、特点、开发环境。第 2 章讲述了数据类型、标识符、关键字、注释语句等。第 3 章讲述了运算符和优先级。第 4 章讲述了控制流，主要是分支语句和循环语句。第 5 章讲述了数组，包括一维数组和多维数组。第 6 章首先讲述了面向对象程序设计的主要概念，然后具体讲述了 Java 的类、继承、对象、接口、包、修饰符。第 7 章讲述了异常和异常处理过程。第 8 章讲述了线程，包括线程的创建、状态、调度、同步。第 9 章首先讲述了 Java Applet 和 Java Application 的区别，然后讲述了 Java Applet 和多媒体。第 10 章讲述了图形用户界面，主要是 AWT 的事件处理机制和 AWT 的组成。第 11 章讲述了输入/输出处理，包括流、InputStream 和 OutputStream 类、文件 I/O 处理。第 12 章讲述了网络通信，主要是 URL、Socket 通信和数据报通信，并给出了一些实例。

本书的作者从 1996 年起一直从事有关 Java 的科研、开发、教学、培训等工作。

本书是作者在从事有关 Java 语言科研和教学的基础上编写出来的，本书作为一本初学者的入门教材，通俗易懂、简明扼要。

希望本书的编写能对广大初学者有所帮助。因作者水平有限，书中如有不妥之处，敬

请广大读者批评指正。

感谢丁宇新博士、冀振燕博士为我们收集、提供了部分资料。感谢中国科学院软件研究所博士生导师程虎研究员的支持与关心。

最后衷心感谢北京邮电大学白中英教授的关心。

孟祥武　张玉洁

北京邮电大学计算机科学与技术学院

2000 年 12 月

第一章　Java 语言的背景

本章主要讲述 Java 语言的发展与现状、Java 语言的特点及开发环境，并结合一些简单的实例使读者对 Java 语言有个初步的认识。

1.1　Java 语言的发展与现状

计算机的主要用途将从单机应用转向网络应用，这是一大趋势，不可逆转。网络就是计算机，Java 是网络上的世界语。它已成为 Internet 网络编程语言事实上的标准。它是一种跨平台的、适合于网络计算环境的、面向对象的程序设计语言。

最初，Sun 希望进入消费类电子产品市场。对于消费类电子产品，从商业上考虑，要求 CPU 要轻便小巧，软件可靠，价格合理等，最重要的是能够标准化。这就要求编程语言简明、可靠。Sun 专门成立了一个小组开始寻找满足条件的编程语言，小组负责人是 James Gosling。他们认为：目前已存在的程序设计语言，没有能满足要求的。所以要设计一种新的程序设计语言。

考虑到 C++ 语言是当前广泛使用的语言，因此，他们以 C++ 语言为基础，设计出了一种新的程序设计语言 Java。

Java 语言的基础是 C++ 语言，但去除了其中易出错的部分。

Java 是 Sun 公司为家电编程而设计的一种语言，简单紧凑，而且适应不同的 CPU。在 Java 语言开发完成之时，正赶上 Internet 大潮，Java 语言建立的“虚拟运行环境”解决了跨平台问题。Java 语言于 1995 年 5 月推出，获得了极大的成功，甚至改变了计算模式，在世界范围内，引起了一股 Java 热。

最近几年，在 Internet 上出现的特别吸引人的事件，就是 Java 语言和用 Java 语言编写的浏览器 HotJava。

对于 Java 语言程序，首先由 Java 语言编译器将源程序编译为字节码，字节码是一种中间码，这种中间码由 Java 虚拟机解释和执行。

Java 语言的诞生必将对整个计算机产业产生深远的影响，对传统的计算模型提出了新的挑战。

有人预言：Java 语言将是网络上的“世界语”，今后所有用其它语言编写的软件统统都要用 Java 语言来改写。

Java 语言的出现，将会引起一场软件革命，这是因为传统的软件往往都是与具体的实现环境有关，换了一个环境就需要做一番改动，耗时费力；而用 Java 语言编写的程序能运行在不同的机器上，只要所用的机器能提供 Java 语言解释器即可。

