高等学校计算机教材

Java语言及其网络应用

朱振元 朱 承 刘 聆 编著

人民邮电出版社
POSTS & TELECOM PRESS

高等学校计算机教材

Java 语言及其网络应用

朱振元　朱　承　刘　聆　编著

人民邮电出版社

图书在版编目（CIP）数据

Java 语言及其网络应用/朱振元，朱承，刘聆编著.
—北京：人民邮电出版社，2006.7（2007.7 重印）
ISBN 978-7-115-14794-3

Ⅰ. J… Ⅱ. ①朱…②朱…③刘… Ⅲ. JAVA 语言—程序设计—高等学校—教材
Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2006）第 051920 号

内 容 提 要

本书分为两部分：第 1 章～第 8 章全面、系统地介绍 Java 语言的基本概念、基本语法和编程方法，第 9 章～第 14 章结合开发工具 JBuilder 介绍 Java 语言的网络应用。前后两部分关系紧密，不仅介绍开发的步骤与方法，更注重分析由 JBuilder 自动生成的各类应用程序的结构框架及代码设置的来龙去脉，从而将 Java 基本概念和语法成分等知识有机地融入到 Java 应用程序的开发之中。

本书语言通俗、条理清晰、应用性强。可作为应用型本科计算机专业的教材，也可作为应用程序开发人员及计算机爱好者的参考书。

高等学校计算机教材

Java 语言及其网络应用

◆ 编　　著　朱振元　朱　承　刘　聆
责任编辑　赵桂珍

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京艺辉印刷有限公司印刷
新华书店总店北京发行所经销

◆ 开本：787×1092　1/16
印张：18.75
字数：449 千字　　2006 年 7 月第 1 版
印数：3 001-4 500 册　　2007 年 7 月北京第 2 次印刷

ISBN 978-7-115-14794-3/TP

定价：26.00 元

读者服务热线：(010)67170985　印装质量热线：(010)67129223

前　言

近年来，计算机的软件开发技术有了变革性的飞速发展，一大批面向对象的程序设计语言及其开发工具相继出现，并已日益显示出其强大的生命力。

Java是一种完全面向对象的程序设计语言，它在C++语言的基础上发展起来，继承了C++中大量有效的语法成分，同时也作了不少改进，抛弃了C++中容易引起问题的成分。同时Java语言又是一种网络程序设计语言，它具有平台无关性，使用这种语言编写的程序，可不作修改地在任何一台计算机上正确运行。Java语言的这些特点已使它成为目前开发网络应用软件所使用的主流语言。作为一种通用语言，Java语言可以说近乎是完美的。

JBuilderX是一个典型的面向对象开发工具，它以Java语言为基础，将面向对象的程序设计方法与数据库技术、网络技术以及可视化、事件驱动和代码自动生成等先进技术完美地结合在一起，使用它可以直观地、快速地开发出高质量的Java网络应用程序。

程序设计语言的学习应以应用程序的开发作为归结点，可视化的开发工具为应用程序的开发提供了完善、便捷的开发环境，但对开发工具的理解与掌握则要求以深入理解的语言知识为前提，这两者是互相依存的。

Java应用程序的源代码往往是由系统自动生成的代码与人工设置的代码交织在一起，离开开发工具，完全由人工编写太麻烦，完全由系统自动生成更不可能。所以Java程序员既要深入理解语言基础，又要熟悉开发环境，熟悉系统自动生成代码的规律与结构。

本书将这两者有机地结合起来，既体现了可视化开发工具的强大功能，又强调了语言基础在应用程序开发中的重要作用，且注重于编程思想的分析与说明，力求揭示面向对象程序设计语言及其开发工具间的内在联系。

本书分为前后两部分，前一部分全面、系统地介绍Java语言的基本概念、基本语法和编程方法，后一部分结合开发工具JBuilderX介绍可视化的开发方法。

本书有以下特色。

（1）立足于Java语言的基础（语法、语义及相关的概念）。作为一名优秀的Java程序员必须对语言本身有深入的理解，本书的前一部分以较大的篇幅全面、系统地介绍Java的基本概念、基本语法和编程方法。

