Java
面向对象程序设计教程

解绍词　编著

Java
MIANXIANG DUIXIANG CHENGXU SHEJI JIAOCHENG

中国水利水电出版社
www.waterpub.com.cn

Java
面向对象程序设计教程

解绍词　编著

中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

在面向对象程序设计语言中，Java 占据重要地位，故本书以 Java 语言为切入点，介绍了面向对象程序设计的核心思想和理念。本书内容精简，所选内容皆为 Java 语言编程中的核心内容。本书主要内容包括绪论、类与对象、继承与多态、多线程程序设计、输入输出和异常处理、集合与泛型、图形用户界面、网络通信编程、数据库编程等。

本书可作为综合性大学和理工类院校计算机专业及相关专业的本科生使用的教材，也可供兴趣人士自学参考使用。

图书在版编目（CIP）数据

Java面向对象程序设计教程 / 解绍词等编著. -- 北京：中国水利水电出版社，2015.2
ISBN 978-7-5170-2966-3

Ⅰ. ①J… Ⅱ. ①解… 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(2015)第035254号

策划编辑：杨庆川　责任编辑：陈　洁　封面设计：崔　蕾

书　　名	Java 面向对象程序设计教程
作　　者	解绍词　编著
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路 1 号 D 座 100038） 网址：www. waterpub. com. cn E-mail：mchannel@263. net（万水） sales@waterpub. com. cn 电话：（010）68367658（发行部）、82562819（万水）
经　　售	北京科水图书销售中心（零售） 电话：（010）88383994、63202643、68545874 全国各地新华书店和相关出版物销售网点
排　　版	北京鑫海胜蓝数码科技有限公司
印　　刷	三河市天润建兴印务有限公司
规　　格	170mm×240mm　16 开本　15.75 印张　204 千字
版　　次	2015 年 7 月第 1 版　2015 年 7 月第 1 次印刷
印　　数	0001—3000 册
定　　价	48.00 元

前　言

随着计算机技术和网络技术的发展，Java 语言作为面向对象的、跨平台的编程语言，自 1996 年正式发布以来，在短时间内即成为 IT 领域的主流编程语言。面向对象的 Java 语言具备一次编程、任何地方均可运行的能力，使其成为软件服务提供商和系统集成商用以支持多种操作系统和硬件平台的首选解决方案。Java 作为软件开发的一种革命性的技术，其地位已被确定。如今，Java 技术已被列为当今世界信息技术的主流之一。

由于 Java 语言具有简单易学、面向对象、使用范围广等特征，因此，非常适合于普通高等院校程序设计课程，尤其是面向对象程序设计课程。本书采用循序渐进、由浅入深、概念与例程相结合的撰写方式，对内容的安排、例程的选择进行了严格控制，确保难度适中，更贴近于实用。

在学习本书之前，读者应具有基本的计算机操作基础，但不必具有编程基础。掌握一门语言最好的方式就是实践，本书的着眼点是将基础的理论知识讲解和实践应用相结合，使读者在理解面向对象的思想上，快速掌握 Java 编程技术。

作者是有多年丰富教学经验的一线教师，本书是作者在总结多年教学经验，参考国内外 Java 先进程序设计思想及案例的基础上完成的。本书适合 Java 初学者和进阶者阅读。

目前市场上有关 Java 面向对象程序设计的图书很多，但本书有其独到之处，主要体现在以下几个方面：

(1)强调面向对象的分析思路和设计思想。通过生动的实例阐明封装、继承、多态等相关概念，以典型的例子再现封装、继承、多态等概念在实例中的应用。

(2)内容组织。强调知识的系统性、连贯性、实用性。基本概念、编程方法由易到难逐层展开,内容表达环环相扣,读者易学易用。

(3)详略得当。由于 Java 面向对象开发涉及知识面比较广,本书仅对关键部分进行详细介绍,其他部分点到为止,以便读者的注意力能够集中到关键环节。

(4)问题定义清晰,解题思路明确。对于比较复杂的案例,对其的分析、设计过程及出现的问题都有一个全面的介绍,把编程理论和编程实践完美地结合在一起。

本书由 9 个章节组成,内容包括:绪论,类与对象,继承与多态,多线程程序设计,输入输出和异常处理,集合与泛型,图形用户界面,网络通信编程,数据库编程。

本书由解绍词、熊仕勇、杜伟奇共同完成,其中第 6 章、第 7 章、第 9 章由熊老师和杜老师共同完成,其他章节由解老师完成。

由于时间仓促加上作者水平有限,书中实例虽然经过了多次测试,但难免出现疏漏或不妥之处,恳请广大专家读者给予批评指正,不胜感激!