Java 语言将对未来软件的开发产生很大的影响，这可从以下几个方面考虑：

(1) 软件的需求分析。可以将用户的需求进行动态地可视化描述，以提供设计者更

加直观的要求。用户的需求是各色各样的,受地区、行业、部门、爱好的影响。用 Java 语言都可以描述清楚。

(2) 软件的开发方法。由于 Java 语言的面向对象的特性,所以完全可以用面向对象的技术与方法来开发,这是符合最新的软件开发规范要求的。

(3) Java 语言的动画效果远比 GUI 技术更加逼真,尤其是利用 WWW 提供的巨大动画资源空间,可以共享全世界动态画面的资源。

(4) 软件最终产品。用 Java 语言开发的软件可以具有可视化、可听化、可操作化和可交互性。对于动画与动作,要它停就停,要它继续就继续,而这是在电影与电视播放过程中难以做到的。

(5) 其它。使用 Java 语言对开发效益和开发价值都有比较明显的影响。

Java 语言程序和支持它的浏览器(如 HotJava, Netscape 的 Navigator3.0 以上版本, Microsoft Internet Explorer3.0 以上版本)的出现正是为了给 Web 浏览提供动画(Animation)、速度(Speed)、交互性(Interactivity)。Java 语言能从你的浏览器里直接播放声音,播放页面里的动画,还能告诉你的浏览器怎样处理新的类型文件。

Java 语言有着广泛的应用前景,大体上可以从以下几个方面来考虑其应用:

(1) 所有面向对象的应用开发,包括面向对象的事件描述、处理、综合等;

(2) 计算过程的可视化、可操作化的软件的开发;

(3) 动态画面的设计,包括图形、图像的调用;

(4) 交互操作的设计(选择交互、定向交互、控制流程等);

(5) Internet 的系统管理功能模块的设计,包括 Web 页面的动态设计、管理和交互操作设计等;

(6) Intranet(企业内部网)上的软件开发(直接面向企业内部用户的软件);

(7) 与各类数据库连接查询的 SQL 语句实现;

(8) 其它应用类型的程序。

1.2 Java 语言的特点

Java 语言具有简单性、面向对象、分布性、编译和解释性、健壮性、安全性、中性的体系结构、可移植性、高性能、多线程和动态性等特点。它可在消费类电子产品、个人计算机、网络计算机、工作站和主机上使用。

Java 语言不仅能创建安全的能在网络上使用的应用程序,而且还改变了人们对未来计算机的看法。

Java 语言是一种很有潜力的程序设计语言。随着计算机网络的普及,Java 语言会变得越来越重要。

Java 语言的特点主要是:

(1) 简单性

语言本身的组成成份较少,结构较小。与已有语言类似,用户容易熟悉掌握。Java 语言把 C,C++ 中一般程序员很少使用的、容易出错的特征加以剔除。Java 语言略去了运算

符重载、多重继承等模糊的概念，并且通过实现自动无用单元收集，大大简化了程序设计者的内存管理工作。

系统简单，它的基本解释器及类的支持只有40 KB左右，加上标准类库和线程的支持也只有215 KB左右。

Java语言的创造者希望通过创建一门简单的语言，以鼓励大家编写简单易懂的程序。各类Java语言程序便于在小型机器上执行，通过网络也容易下载。

(2) 面向对象

Java语言从基础上就被设计成面向对象的，不能在类外面定义单独的数据和函数。所有对象都派生自同一个基类Object，并共享它所有的功能。

也就是说，Java语言最外部的数据类型是对象。Java语言中不允许有独立的变量、常量或者函数。所有的元素都要通过类和对象来访问。

(3) 分布性

Java语言从设计上就支持网络应用，通过它提供的类库可以处理TCP/IP协议，用户可以通过URL地址在网络上很方便地访问其它对象，它是分布式语言。

(4) 编译和解释性

Java语言编译器把Java语言源程序编译成Java字节码。在不同的平台上，用该平台上的Java语言解释器来执行Java字节码。通过编译和解释两步实现了跨平台。字节码本身携带了许多编译信息，使得连接过程更加简单。

