KH北京科海
25年·IT技术出版专家

21世纪高职高专规划教材
计算机专业教育系列

Java
面向对象程序设计

王宏宇　贾仰理　主　编
毋玉芝　刘　颖　白玉芹　副主编

中国人民大学出版社
北京科海电子出版社
www.khp.com.cn

21世纪高职高专规划教材·计算机专业教育系列

Java面向对象程序设计

王宏宇　贾仰理　主　编
毋玉芝　刘　颖　白玉芹　副主编

中国人民大学出版社
·北京·
北京科海电子出版社
www.khp.com.cn

图书在版编目(CIP)数据

Java 面向对象程序设计/王宏宇，贾仰理主编.
北京：中国人民大学出版社，2009
21 世纪高职高专规划教材·计算机专业教育系列
ISBN 978-7-300-10274-0

Ⅰ. J…
Ⅱ. ①王…②贾…
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（2009）第 010700 号

21 世纪高职高专规划教材·计算机专业教育系列
Java 面向对象程序设计
王宏宇　贾仰理　主编

出版发行	中国人民大学出版社　北京科海电子出版社		
社　　址	北京中关村大街 31 号	邮政编码	100080
	北京市海淀区上地七街国际创业园 2 号楼 14 层	邮政编码	100085
电　　话	（010）82896442　62630320		
网　　址	http: //www.crup.com.cn		
	http: //www.khp.com.cn（科海图书服务网站）		
经　　销	新华书店		
印　　刷	北京市鑫山源印刷有限公司		
规　　格	185 mm×260 mm　16 开本	版　　次	2009 年 3 月第 1 版
印　　张	17	印　　次	2009 年 3 月第 1 次印刷
字　　数	411 000	定　　价	29.00 元

内容提要

本书通过丰富、实用的精选实例系统地介绍了使用 Java 语言进行面向对象程序设计的方法和技术，注重提高读者运用 Java 语言和面向对象技术解决实际问题的能力。全书共 12 章，内容包括面向对象程序设计概述，Java 语言概述，Java 语言基础，类、对象和接口，包和 Java 基础类，Java 异常处理，Java 图形用户界面设计，Applet 及其应用，I/O 系统，多线程编程，多媒体编程和网络编程等。

本书编排合理，重点突出，语言流畅，示例丰富。内容上注重科学性、实用性、针对性，突出当今社会对人才应用能力的培养要求；针对所阐述的理论列举了比较典型的实例，便于读者学习、掌握；全部代码都在 Java SDK 1.4.2 运行环境下调试通过；每章都配有小结和习题，方便读者复习巩固本章知识。

本书可作为高职高专院校相关专业教材，也可作为计算机培训和全国计算机等级考试辅导的教学用书，还可供程序开发人员和自学者参考。

前言

使用面向对象程序设计的思想和方法进行系统设计和编程，从根本上实现了从现实世界的问题空间到计算机解题空间的直接映射，使所设计的系统能更加准确地模拟现实世界。同时，面向对象程序设计技术有利于提高程序的复用性、易维护性和易扩展性。因此，面向对象程序设计已经取代面向过程程序设计成为当前程序设计的主流方法。

Java 是新一代面向对象和网络的程序设计语言，它将平台无关性、面向对象、多线程、安全可靠、内嵌的网络支持等诸多特征集于一身，为软件开发人员提供了良好的编程环境，特别适用于 Internet/Intranet 上应用软件的开发，成为编写网络应用软件的首选语言。

《Java 面向对象程序设计》是培养计算机类专业学生的专业技能和基本素质的核心骨干课程，是后续课程和专业学习的基础。同时，Java 语言是全国计算机等级考试的主要语言。