（2）注重于程序代码的分析与说明。本书注重分析由JBuilder自动生成的各类应用程序的结构框架及代码设置的来龙去脉，向读者揭示面向对象的程序设计语言及其开发工具间的内在联系，使读者加深对Java编程思想的理解。

（3）归结于Java网络应用程序的开发。在本书中包含各种类型的Java网络应用程序的开发实例，这些综合开发实例在内容上是完整的和实用的。在实例中运用了可视化的开发方法，体现了语言的基本概念与面向对象程序设计的思想，使读者在掌握设计思想的基础上，应用开发工具所提供的强大的功能，快速地开发出实用的Java网络应用程序。

由于作者水平有限、时间仓促，书中难免存在缺点与疏漏，敬请读者及同行们予以批评指正。

编　者

2006年5月

目　录

第 1 章　面向对象语言与 Java 概述

Java 是一种完全面向对象的程序设计语言，它在 C++语言的基础上发展起来，继承了 C++中大量有效的语法成分，但也对 C++做了不少改进，抛弃了 C++中容易引起问题的成分。同时 Java 语言又是一种网络程序设计语言，它所具有的平台无关性可以使用这种语言编写的程序能不做修改地在任何一台计算机上正确运行。Java 语言的这些特点使它已成为目前开发网络应用软件所使用的主流语言。作为一种通用语言，Java 语言可以说几乎是完美的。在学习 Java 语言前，先了解面向对象程序设计的方法及特点是有必要的。本章将介绍面向对象程序设计方法的特点、面向对象语言所涉及的基本概念、Java 语言的特点及 Java 程序设计的概要。

1.1　什么是面向对象程序设计

在介绍面向对象程序设计方法之前，先来考察一下传统的面向过程程序设计方法的一些特点。

假如要编制一个程序求矩形的面积与周长，在程序中先要定义两个变量 width、height，分别表示矩形的宽与高。如果在程序中多处要用到矩形的面积与周长，往往先将求矩形的面积与周长的过程分别编成函数，然后在主程序中调用这些函数。使用 C++语言中的面向过程程序设计方法可编制以下一段程序：

```
int width=6,height=8;
int getarea(int w,int h) {
    return w*h;
}
int getperi(int w,int h) {
    return 2*(w+h);
}
void main(){
   int area,peri;
   area=getarea(width,height);
   peri=getperi(width,height);
   cout<<"area="<<area<<" peri="<<peri<<endl;
}
```

在上述程序中，定义了两个变量 width、height 和两个函数 getarea()、getperi()，分别求矩形的面积与周长，它们都与矩形的操作有关。但在程序中，表示矩形的数据与矩形的操作函数之间并没有建立起必然的联系，程序中的任何地方都可以对 width、height 变量进行访问。由此可见在面向过程程序设计方法中，数据和操作是分离的，而且程序是从开始至结束顺序地执行的。

这种方法着眼于系统要实现的功能，从系统的输入和输出出发，分析系统要做哪些事情，

进而考虑如何做这些事情，自顶向下地对系统的功能进行分解，来建立系统的功能结构和相应的程序模块结构。

面向过程程序设计方法存在明显的不足。首先是数据安全性问题，例如上述程序中的数据width、height，由于在程序中的任何地方都可以对它们进行访问，因此是不安全的，出错时也很难查明原因；其次是可维护性差，如果是更复杂一些的程序，一旦程序中的数据结构需要改变时，常常要涉及整个程序，这样的修改工作量极大并容易产生新的错误。

而面向对象程序设计方法完全避免了面向过程程序设计方法中所存在的问题。

在面向对象程序设计方法中，相关的数据及操作被统一在一个整体—对象之中。例如对于上述程序，可以先将矩形定义成一个类：

```
class Rect {
  private int  w,h;
  public Rect(int w0,int h0) {
    w=w0;h=h0;
  };
  public  int getarea() {
    return w*h;
  }
  public  int getperi() {
    return 2*(w+h);
  }
}
```

并创建一个相应的对象：

```
Rect r1 = new Rect(6,8);
```

然后通过对该对象执行相应的操作来实现程序的功能。例如，求r1的面积，可执行下述代码：

```
r1.getarea();
```

对于本例，在1.4节中还会做具体的说明。在此，读者不必仔细阅读代码，只要大致领略一下面向对象程序设计的基本方法，就会明白上述代码比较简洁，且容易理解。

如果说在面向过程程序设计方法中，程序可表示为

程序=数据结构+算法

即程序的要素是数据结构及算法，它们都是一个个独立的整体，互相之间没有必然的联系，那么在面向对象程序设计方法中，程序的要素是对象，对象将算法及相应数据结构捆绑在一起，程序的定律可改写为