作　者

2014 年 12 月

目　录

前言 …… 1
第 1 章　绪论 …… 1
1.1　面向对象程序设计 …… 1
1.2　Java 概述 …… 14
1.3　Java 语言基础 …… 21
1.4　Java 程序 …… 24
1.5　本章小结 …… 33
第 2 章　类与对象 …… 35
2.1　类的定义 …… 35
2.2　对象的创建 …… 37
2.3　方法 …… 41
2.4　静态成员 …… 44
2.5　包和实用类 …… 50
2.6　封装 …… 57
2.7　本章小结 …… 61
第 3 章　继承与多态 …… 63
3.1　继承与 Java 中的继承 …… 63
3.2　终止继承：final 类和 final 方法 …… 77
3.3　抽象类 …… 79
3.4　多态 …… 84
3.5　本章小结 …… 90
第 4 章　多线程程序设计 …… 92
4.1　进程与线程 …… 92
4.2　Java 线程类和接口 …… 93
4.3　线程调度与控制 …… 99
4.4　线程的同步机制 …… 104

4.5 本章小结 …… 107
第 5 章 输入输出和异常处理 …… 109
5.1 数据流概述 …… 109
5.2 字节流与字符流 …… 113
5.3 文件操作 …… 123
5.4 对象流 …… 128
5.5 异常处理 …… 133
5.6 本章小结 …… 143
第 6 章 集合与泛型 …… 146
6.1 集合 …… 146
6.2 泛型 …… 163
6.3 本章小结 …… 168
第 7 章 图形用户界面 …… 169
7.1 图形用户界面概述 …… 169
7.2 Swing 图形用户界面 …… 172
7.3 界面布局 …… 183
7.4 常用控件及事件响应 …… 188
7.5 本章小结 …… 200
第 8 章 网络通信编程 …… 202
8.1 Java 网络编程概述 …… 202
8.2 URL 类及相关类 …… 207
8.3 Socket 套接字编程 …… 213
8.4 Datagram 数据报编程 …… 222
8.5 本章小结 …… 226
第 9 章 数据库编程 …… 227
9.1 Java 数据库编程概述 …… 227
9.2 JDBC 主要类与接口 …… 234
9.3 JDBC 数据库访问操作 …… 238
9.4 本章小结 …… 243
参考文献 …… 245

第1章 绪 论

1.1 面向对象程序设计

1.1.1 面向对象程序设计思想的诞生

随着软件复杂度的提高，以及 Internet 的迅猛发展，原先面向过程的软件开发方式已经越来越难以满足软件开发的需要。针对复杂程度越来越高的软件需求，面向对象的软件开发模式诞生了。目前作为针对软件危机的最佳对策，面向对象(Object Oriented，OO)技术已经引起人们的普遍关注。许多编程语言都推出了面向对象的新版本，一些软件开发合同甚至也指明了必须使用基于 OO 的技术和语言。下面简要列出了 OO 技术的发展历程。

①诸如“对象”和“对象的属性”这样的概念，可以一直追溯到 20 世纪 50 年代初，首先出现在关于人工智能的早期著作中。然而，1966 年，具有当时更高级抽象机制的 Simula 语言的开发代表了 OO 的实际发展。

②Simula 语言提供了比子程序更高一级的抽象和封装，并且为仿真一个实际问题，引入了数据抽象和类的概念。后来一些科学家吸取了 Simula 类的概念，开发出了 Smalltalk 语言。

③几乎在同时，“面向对象”这一术语被正式确定。在 Smalltalk 中一切都是对象——即某个类的实例。最初的 Smalltalk 世界中，对象与名词联系紧密。

④Smalltalk 语言还影响了 20 世纪 80 年代早期和中期的很

多面向对象语言，如 Objective-C(1986 年)、C++(1986 年)、Self(1987 年)、Eiffel(1987 年)、Flavors(1986 年)。同时，面向对象的应用领域也被进一步拓宽，对象不再仅仅与名词相联系，还牵扯到了事件和过程。

⑤到了 20 世纪 90 年代，随着 Internet 的迅猛发展，Sun 公司于 1995 年推出了纯面向对象的 Java 语言，自此之后，OO 技术在开发中越来越占主导地位。

1.1.2 面向对象的开发方法

目前，面向对象开发方法的研究成熟度越来越高，国际上已有不少面向对象产品出现。面向对象的开发方法有 Booch 方法、Coad 方法和 OMT 方法等。

(1)Booch 方法

Booch 最先描述了面向对象的软件开发方法的基础问题，指出面向对象开发是一种在本质上区别于传统的功能分解的设计方法。面向对象的软件分解与人对客观事务的理解更加接近，而功能分解只通过问题空间的转换获得。