本教材主要面向高职高专院校计算机类专业及其他工科类相关专业的学生，根据高职高专学生的培养目标，结合作者多年来在《Java 面向对象程序设计》教学、科研和工程培训的实践经验编写而成。全书共 12 章：第 1 章介绍了面向对象程序设计的发展历程、基本概念、基本特征、面向对象程序设计语言；第 2 章介绍了 Java 语言的产生历史、特点、Java 开发工具与环境、简单的 Java 程序和集成开发环境 JCreator；第 3 章介绍了 Java 语言的基本组成，基本数据类型，运算符、表达式和语句，流程控制，数组；第 4 章介绍了 Java 面向对象程序设计中类和对象的定义、类的继承和多态、Java 接口；第 5 章对包、Java 类库和 API 文档、字符串类、数学类以及其他常用类进行了介绍；第 6 章介绍了异常的概念、Java 的异常处理类、异常处理机制、如何创建和使用用户自己定义的异常类；第 7 章介绍了图形用户界面设计的基本概念、Java 的 AWT 事件处理机制、Swing 包以及 AWT 图形设计；第 8 章介绍了 Java Applet 的概念、特点及其安全机制，Java Applet 类的运行机制，HTML 如何向 Applet 传递参数，Applet 应用；第 9 章介绍了 I/O 流的概念，并依次详细介绍了 Java 字节流类、字符流类和文件类；第 10 章介绍了线程的基本概念，然后介绍了多线程的实现、控制与调度、互斥与同步；第 11、12 章从实用角度出发，分别介绍了 Java 多媒体编程和网络编程的知识。

本教材在内容上注重科学性、实用性、针对性，突出当今社会对人才应用能力的培养要求；力求内容安排合理，保证知识结构的系统性和完整性，同时在选材上兼顾了初学者的接受能力；注重对重点内容和核心内容的讲解，力求循序渐进，在详细介绍的同时辅以图、表和典型的实例；力求在注重基本知识的基础上，突出实用性。本教材每一章最后安排有“本章小结”，对本章内容进行归纳和总结，便于学生提纲挈领，抓住重点掌握本章内容。

本书的作者是从事计算机专业课程教学的一线教师，对 Java 面向对象程序设计课程的教学规律、教学特点等有深刻的认识和系统的研究。本书是作者在吸收并借鉴已有教材长处的基础上，融入多年的教学实践经验和教学研究成果编写而成。

由于时间仓促，加之编者水平有限，不足之处在所难免，恳请广大读者不吝指正。

编　者

2009 年 2 月

目　录

第 1 章　面向对象程序设计概述

本章要点

- 了解程序设计方法与语言的发展
- 掌握面向对象的基本概念
- 掌握面向对象系统的特性
- 了解常见面向对象程序设计语言

本章首先介绍程序设计方法的发展历程，然后详细介绍面向对象的基本概念、基本特征，最后介绍常见的面向对象程序设计语言。

1.1　程序设计方法的发展

1. 早期程序设计方法和语言的发展

程序设计就是针对某一要处理的问题，按照特定的程序设计方法设计出解决该问题的计算机指令序列。进行程序设计要借助某种计算机语言来编写程序，这种计算机语言我们称为程序设计语言。

自第一台计算机诞生以来，程序设计方法与程序设计语言都在不断发展。到目前为止，程序设计方法经历了面向机器（Machine-Oriented）、面向过程（Procedure-Oriented）和面向对象（Object-Oriented）的发展历程。程序设计语言也经历了从低级语言（机器语言和汇编语言）到高级语言的发展历程。面向机器的程序设计方法使用针对特定机器型号的低级语言开发程序，不利于程序的编写和维护，程序的生产效率很低，质量难以保证，可移植性差。因此，面向过程的程序设计方法和相应的高级语言就应运而生。在面向过程程序设计中，问题被看作一系列需要完成的任务，相应的函数用于完成这些任务，这些函数是面向过程的，即函数关注如何根据规定的条件完成指定的任务。早期面向过程的高级程序设计语言有 FORTRAN、ALGOL、BASIC 等。

由于早期计算机有限的运算速度与存储空间都迫使程序员追求高效率，程序的编写过分依赖技巧，不太注重程序的结构。一个典型问题就是程序中的控制随意跳转，可不加限制地使用 goto 语句，这样的程序不利于阅读和维护。随着程序规模与复杂性的不断增长，软件开发人员