对象=数据结构+算法

程序=对象+对象……

面向对象程序设计方法将应用问题中所涉及的对象，识别为解决问题所需的各种对象、对象的属性及相应的操作，从而建立起对象的类结构。通过对类的实体施与相应的操作以及各对象间的消息传递来实现系统的功能。

面向对象程序设计方法有以下的优点：首先是数据的安全性，在类以外的代码不能直接访问类中的私有数据，只有通过“正规渠道”即通过类对外提供的接口才能对类中数据执行某些操作，从而确保了数据的安全性；其次是可维护性强，当类中数据或操作的实现过程需

要修改时，不会影响外界对该类对象的操作，从而使整个程序保持稳定；还有一个最大的优点就是代码的可重用性，人们可以通过类的继承，逐层扩展类的功能，建立起功能强大的类库，为规模化的软件开发提供了有力的支撑。

因此，与传统的方法相比，用面向对象程序设计方法开发的软件具有更好的可靠性、可维护性、可重用性和可读性。

在问题空间中，我们将客观世界的实体称为对象，不同对象之间的互相作用和互相通信构成了完整的客观世界。如何将问题空间的这一思维模型直接映射到程序空间，也就是说，面向对象程序设计方法靠什么概念来支持这一思维模型？其中最核心的概念是：类、数据封装、继承和多态性。下面就来介绍这些基本概念。

1.2　面向对象程序设计中的基本概念

1.2.1　类和对象

在面向对象程序设计中，对象是指应用问题中所出现的各种实体，它由一组数据和在这组数据上的一组操作（方法）构成。例如在程序中常用的字符串、线性表、栈、队列等或在 Windows 应用程序中常见的窗体、组合框、编辑框、无线按钮等都可以被看作为对象。

类是对一组具有共同特征的对象的抽象。类中定义了与某一种对象相关联的一组数据以及施予这些数据的一组基本操作。类与对象的关系是抽象与具体的关系，类是对多个对象进行抽象的结果，而对象是类中的一个实体。

下面是一个类定义的例子。

```
class Tint {
   private int v;
   public  Tint(int val){v=val;}
   public  void inc(){v++;}
   public  void dec(){v--;}
   public  int get(){return v;}
}
```

在上述类定义中所定义的对象相当于一个计数器，涉及的数据为一个整数 v，相关的操作为设置计数初值、计数增 1、计数减 1 以及读取计数值等。

在面向对象的程序设计中，如果某个对象被定义成属于某一个类，那么该对象即可成为该类中的一个实例。因此在定义中对象与类这两个概念就相当于变量与类型。类、类型是抽象的概念，而对象、变量表示具体事物，例如以下的语句创建了 Tint 类的一个实例：

```
Tint int1 = new Tint(10);
```

又如，在操作界面的设计中，可以在一个窗体中设置多个编辑框，尽管其位置及外观都不相同，但它们都是属于编辑框组件（类）的对象。

1.2.2　数据封装

数据封装（信息隐蔽）是指在面向对象程序设计中，对象的实现过程（包括数据的存储

方式、操作的执行过程）作为私有部分被封装在类结构中，使用者不能看到，也不能直接操作该类中所存储的数据，而只能根据对象提供的外部接口来访问或操作这些数据。

例如，在 1.2.1 小节中所给出的 Tint 类中，变量 v 被封装在类中，类的外界不能直接访问它，因此我们不能使用以下的语句来使对象 int1 中的变量 v 增 1：

```
int1.v=int1.v + 1;
```

而只能通过 Tint 类中所提供的接口来访问它，以下的语句才是合法的：

```
int1.inc();
```

在类中实现数据封装，就像在学校的四周砌了一道高高的围墙，所提供的接口就像学校所设置的传达室，这样外来人员只能通过传达室来访问学校中的人，这对学校中的人来说是比较安全的。