(5) 鲁棒性

Java语言是一个强类型语言，它要求以显示的方法声明，它不支持指针。Java语言还具有无用单元回收，异常处理等功能。Java解释程序也执行许多运行时的检查。用Java语言可编写具有鲁棒性的软件。

(6) 安全性

Java语言实现了几种安全机制，保护用户免遭病毒和恶意代码的袭击。

程序员永远不需要显式地释放一块和对象相关联的内存。收集无用单元(由JVM实现)是Java内建的机制。Java语言不允许对指针进行算术运算，具有很高的安全性。

(7) 中性的体系结构

Java语言编译程序生成的字节码与体系结构无关。Java字节码适合各种体系结构。

Java语言程序能在任何系统上运行，只要该系统实现了Java虚拟机(JVM)。这些字节码指令对应于Java虚拟机中的表示，Java解释器得到字节码后，对它进行转换，使之能够在不同的平台上运行。

(8) 可移植性

中性体系结构是确保可移植性的重要组成部分。Java语言规范中没有与机器相关的特性。Java语言的基本数据类型(如浮点数和整数)的范围是确定的，不依赖于具体平台的实现。Java语言环境本身也是可移植的。Java语言编译器是用Java语言编写的，解释器是用C语言编写的。Java的类库中也实现了与不同平台的接口，使这些类库可以移植。

(9) 高性能

和其它解释执行的语言(如BASIC)不同，Java字节码的设计使其很容易直接转换成

对应于特定 CPU 的机器码或汇编码，从而得到较高的性能。

(10) 多线程性

Java 语言对多线程提供支持。多线程机制使应用程序能够并行，同步机制可以保证对共享数据的正确操作。

通过使用多线程，程序设计者可以分别用不同的线程完成特定的行为，带来了更好的交互能力和实时运行能力。

(11) 动态性

Java 语言设计成适应于变化的环境，Java 中的类是根据需要而装入的。在类库中可以自由地加入新的方法和实例变量而不会影响用户程序的执行。并且 Java 语言通过接口来支持多重继承，使之比严格的类继承具有更灵活的方式和扩展性。

总之，Java 是一种程序设计语言，但它不仅仅是一种程序设计语言。

(1) Java 首先是一种面向对象的程序设计语言。

(2) Java 是 Internet 上的通用语言，也是 Internet 上第一个实际应用的语言，它获得了广泛的支持。

(3) Java 是一种编程环境，它包括一个相当规范、较大的类库。这些类库给 Java 提供了强大的开发能力。

(4) Java 是一种操作环境，Java 语言在 Java 虚拟机上运行，通过 Java 虚拟机，Java 可以在不同的机器上运行，而不用重新编译。具体流程如图 1-1 所示。

图 1-1 Java 语言的流程

Java 语言源程序(文件 *.java)被编译生成 Java 语言字节码文件(文件 *.class)，字节码运行在 Java 虚拟机上(Java virtual machine)，Java 虚拟机运行在宿主计算机上。

1.3 Java 语言开发环境

目前已出现了多种 Java 语言开发环境，各有特点。常见的 Java 语言开发环境有以下几种：

1. JDK(Java Developers Kit，Java 开发工具)

1996 年 1 月，Sun 公司推出 JDK1.0，它是命令行工具，包括：

(1) javac：Java 编译器，其作用是生成包含字节码的.class 文件；

(2) java：Java 解释器，Java 虚拟机，也叫 Java 字节码解释器，Java 运行环境；

(3) jdb：Java 调试器，与有些调试器相比，jdb 不复杂，但很有用；

(4) javadoc：API 文档生成器，将 Java 源程序文件转换成 HTML 文档；

(5) appletviewer：Applet 小程序浏览器；

(6) javah：头文件生成器，产生可调用 Java 过程的 C 过程，或建立能被 Java 程序调用的 C 过程的头文件。通过此方法，可用 Java 和 C 共同写一段代码；