迫切需要采用新的程序设计方法。在这一背景下，20 世纪 60 年代后期诞生了另外一种面向过程的程序设计方法——结构化程序设计方法。其主要思想是采用自顶向下、逐步求精的方法，将整个程序结构划分成若干个功能相对独立的模块，模块之间的联系尽可能简单；每个模块用顺序、选择、循环三种基本结构来实现；每个模块只有一个入口和一个出口。结构化程序设计方法的代表语言是 C、Pascal、Ada 等。结构化程序设计有很多优点：各模块可以分别编程，使程序易于阅读、理解、调试和修改；方便新功能模块的扩充；功能独立的模块可以组成子程序库，有利于实现代码复用等。但是，结构化程序设计方法以解决问题的过程作为出发点，其方法是面向过程的。它把程序定义为“数据结构+算法”，程序中数据与处理这些数据的算法（过程）是分离的。这样，对不同的数据结构作相同的处理，或对相同的数据结构作不同的处理，都要使用不同的模块，从而降低了程序的可维护性和可复用性。同时，由于这种分离，导致了数据可能被多个模块使用和修改，难于保证数据的安全性和一致性。因此，对于小型程序和中等复杂程度的程序来说，它是一种较为有效的技术，但对于复杂的大规模软件的开发来说，用这种方法开发的软件仍存在可维护性和可复用性差等问题。

2. 面向对象程序设计方法

20 世纪 80 年代兴起的面向对象程序设计方法在结构化程序设计的基础上提出了一种新的设计思路。面向对象的程序设计方法以类作为构造程序的基本单位，具有封装、数据抽象、继承、多态性等特点。其基本思想是使用对象、类、消息等基本概念来进行程序设计，从现实世界中客观存在的事物（即对象）出发来构造软件系统，并且在系统构造中尽可能运用人类的自然思维方式。目前，面向对象的程序设计已经是目前软件工业的主流，绝大多数的系统程序、应用程序都是采用面向对象的思想来设计开发的。

面向对象程序设计方法需要相应的语言支持，常见的面向对象程序设计语言包括 Simula、Smalltalk、Eiffel、C++、Java 等。

1.2 面向对象程序设计基本概念

“面向对象”是由英文 Object Oriented 翻译而来的，简称为 OO。在面向对象概念中，整个世界是由各种各样的对象组成的，具有相同属性和行为的对象可以划分为一类。因此，类的成员中不仅包含有描述类对象属性的数据，还包含有对这些数据进行处理的程序代码。所谓面向对象的方法学，就是使我们分析、设计和实现一个系统的方法尽可能地接近我们认识一个系统的方法，包括面向对象的分析（Object-Oriented Analysis，OOA）、面向对象的设计（Object-Oriented Design，OOD）、面向对象的程序设计（Object-Oriented Program，OOP）等。

在面向对象程序设计中，对象是构成软件系统的基本单元，并从相同类型的对象中抽象出一种新型的数据类型——类，对象只是类的实例。将对象的属性和行为放在一起作为一个整体的方法称为封装，它将对象的大部分行为的实现隐蔽起来，仅通过一个可控的接口与外界交互。软件系统中，对象与对象之间存在着一定的联系，这种联系通过消息的传递来实现。在面向对象程序设计中，消息表现为一个对象对另一个对象的行为的调用。

在面向对象程序设计中，经常使用一些术语，下面介绍几个常用的术语。

1. 对象

对象是系统中用来描述客观事物的一个实体，它是构成系统的一个基本单位。现实生活中的对象，如狗、桌子、电视、汽车等有两个共同特征：它们都有状态和行为。例如，狗有自己的状态（如名字、颜色、饥饿等）和行为（如摇尾巴等），汽车既具有型号、颜色、载重等特点，又有完成启动、行驶、刹车等功能。面向对象技术中的对象实际上是现实世界对象的抽象，它同样有状态和行为，一个对象由一组属性和对这组属性进行操作的一组方法组成。从更抽象的角度来说，对象是问题域或实现域中某些事物的一个抽象，它反映该事物在系统中需要保存的信息和发挥的作用。显然，对象是一组属性和对这些属性进行操作的一组方法的封装体，具有唯一的名字。对象的属性即一组数据，用来描述对象的静态特征。例如，汽车的颜色、型号、马力、生产厂家等。对象的方法也称为成员方法，或称为服务、操作，它是对象动态特征（行为）的描述。每一个方法确定对象的一种行为或功能。例如，汽车的行驶、转弯、停车等动作可分别用 move( )、rotate( )、stop( )等方法来描述。