在传统的面向过程程序设计方法中，虽然也提供过程与函数的调用接口，但并没有将对象作为程序的基本构件，仅是将一组相关的数据及操作集中在一起，在数据与操作之间缺乏明确的关系，这就不能达到数据封装的目的。

无论是对于使用者还是实现者，数据封装都是相当有利的。从使用者的角度来看，只要了解类定义的接口部分，即可操作对象实例，而不必去关心其实现细节。这就好比我们使用手表，只需按操作说明使用它，而不必了解手表的内部结构。这样，使用者在开发过程中就可以集中精力去解决应用中所出现的问题，使问题得到简化，设计出的程序也更加简练、直观。从实现者的角度来看，数据封装也有利于编码、测试及修改。因为只有类中的成员函数才能访问它的私有成员，这样做可以使错误局部化，一旦出现错误，要改变数据的存储方式或改变内部的处理过程，也不至于影响其他模块。只要向外界提供的接口不变，其他所有使用该对象的程序都可以不变，从而大大提高了程序的可靠性和稳定性。

数据封装是通过将相应的数据定义成私有成员来实现的。一般可将要封装的数据定义成私有成员，而将向外界提供的对该数据进行访问的方法定义为公有成员。

1.2.3 继承性

在面向对象程序设计方法中，类与类之间存在着继承关系，继承机制是面向对象程序设计方法中最具特色的方面。所谓继承是指从已定义的类生成新类时，新类将自动包括原有类的相应数据和方法，这种生成新类的过程称为派生。

假设有两个类，类 A 与类 B，若类 B 继承了类 A，则类 B 中将自动包括类 A 的相应数据和方法，这时我们称类 A 为类 B 的超类（或称基类或父类），称类 B 为类 A 的子类（或称派生类或扩展类），称类 B 是从类 A 派生出来的。

这种继承关系与客观世界中存在的一般与特殊的关系相类似。例如：在 Windows 的界面设计中，可将窗体分为一般窗体和对话框，而对话框又可分为打开对话框、确认对话框等，它们都有窗体的共同特征，但又有自身的不同特征。我们可以将窗体定义成超类，建立它的子类，如对话框、打开对话框等。在设计窗体类及其子类时，可将各子类中的共同部分集中到超类中去，子类中只保留自己特有的属性和方法，子类的各对象独享该子类的属性和方法，同时还能继承超类中共有的属性和方法，这样做的好处是可以合理地将各个对象共有的属性和方法分配到所有的类中，减少程序代码的重复。

例如，在员工管理系统中，包括教师、工人等管理对象，由于他们都具有人的共性，可以先将人定义成一个超类 Person，类中包括姓名、年龄、性别和籍贯等数据成员。然后在创建教师类时，由于指定了其超类是 Person，所以自动包括了该超类的相应成员，在教师类中只要增加本类中特有的成员，例如教师的职称、专业方向等。在创建工人类时，其超类也是 Person，也自动包括了该超类的相应成员，在工人类中只要增加本类中特有的成员，例如工人的级别、技术专长等。

继承有单继承与多继承两种情形。单继承是指一个子类只直接继承了一个超类，例如前面提到的对话框对窗体的继承，教师对 Person 的继承都是单继承。多继承是指多个超类（接口）派生出一个子类的继承关系，例如生活中常见的玩具车、沙发床等都是多继承的例子，因为玩具车同时继承了玩具与车的特征，而沙发床同时继承了沙发与床的特征。在 Java 中多继承的情形是通过实现多重接口来实现的。

继承可以是一个传递的过程。假设类 B 从类 A 派生出来，而类 C 又是从类 B 派生出来，从而构成了类的层次体系。这样我们又有了直接超类和间接超类的概念。类 A 是类 B 的直接超类，是类 C 的间接超类，类 C 不但继承了直接超类的相应成员，还继承了所有间接超类的相应成员。

除最顶层的类以外每个类都有一个超类，除最底层的类以外每个类都有一些子类，子类既具有从它的超类那里继承来的数据和操作，又可以扩充自己特有的数据和操作。这样，越是后面的类，包括的数据和方法就越丰富。有了类的继承性及其层次结构，不同对象的共同性质只需定义一次，这符合软件重用的目标。它为我们带来的最大好处是能够充分利用前人的成果。大量经过验证的类以层次的结构存放在类库中，用户可以根据需要加以继承，从而避免了程序设计中一切都要从零开始。