(7) javap：Java 反汇编器，也叫 Java 字节码反汇编器。

用 JDK 要使用大量的命令行工具，包括 Java 编译器，解释器等。这意味着，在 Windows 环境下还要打开 DOS，敲入命令和正确的参数，而习惯于在诸如 Windows 窗口模式环境下操作的人，是不愿再用命令行工具的。

2. JavaWorkShop

1996 年 3 月 Sun 公司推出一个跨平台的 Java 语言开发环境——JavaWorkShop。这是一个复杂的 Java 开发平台。它还带有一个非常友好的图形界面。

JavaWorkShop 像 JDK 的一个成熟的外壳(shell)程序，能使人在熟悉、方便的 Windows 环境下编程。使用 JavaWorkShop，只要按一下鼠标，就可以使用 JDK 的工具了。

JavaWorkShop 是基于 Web 的 Java 快速开发环境，可开发和发布 Java applets 和应用程序。它是由一套 Java 语言编写的集成开发工具组成的。这意味着 JavaWorkShop 是多平台的。这给开发者提供了灵活性，可为企业中不同的计算机系统提供相同的 Java 开发环境。

JavaWorkShop 是由 Java 语言编写的，这还意味着新的 Java 类能被自动识别和支持，而不必等待新版本去支持新类。

3. Visual Cafe

Symantec 公司推出 Cafe，Visual Cafe。Visual Cafe 是美国 Symantec 公司推出的可视化 Java 语言集成开发环境。可以在该环境中完成 Java 语言程序的建立、调试、修改和运行等全过程，大大提高了 Java 语言程序的开发效率，缩短了开发周期。

4. Visual J++

微软(Microsoft)公司推出可视化 Java 语言集成开发环境——Visual J++，它继承了微软公司集成开发环境(IDE)的一贯传统，该软件非常简便易用。

开发环境中菜单、工具栏以及窗口等功能齐全，解除了 JDK 命令的记忆之苦。

1.4　Java 语言程序实例

下面我们先介绍两个简单的 Java 语言程序，并对其进行分析。

例 1.1

```
// 一个应用程序(application)
public class HelloWorldApp {
  public static void main(String args[]) {
    System.out.println("HelloWorld!");
  }
}
```

本程序的作用是输出下面一行信息：

```
HelloWorld!
```

程序中，首先用保留字 class 来声明一个新的类，其类名为 HelloWorldApp，它是一个公共类(public)。整个类定义由大括号{ }括起来。

在该类中定义了一个方法 main()，其中 public 表示访问权限，指明所有的类都可以使用这一方法；static 指明该方法是一个类方法，它可以通过类名直接调用；void 则指明方法 main()不返回任何值。

对于一个应用程序来说，方法 main()是必需的，而且必须按照如上的格式来定义。Java 解释器在没有生成任何实例的情况下，以方法 main()作为入口来执行程序。Java 语言程序中可以定义多个类，每个类中可以定义多个方法，但是最多只能有一个公共类，方法 main()也只能有一个，作为程序的入口。方法 main()定义中，括号()中的 String args[]是传递给方法 main()的参数，参数名为 args，它是类 String 的一个实例，参数可以为 0 个或多个，每个参数用"类名.参数名"来指定，多个参数间用逗号分隔。在方法 main()的实现(大括号)中，只有一条语句。例如在上例中：

```
System.out.println("HelloWorld!");
```

该句是用来实现字符串的输出，这条语句与 C 语言中的 printf 语句有相同的功能。另外，// 后的内容为注释。

现在我们可以运行该程序。首先把它放到一个名为 HelloWorldApp.java 的文件中，这里，文件名应和类名相同，因为 Java 解释器要求公共类必须放在与其同名的文件中，然后对它进行编译。具体操作如下：

```
C > javac HelloWorldApp.java
```