2. 类

在现实世界中，经常可以看到相同类型的许多对象。例如，我们的自行车只是现实世界中许多自行车的其中一辆。使用面向对象技术，可以说我们的自行车是自行车对象类的一个实例。通常，自行车有一些状态（当前档位、两个轮子等）以及行为（改变档位、刹车等）。当厂家制造自行车的时候，厂商利用了自行车共有的特性来根据相同的图纸制造许多自行车。如果制造一辆自行车就要产生一个新图纸，那效率就太低了。面向对象程序设计技术也是采用这样一种思想，我们先定义类，再利用类来实例化对象。类是具有相同属性和操作的一组对象的集合，它为属于该类的所有对象提供了统一的抽象描述，其内部包括属性和方法两个主要部分。

在面向对象的编程语言中，类是一个独立的程序单位，描述一个类需要指明三个方面的内容。

（1）类标识：类的一个有别于其他类的名字。这是必不可少的。

（2）属性说明：用来描述相同对象的静态特征。

（3）方法说明：用来描述相同对象的动态特征。

类与对象的关系就如模具和铸件的关系，类的实例化结果就是对象，而对一组对象的抽象就是类。例如：如果把“人”看成是一个抽象的类，我们每一个具体的人，就是“人”类中的一个实例，即一个对象，每个人的姓名、年龄、身高、体重等特征可作为“人”类中的数据，吃饭、走路、工作等行为作为类的方法。

3. 消息

一个对象向另一个对象发出的请求称为消息，它是一个对象要求另一个对象执行某个操作的规格说明，通过消息传递才能完成对象之间的相互请求和协作。对象之间利用消息进行交互。单一的对象通常是没有意义的，应用程序通常包含多个对象。通过这些对象的交互作用，程序可以完成相应的功能以及更为复杂的行为。面向对象程序设计技术中的对象与其他对象进行交互与通讯是利用发送给其他对象来实现的。当对象 A 需要对象 B 来执行一个 B 中的方法，对象 A 就会发送一个消息给对象 B。有时候，接收的对象需要更多的信息来确定该如何做，例如，当你想改变自行车的齿轮，你就必须指出哪个齿轮。这个信息是将信息作为参数来传递的。消

息通常包含三个方面的信息：接受消息的对象、需要执行的方法和执行方法需要的所有参数。

通常把发送消息的对象称为消息的发送者或请求者，而把接收消息的对象称为消息的接受者或目标对象。接受者只有在接收到消息时，才能被激活，之后才能根据消息的要求调用某个方法完成相应的操作。所以，消息传递的实质是方法的调用。

由此可见，类提供了完整地解决特定问题的能力，它描述了数据结构（对象属性）、算法（方法）和外部接口（消息）。对象通过外部接口接受它能识别的消息，按照自己的方式来解释这个消息并调用某个方法来执行对数据的处理，从而完成对特定问题的解决。

1.3 面向对象的基本特征

在面向对象方法中，我们通过对现实世界的事物进行抽象来定义类和对象，通过封装将对象的数据和方法绑定，并实现数据和方法的隐藏。通过继承能体现类与类之间的关系，实现代码的复用。抽象、封装、继承和多态一起构成了面向对象的基本特征。

1. 抽象性

人们对世界的各种事物总是分成不同的类来管理的，而抽象是人类对事物进行分类的最基本的方法和手段。面向对象程序设计中的抽象是对一类对象进行分析和认识，经过概括，抽出一类对象的公共性质，并加以描述的过程。

对一个事物的抽象一般包括两个方面：数据抽象和行为抽象。数据抽象是对对象的属性和状态的描述，使对象之间互相区别的特征量的描述。行为抽象是对数据需要进行的处理的描述，它描述了一类对象的共同行为特征，使一类对象具有共同的功能，因此，又称行为抽象为代码抽象。

例如，要设计绘制圆的程序。通过分析可知，圆是这个问题中的唯一事务。对于具体的圆，有的大些，有的小些，圆的位置也不尽相同，但可用三个数据即圆心的横、纵坐标和圆的半径就可以描述圆的位置和大小，这就是对圆这个事物的数据抽象。由于抽象后没有具体的数据，它不能是一个具体的圆，只能代表一类事务——圆类。要能画出圆，该程序还应有设置圆形位置、半径大小、绘制圆形的功能，这就是对圆这个事物的行为抽象。

由上面的例子看出，类的数据成员的实质就是解决问题所需要的数据，它是数据抽象的结果；而成员方法的实质是完成对类中的这些数据进行加工处理的代码，它是类的行为，用行为抽象来描述。因此，抽象性是面向对象的核心。