1.2.4　多态性

在现实生活中存在多态性。例如同样是一个“打”字，可以是打网球、打乒乓球等，这就是体现了多态性。同样是“启动汽车”，对于不同类型的汽车对象所表现的行为方式也不同，这也体现了多态性。

在面向对象的语言中，多态性是指同一个方法的多种形态，即允许存在同名而行为方式不同的方法，在编译运行时系统将自动进行识别并调用相应的方法。多态性增加了代码使用的灵活性，为程序设计提供了方便。

有两种多态性，即编译时的多态性和运行时的多态性。编译时的多态性是由方法的重载所引起的，方法名相同但其参数的个数或类型不同，编译时系统根据方法调用所匹配的参数确定所调用的方法。运行时的多态性是由类的继承及对象赋值兼容规则所引起的，由于派生类和其基类可能存在同名的方法，而属于基类的引用变量可以引用属于派生类的对象，因此当对属于基类的引用变量执行该同名的方法时，在编译时就无法确定究竟是调用哪个方法，只有在运行时才能确定调用哪个类中的方法。

1.3　Java 语言的特点

作为一种通用语言，Java 语言可以说几乎是完美的，使用 Java 语言可以开发出面向对象

的、平台无关的、健壮安全高性能的程序。下面介绍Java语言的主要特点并与C++语言作了比较。

1.3.1 Java语言的主要特点

1. 面向对象

Java语言是一种完全面向对象的语言。它不像C++语言那样还保持着对C语言的兼容性。整个Java程序完全由类组成，main方法也作为类的特殊成员（public、static、void）放在起始类中，程序从起始类的main方法开始执行。

2. 平台无关

Java语言所具有的平台无关性是指：使用这种语言编写的程序能不作修改地在任何一台计算机上正确运行，Java语言的这一特点使它成为目前网络应用开发中最受欢迎的程序设计语言。

Java语言的这种平台无关性是由其本身特有的运行机制确定的。要运行一个Java的源程序，首先要有Java编译器将其翻译成中间代码（称为字节码，以“.class”为后缀的文件格式存放），然后由Java虚拟机（解释程序）在特定的计算机平台上对字节码进行解释执行。

Java的字节码也就是Java的虚拟机代码，它与具体的计算机处理器代码无关，因为字节码文件并不是直接运行在计算机平台上，而是运行在虚拟机上，任何由Java编译器产生的字节码都可以在装有Java虚拟机的不同的计算机平台上正确运行。

与某种具体的计算机处理器不同，Java虚拟机通常不是由硬件而是由软件实现的，这个软件就是Java解释器。Java虚拟机的硬件实现称为Java芯片。

要运行Java字节码程序，除Java虚拟机外，还需要Java核心API的支撑，Java API是一些class文件的集合，所以也称为Java类库。核心API与Java虚拟机共同构成Java运行环境（Java Runtime Environment，JRE），也被称为是Java平台。

3. 适合网络应用

作为一种网络程序语言，能够使用Java语言及其开发工具方便地开发出包括客户端（Applet）、本地（Application）和服务器端（Servlet、JSP）的多种不同运行机制的网络应用程序。此外，Java还为网络应用程序的开发提供了java.net类库，使开发者能比较容易地实现基于TCP/IP的分布式应用系统。

4. 可读性好

Java语言对C++语言作了许多改进，其中重要的一点是：去掉了C++中的指针类型，对程序中出现的对象、数组及字符串类的变量统一作为引用变量来处理。作为引用变量，其特点是先要创建或确定它所引用的对象，这可以通过创建后赋值或赋初值等多种手段来实现。引用变量的优点是：它既具有指针变量所具有的效率高、功能强等特点，又具有较好的可读性，完全可以像C++语言中的直接定义的对象那样使用。

此外，Java语言中还去掉了C++语言中的结构和联合、宏定义、全局变量、goto语句、类的多重继承等容易引起问题的语法成分，从而大大地提高了程序的可读性。

5. 安全性

Java语言具有很高的健壮性与安全性。在程序的编译和运行时，Java都要对代码中可能出现的问题进行严格地检查。与此同时还提供了字节码校验器、文件访问控制等安全保