编译的结果是生成字节码文件 HelloWorldApp.class。

最后用 Java 解释器来运行该字节码文件：

```
C > java HelloWorldApp
```

结果在屏幕上显示：

```
HelloWorld!
```

我们再来看下面的一个例子：

例 1.2

```
import java.awt.*;
import java.applet.*;
// 一个 Applet(小应用程序)
public class HelloWorldApplet extends Applet {
  public void paint(Graphics g) {
    g.drawString("HelloWorld!",20,20);
  }
}
```

这是一个简单的 Applet(小应用程序)。在该程序中，首先用 import 语句输入 java.awt 和 java.applet 下所有的包，使得该程序可以使用这些包中所定义的类，它类似 C 语言中的 #include 语句。然后声明一个公共类 HelloWorldApplet，用 extends 指明它是 Applet 的子类。在类中，我们重写父类 Applet 的方法 paint()，其中参数 g 为 Graphics 类，它表明当前

作画的上下文。在方法 paint()中，调用 g 的方法 drawString()，在坐标(20,20)处输出字符串"HelloWorld!"，其中坐标是用像素点来表示的。

这个程序中没有实现方法 main()，这是小应用程序(Applet)与应用程序(Application，如例 1.1)的区别之一。为了运行该程序，首先我们也要把它放在文件 HelloWorldApplet.java 中，然后对它进行编译：

```
C > javac HelloWorldApplet.java
```

编译的结果是得到字节码文件 HelloWorldApplet.class。

由于 Applet 中没有方法 main()作为 Java 解释器的入口，我们必须编写 HTML 文件，把该 Applet 嵌入其中，然后用 appletviewer 来运行，或在支持 Java 的浏览器上运行。

它的 HTML 文件如下：

```
< HTML >
< HEAD >
< TITLE > An Applet < /TITLE >
< /HEAD >
< BODY >
< applet code = "HelloWorldApplet.class" width = 200 height = 40 >
< /applet >
< /BODY >
< /HTML >
```

其中用 < applet > 标记来启动 HelloWorldApplet，code 指明字节码所在的文件，width 和 height 指明 applet 所占的大小，把这个 HTML 文件存为 Example.html，然后运行：

```
C > appletviewer Example.html
```

这时屏幕上弹出一个窗口，其中显示 HelloWorld!，显示结果如图 1-2 所示。

从上述例子可以看出，Java 语言程序是由类构成的，对于一个应用程序来说，必须在一个类中定义方法 main()，而对 Applet 来说，它必须作为 Applet 的一个子类。在类的定义中，应包含类变量的声明和类中方法的实现。