2. 封装性

封装是把对象的数据（属性）和行为（方法）绑定在一起的一种机制，该机制保证了数据和方法都不受外部干扰且不被误用。理解封装性的一个方法就是把它想成一个黑匣子，它可以阻止在外部定义的代码随意访问内部代码和数据。对黑匣子内代码和数据的访问通过一个适当定义的接口严格控制。因此，封装性指尽可能隐蔽对象的内部细节，它包含两个含义：把对象的全部属性和全部行为结合在一起，形成一个不可分割的独立单位（即对象）；信息隐蔽，即尽可能隐蔽对象的内部细节，对外形成一个边界，只保留有限的对外接口使之与外部发生联系。

3. 继承性

在面向对象程序设计中，继承表达的是类之间的关系，这种关系使得一类对象可以继承另一类对象的属性（数据）和行为（操作），从而提供了通过现有的类创建新类的方法。特殊类的对象拥有一般类的全部属性与方法，称做特殊类对一般类的继承。例如，轮船、客轮；人、大人。一个类可以是多个一般类的特殊类，它从多个一般类中继承了属性与方法，这称为多继承。例如，客轮是轮船和客运工具的特殊类。在 Java 语言中，通常我们称一般类为父类（superclass，超类），特殊类为子类（subclass）。通过对象、类，我们实现了封装，通过子类我们可以实现继承。例如，公共汽车、出租车、货车等都是汽车，但它们是不同的汽车，除了具有汽车的共性外，它们还具有自己的特点（如不同的操作方法，不同的用途等）。这时我们可以把它们作为汽车的子类来实现，它们继承父类（汽车）的所有属性和行为，同时增加自己的状态和行为。通过父类和子类，我们实现了类的层次化，可以从最一般的类开始，逐步特殊化，定义一系列的子类。同时，通过继承也实现了代码的复用。

4. 多态性

多态是面向对象程序设计的一个重要特征。多态有两种形态，一种是同名的不同方法共存的情况，即多个方法具有相同的名字，但各方法的参数个数或类型等信息必须有不同之处。例如，同样是求面积方法，我们可以根据方法的参数分别对三角形、圆等求面积。另一种形态的多态和继承有关。同一个方法为不同的子类对象调用时可产生完全不同的行动，例如，哺乳动物的子类狗和猫都可以进行哺乳动物具有的“喊叫”行为，狗产生的声音是“汪汪”，而猫产生的声音是“喵喵”。支持多态性的程序中存在这些方法同名的原因是它们的最终功能和目的都相同，但是由于在完成同一功能时，可能遇到不同的具体情况，所以需要定义含不同的具体内容的方法，来代表多种具体实现形式。多态的目的是为了提高程序的抽象度、封闭性和简洁性，统一一个或多个相关类对外的接口。

1.4　面向对象程序设计语言

目前，比较有影响的面向对象程序设计语言有 Simula、Smalltalk、Object-c、Eiffel、C++、Java 等。面向对象程序设计语言的鼻祖是 20 世纪 60 年代开发的 Simula 67，它提供了对象、类、继承等概念，提出了面向对象的术语，奠定了面向对象语言的基础。它的主要用途是进行建模仿真。

20 世纪 70 年代到 80 年代期间的 Smalltalk 语言，是当时最有影响的面向对象语言之一。它包括了 Simula 面向对象的所有特征，而且数据封装比 Simula 更严格。它经历了 Smalltalk-72、Smalltalk-76 和 Smalltalk-80 等几个版本，现在一般使用 Smalltalk-80。

Object-c 是在 1983 年以后开发的、对 C 进行扩充而形成的面向对象语言，但它的语法更像 Smalltalk。它的扩充主要是新引入的构造和运算符，用他们来完成类定义和消息传递。

Eiffel 除了封装和继承以外，还提供了如参数化多态性、对方法实施前置条件和后置断言等面向对象的特征，使得它在商业上具有很大的潜力。从理论上说，Eiffel 是最好的面向对象程序设计语言。

C++在 C 的基础上扩充了对面向对象的支持，既支持面向过程的设计方法，又支持面向对象的设计方法，还有丰富的开发环境，因而得到了广泛的应用。

Java 语言是 Sun 公司于 1995 年推出的、适用于分布网络环境的面向对象语言。它采用了与 C++语法基本一致的形式，将 C++中与面向对象无关的部分去掉，使其语义成为纯面向对象的语言。它使应用程序独立于异构网络上的多种平台，具有能编译或解释执行、连接简单、支持语言级的多线程等特点，也是一种目前广泛使用的面向对象语言。

1.5 本章小结

程序设计方法经历了面向机器（Machine-Oriented）、面向过程（Procedure-Oriented）和面向对象的发展历程。程序设计语言也可分为低级语言（机器语言和汇编语言）到高级语言（面向过程和面向对象语言）。Java 语言是一种完全面向对象的程序设计语言，本章介绍了面向对象核心概念——类与对象，是学习 Java 编程的关键之一。详细介绍对象、类、消息的基本概念，抽象性、封装性、继承性和多态性面向对象的基本特征，常见面向对象程序设计语言。

1.6 习　题

1. 什么是类？什么是对象？它们之间的关系是怎样的？
2. 面向对象的核心特性是什么？
3. 什么是类的继承？它有什么作用？
4. 什么是类的多态性？

第 2 章　Java 语言概述

本章要点

- 了解 Java 语言的产生和发展历史
- 了解 Java 语言的特点
- 掌握 Java 开发环境的配置及如何运行 Java 程序
- 掌握 Java 程序的种类和基本结构
- 了解 Java 的集成开发环境的使用

Java语言是广泛应用的面向对象程序设计语言，它具有简单、面向对象、与平台无关、安全、高效、多线程等特性。Java语言不仅可以用来开发大型的软件系统，而且特别适用于Internet上应用系统的开发，是当前网络编程的首选语言。

本章将对Java语言的产生历史和发展、特点、Java程序的开发和运行环境、Java程序的分类和集成开发环境等方面作简要的介绍，以使读者对Java语言有一个初步的了解。

2.1　Java 语言的产生历史

1991 年，美国Sun公司启动了一个称为Green的项目，该项目旨在使电视机、电话、冰箱、PDA、机顶盒等家庭消费电子产品计算机化，能够相互通信。消费电子产品种类繁多，即使是同一类型的消费电子产品所采用的处理芯片和操作系统也不相同，存在着跨平台的问题。为了能够在消费电子产品上开发应用程序，项目小组成员最初考虑采用当时最流行的编程语言——C++语言作为开发工具，但是研究表明，对于消费电子产品而言C++语言过于复杂和庞大，并不适用，安全性也不令人满意。于是，项目负责人James Gosling设计和开发出一种新的语言，称之为Oak（橡树）。Oak语言的设计目标是用于开发可靠、紧凑、易于移植的分布式嵌入系统，该语言采用了许多C语言的语法，提高了安全性，并且是面向对象的语言。由于当时这些智能化家用电器的市场需求没有预期的高，Sun公司后来放弃了此项计划。

峰回路转，20 世纪 90 年代中期，随着互联网在世界上蓬勃发展，迫切需要一种跨平台的编程语言，用于编写在网络中的各种计算机上都能够正常运行的程序。Sun公司发现原先开发的Oak语言所具有的跨平台、面向对象、安全性高等特点非常符合互联网的需要，于是改进了该语言，Java语言应运而生。

1995 年，Sun公司正式向IT业界推出了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进行运行。IBM、Microsoft、Oracle、Netscape等大公司纷纷与Sun公司签订合同，授权使用Java平台技术。正是因为Java语言符合互联网时代的发展要求，使它获得了巨大的成功。如今的Java语言与当初的Oak语言已不可同日而语，成为最流行的网络编程语言。

随着Java语言的流行，Sun公司和其他公司一起为Java程序员提供了一系列Java开发工具、运行环境、支持Java应用开发的各种API类库等，使Java由单纯的编程语言发展成为Java技术。Java技术包括Java编程语言和支持语言开发、运行的Java平台。目前Sun公司针对不同的市场目标和设备进行定位，把Java平台划分成J2EE、J2SE、J2ME三个平台。J2SE是Java 2 Standard Edition，主要目的是为台式机和工作站提供一个开发和运行的平台。我们在学习Java的过程中，主要是采用J2SE来进行开发的。J2EE是Java 2 Enterprise Edition，主要目的是为企业计算提供一个应用服务器的运行和开发平台。J2ME是Java 2 Micro Edition的简称，主要是面向消费电子产品，为其提供一个Java的运行平台，使得Java程序能够在手机、机顶盒、PDA等产品上运行。