Java 语言在基本数据类型、运算符、表达式、控制语句等方面与 C，C++ 基本上是相同的，但它同时也增加了一些新的内容，在以后的各章中会详细介绍。

本章只是使大家对 Java 语言程序有一个初步的了解。

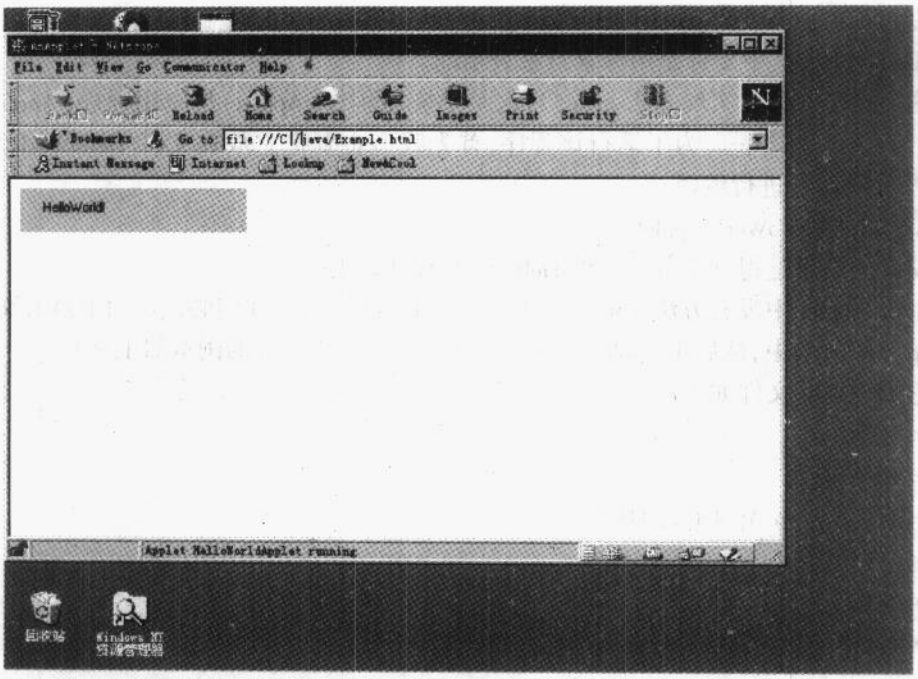

图 1-2　显示 HelloWorld! 的 Java Applet

第二章　数据类型

本章主要介绍 Java 语言标识符、关键字、注释、数据类型等。

2.1 标识符

Java 语言采用的是 Unicode 编码字符集，在这种字符集中，每个字符用 2 个字节，即 16 位来表示。这个字符集包含 65 535 个字符。其中，前面 256 个表示 ASCII 码，使其对 ASCII 码具有兼容性；后面 21 000 个字符用来表示汉字、日文片假名、平假名和朝鲜文等。

但 Unicode 只用在 Java 平台内部，当涉及打印、屏幕显示、键盘输入等外部操作时，仍由具体计算机的操作系统决定表示方法。

Java 语言标识符是以字母、下划线或美元符“$”开始的，包含字母、数字、下划线或美元符“$”的字符序列。

标识符是大小写相关的，且可以任意长。标识符可以包含数字，但不能以数字打头。标识符不能包含空白(Tab，空格，换行符或回车)。关键字不能作标识符。

标识符的命名也很重要，好的标识符能使编程容易、规范，减少错误，便于阅读和维护。

一般来说，标识符的命名应注意以下几点：

(1) 标识符的长度合适，能够反映它所代表的含义即可，即“见名知意”原则。不宜太短，太短很容易混淆，避免 a、ab 等不具任何实际含义的标识符；也不宜太长，增加录入工作量和出错的可能性。

(2) 尽量少使用下划线、美元符和 26 个英文字母以外的字符，以减少录入工作量，提高编程效率。

(3) 因为一些 C 语言程序的库名是用下划线或美元符“$”开头，所以在 Java 语言程序中，标识符最好避免用这些字符开头。如果程序使用了下划线或钱币符作为标识符的开头字母，在向程序中导入 C 语言程序库时可能会造成名字冲突和混乱。

除了这些限制，为了使标识符具有较好的可读性，应当遵守一定的命名规范。尽管这些命名规范不影响编译器的编译，但是遵守它们是一种好的编程习惯。下面列出了不同类型标识符的命名规范。

标识符类型	常规	示例
类	每个单词的首字母都大写	Mammal，SeaMammal
函数	第一个字母小写，其它单词的首字母大写	getAge，setHeight
变量	第一个字母小写，其它单词的首字母大写	age，brainSize
常量	所有字母大写，单词之间使用下划线分开	MAX_HEIGHT，MAX_AGE

标识符是一个唯一标识一个变量，方法和类的名字。标识符用于唯一识别变量，方法

和类。

大写字母和小写字母被认为是两个不同的字符，Java 语言对大小写是区分的。例如，My 和 my 被认为是两个不同的标识符。

变量是指在程序的运行过程中，其值会改变的量。在程序中用作保存中间结果或最终数据，也就是一个用以储存数据的计算机内存的名字。

变量是 Java 语言程序中的基本存储单元，它的声明包括变量名、变量类型和作用域几个部分。

变量名是一个合法的标识符，Java 语言对变量名区分大小写，变量名不能以数字开头，而且不能为关键字。

合法的变量名如：myName、value - 1、dollar $ 等。

非法的变量名如：2mail、class(关键字)等。

变量名应具有一定的含义，以增加程序的可读性。

变量类型可以为任意一种数据类型。

变量的作用域指明可访问该变量的一段代码。变量在使用前要先进行声明，声明一个变量的同时也就指明了变量的作用域。

按作用域来分，变量有：局部变量、类变量、方法参数、异常处理参数。

局部变量在方法或方法的一块代码中声明，它的作用域为它所在的代码块(整个方法或方法中的某块代码)。

类变量在类中声明，而不是在类的某个方法中声明，它的作用域是整个类。

方法参数传递给方法，它的作用域就是这个方法。

异常处理参数传递给异常处理代码，它的作用域就是异常处理部分。

在一个确定的域中，变量名应该是唯一的。通常，一个域用大括号{ }来划定。

Java 语言中没有全局变量，但可以声明类变量。使用局部变量是安全的，因为它们的生命周期有限。例如，在方法内部声明的变量只在方法内部可以访问，所以就避免在程序其它位置错误使用的危险。

2.2 Java 语言关键字

关键字是预定义的标识符，它对于 Java 语言编译器有特殊意义，不能重新定义。也就是说，关键字是 Java 语言中已经被赋予特定意义的一些单词。关键字也叫保留字。在编程序时，不能用关键字作标识符。

Java 语言很多关键字都是从 C,C++ 借过来的。和 C,C++ 一样，关键字也都是小写。

下面是 Java 语言的关键字：

abstract boolean. break byte case catch char class continue default do double else extends false final finally float for if implements import instanceof int interface long native new null package private protected public return short static super switch synchronized this throws transient true try void volatile while

一般来说，Java 语言关键字可以根据其用途分为下面几类：

(1) 数据声明关键字(boolean,float,int)

(2) 循环关键字(continue,while,for)

(3) 条件关键字(if,else,switch)

(4) 异常关键字(try,throws,catch)

(5) 结构关键字(class,extends,implements)

(6) 修饰符和访问关键字(private,public,transient)

(7) 其它关键字(true,null,super)

2.3 注释语句

注释是程序中不可缺少的部分。一个好的程序员,无论使用何种语言编程,给程序写注释都是一个好习惯,Java 语言程序也不例外。注释不仅有助于对程序的修改和维护,而且有助于别人阅读你的程序。此外,注释还是增强代码可重用性的最简单而有效的工具。

Java 语言中可以采用三种注释方式:

1. //用于单行注释。注释从//开始,终止于行尾。

主要用于只注释一行。当然,对于多行注释,也可用此方式,在每行前加"//"。

如: // comment on one line

2. /*…*/用于多行注释。注释从/* 开始,到 */结束,且这种注释不能互相嵌套。

主要用于要插入一段注释,在注释首尾分别加"/*"和"*/"。

如: /* comment on one

or more lines */

3. /**…*/是 Java 语言所特有的 doc 注释。它以/** 开始,到 */结束。这种注释主要是为支持 JDK 工具 Javadoc 而采用的。Javadoc 能识别注释中用标记@标识的一些特殊变量,并把 doc 注释加入它所生成的 HTML 文件。

这类注释通常用在创建 web 页面的 HTML 文件中。

如: /** documenting comment

having many lines */

2.4 数据类型

数据类型指明了变量或表达式的状态和行为。Java 语言不支持 C,C++ 中的指针类型、结构体类型和共用体类型。

1. 整型数据

(1) 整型常量

与 C,C++ 相同,Java 语言的整型常数有三种形式:

① 十进制整数,例如 123, -456,0

② 八进制整数,以 0 开头,例如 0123 表示十进制数 83, -011 表示十进制数 -9。

③ 十六进制整数,以 0x 或 0X 开头,例如 0x123 表示十进制数 291, -0X12 表示十进

制数 - 18。

对于 long 型值,则要在数字后加 L 或 l,如 123L 表示一个长整数。

(2) 整型变量

整型变量的类型有 byte,short,int,long 四种。下面列出各类型所在内存的位数和其表示范围。

数据长度	整数类型	表示范围
8 位	byte	-2^7——2^7-1
16 位	short	-2^{15}——$2^{15}-1$
32 位	int	-2^{31}——$2^{31}-1$
64 位	long	-2^{63}——$2^{63}-1$