2.2 Java 语言的特点

Java语言之所以流行，是由它的特点决定的。Sun公司在“Java白皮书”中对Java的定义是：Java是一种具有“简单、面向对象、分布式、解释型、健壮、安全、与平台无关、可移植、高性能、多线程和动态执行”等特性的语言。下面简要介绍Java的这些特性。

1. 简单性

Java的语法风格非常近似于C++语言，但摒弃了C++中容易引发程序错误的地方，如指针、多重继承、操作符重载等。Java简化了内存管理和文件管理，而且提供了C++中不具备的自动内存回收机制，从而减轻了编程人员进行内存管理的负担，使其可以将更多的精力投入到程序功能的实现方面，而无须考虑诸如内存释放等枝节问题。Java提供了丰富的类库，程序开发人员可以调用本地甚至远程的类库，这使得应用程序的开发非常简单。

2. 面向对象特性

面向对象可以说是Java最重要的特性。Java语言的设计完全是面向对象的，不允许定义独立于类的变量和方法，任何变量和方法都只能包含于某个类的内部。这就使程序的结构更加清晰，为继承和重用带来便利。

3. 分布式

Java支持客户机/服务器模式。Java提供了一个URL对象，利用该对象我们可以访问网络上的数据，其访问方式与访问本地文件系统几乎完全相同，这样Java程序处理的数据可以分散存放于网络上不同的主机中，以解决海量数据的存储问题；Java的客户机/服务器模式可以把计算

任务从服务器分散到不同的客户端进行处理，从而提高整个系统的执行效率，避免瓶颈制约，增加动态可扩充性。

对于编程人员来说，Java的网络类库是对分布式编程的最好支持。Java网络类库是支持TCP/IP协议的子例程库，目前支持HTTP和FTP等协议。

4. 解释执行

Java程序的最终执行需要经过两个步骤：编译和解释。Java程序经过编译形成字节码，然后在虚拟机上解释执行。任何一台机器，只要配备了虚拟机，就可以运行Java字节码，而不管这种字节码是在何种平台上生成的。因此，Java编程人员在进行软件开发时，不必考虑软件的运行平台。此外，Java通过预先把源程序编译成字节码，提高了传统解释型语言的执行效率。

5. 健壮性

Java的强类型机制、异常处理、自动内存回收等是Java程序健壮性的重要保证。另外，与C/C++不同，Java中不存在指针，Java程序不可能非法访问内存，也不可能意外覆盖内存缓冲区的区域。Java的安全检查机制也使得Java更具健壮性。

6. 安全性

对于网络分布式应用来说，软件的安全性是极为重要的。Java将安全性放在第一位，设置了层层防范。Java摒弃C++语言在安全性和稳定性方面造成很多问题的指针数据类型，同时提供了数组下标越界检查机制。在编译时，Java会进行语法、语义的检查。链接时，Java还要再进行一遍编译级的类型检查，消除间接对象访问。运行时，Java的运行时系统将进行字节码检验，并记录对象的存储情况，将访问限制在安全范围之内。本地的类与远程的类分开运行，阻止远程系统对本地系统的破坏。支持Java的浏览器还允许用户控制Java软件对本地系统的访问。各种综合的措施使Java程序的安全性得到保证。

7. 平台独立性和可移植性

程序的可移植性指的是程序不经修改就可在不同硬件或软件平台上运行的特性。可移植性在一定程度上决定了程序的可应用性。Java的应用程序接口（API）和运行时系统是可移植性的关键。Java为支持它的各种操作系统提供了一致的API。在API界面上，所有Java程序将都不依赖于平台。Java的运行时系统在解释执行程序时，将字节码转化为当前机器的机器码。程序开发人员无须考虑应用时的硬件条件和操作系统结构，用户只需有Java的运行时环境，就可运行编译过的字节码。

8. 高性能

虽然Java是解释执行的，但它仍然具有非常高的性能。Java字节码可以快速地转换成为机器码来执行，而且Java字节码格式的设计就是针对机器码的转换，实际转换时相当简便，自动的寄存器分配与编译器对字节码的一些优化可使之生成高质量的代码。随着Java虚拟机的改进和“即时编译”（just in time）技术的出现使得Java程序的执行速度有了更大的提高。

9. 多线程机制

线程是现代操作系统提出的一个新概念，是比传统的进程更小的一种可并发执行的执行单