int 类型是最常使用的一种整数类型。对于大型计算,常会遇到很大的整数,超出 int 类型所表示的范围,这时要使用 long 类型。

由于不同的机器对于多字节数据的存储方式不同,可能是从低字节向高字节存储,也可能是从高字节向低字节存储,这样,在分析网络协议或文件格式时,为了解决不同机器上的字节存储顺序问题,用 byte 类型来表示数据是合适的。而通常情况下,由于其表示的数据范围很小,容易造成溢出,应避免使用。

short 类型则很少使用,它限制数据的存储为先高字节,后低字节,这样在某些机器中会出错。

(3) 整型变量的声明

例如:

```
byte b;        //声明变量 b 为 byte 型
short s;       //声明变量 s 为 short 型
int i;         //声明变量 i 为 int 型
long l;        //声明变量 l 为 long 型
```

2. 浮点型(实型)数据

(1) 实型常量

与 C,C++ 相同,Java 语言的实型常数有两种表示形式:

① 十进制数形式,由数字和小数点组成,且必须有小数点,如 0.123,.123,123.,123.0

② 科学计数法形式,如:123e3 或 123E3,其中 e 或 E 之前必须有数,且 e 或 E 后面的指数必须为整数。

实型常数在机器中占 64 位,具有 double 型的值。对于 float 型的常量,只要在数字后加 f 或 F,如 12.3 F,它在机器中占 32 位,且表示精度较低。

(2) 实型变量

实型变量的类型有 float 和 double 两种,下面列出这两种类型所占内存的位数和其表示范围。

数据长度	整数类型	表示范围
32	float	3.4e - 038 ~ 3.4e + 038
64	double	1.7e - 308 ~ 1.7e + 308

双精度类型 double 比单精度类型 float 具有更高的精度和更大表示范围，常常使用。

(3) 实型变量声明

例如：

```
float f;      //声明变量 f 为 float 型
double d;     //声明变量 d 为 double 型
```

[注]与 C,C++ 不同，Java 语言中没有无符号型整数，而且明确规定了整型和浮点型数据所占的内存字节数，这样就保证了安全性、鲁棒性和平台无关性。

3. 字符型数据

(1) 字符常量

字符常量是用单引号括起来的一个字符，如'a','A'。此外，与 C,C++ 相同，Java 也提供转义字符，以反斜杠(\)开头，将其后的字符转变为另外的含义。

与 C,C++ 不同，Java 语言中的字符型数据是 16 位无符号型数据，它表示 Unicode 集，而不仅仅是 ASCII 集，例如 \u0061 表示 ISO 拉丁码的'a'。

在 Unicode 字符集中，每个字符用 2 个字节表示，整个字符集包含 65 535 个字符。其中，前面的 256 个表示 ASCII 码，使其与 ASCII 码兼容；后面的字符用来表示汉字、日文和朝鲜文等。

下面列出了 Java 语言中的转义字符。

\'	单引号字符
\\	反斜杠字符
\r	回车
\n	换行
\f	换页
\t	横向跳格
\b	退格
\ddd	1 到 3 位八进制数据所表示的字符(ddd)，八进制特殊字符序列，ASCII 码八进制值的字符，ddd 为三个八进制数(0—7)，如 \071 是 ASCII 码为八进制数 71(十进制数 57)的字符。
\uxxxx	1 到 4 位十六进制数所表示的字符(xxxx)，Unicode 换码序列。Unicode 十六进制值的字符，xxxx 为四个十六进制数(0—9,A—F)，如 \u0041 是 Unicode 码为十六进制数 41(十进制数 65)的字符。

(2) 字符型变量

字符型变量的类型为 char，它在机器中占 16 位，其范围为 0～65 535。字符型变量的声明如：

```
char c = 'a';      //声明变量 c 为 char 型，且赋初值为'a'
```

与 C,C++ 不同，Java 语言中的字符型数据不能用作整数，因为 Java 语言不提供无符号整数类型。但可以把字符型数据当作整数数据来操作。

例如：

```
int three = 3;
```