(2)Coad 方法

Coad 方法是 1989 年由 Coad 和 Yourdon 提出的面向对象的开发方法。该方法的主要优点是通过多年来大系统开发的经验与面向对象概念的有机结合，在对象、结构、属性和操作的认定方面，提出了一套系统的原则。该方法完成了从需求角度进一步进行类和类层次结构的认定。尽管 Coad 方法没有引入类和类层次结构的术语，但事实上已经在分类结构、属性、操作、消息关联等概念中体现了类和类层次结构的特征。

(3)OMT 方法

1991 年，由 James Rumbaugh 等 5 人提出了 OMT 方法，其经典著作为《面向对象的建模与设计》。

该方法是一种新兴的面向对象的开发方法，对真实世界的对

象建模可以说是开发工作的基础所在，然后围绕这些对象使用分析模型来进行独立于语言的设计，面向对象的建模和设计促进了对需求的理解，有利于开发更清晰、更容易维护的软件系统。该方法为大多数应用领域的软件开发提供了一种实际的、高效的保证，是努力寻求一种问题求解的实际方法。

(4)UML(Unified Modeling Language)语言

软件工程领域在1995—1997年取得了前所未有的进展，其成果超过软件工程领域过去15年的成就总和，其中最重要的成果之一就是统一建模语言(UML)的出现。UML将是面向对象技术领域内占主导地位的标准建模语言。

UML不仅统一了Booch方法、OMT方法、OOSE方法的表示方法，而且对其作了进一步的发展，最终统一为大众能接受的标准建模语言。UML是一种定义良好、易于表达、功能强大且普遍适用的建模语言。它融入了软件工程领域的新思想、新方法和新技术。它的作用域不限于支持面向对象的分析与设计，从需求分析开始的软件开发全过程也是它支持的一方面。

1.1.3 面向对象程序设计的三大特征

学习程序设计语言的关键在于学习其编程思想。这一点，从Bruce Eckel的《Thinking in Java》可以体会到。那面向对象语言的编程思想体现在哪儿呢？怎样真正地“Think in Java”，而不仅仅是“Programming in Java”？起点就在于其三大特征——封装、继承、多态。

有人认为采用封装、继承、多态语法写出一个Java程序，就完全掌握了面向对象语言？那只能说学到了皮毛，仅仅会用Java语法写程序而已。事实上，封装、继承、多态是一种设计理念，一种程序艺术，与程序语言可以说是毫无关系。如果真正透彻地理解了这三大特征的真谛，即使不用C++、Java、Objective-C等面向对象的语言，用C语言也能写出面向对象的程序。在面对一个

项目时，有经验的开发者可以立刻在脑海里构思出怎样从软件角度设计它，如何将类与类之间关联起来，怎样用封装、继承、多态等机制勾画出程序的基本架构蓝图，以及优化程序架构。至于一些细节问题，如采用什么语言，C++还是 Java，从语法上怎样来实现，这些都不是关键，毕竟语言只不过是程序设计思想的一种表现形式而已。当然，要达到这种集设计与编程为一体的境界，还需要脚踏实地，稳扎稳打，从编程中逐步提高，从实践中成长。

我们将把这三大特征的设计理念作为一条主线，一步一步贯穿到后面的“类与对象”、“继承与多态”章节中，以实例的方式逐步详细地讲解这三大特征在程序设计中是如何体现的，该怎样去运用它们。

现在我们先简单解释一下这三大特征。

1. 封装

封装，即将对象的数据和基于数据的操作封装成一个独立性很强的模块。封装是一种信息隐蔽技术，使得用户只能见到对象的外特性，而其内特性不被看到。封装的目的就是将对象的使用者和所有者分开，使用者不必知道对象的内部细节，只需通过对象所有者提供的通道来访问对象。

通俗地解释就是，把对象不需要对外提供的数据隐藏起来，对外形成一道屏障。数据如何隐藏？藏哪儿？藏在类里，基于数据的操作便也隐藏在类里。这样一来细节都隐藏起来，那如果外界想对封装的数据进行访问怎么办呢？这时候就需要借助于对象定义的公共接口了，这种公共接口就如同与外界沟通的一个桥梁。以下我们举例说明封装的概念。如果你是“学生”这个类的一个对象，具有姓名、性别、学号、英语成绩、C 语言成绩、高数成绩等数据，一般来说，这些都是你的私有信息，别人是无权知道的。如果班长要统计全班成绩并排名又如何操作呢？这时就需要得到你的允许，即把成绩公开。可以定义一个公有的方法，比

如 getGrade(),通过这个方法访问你的某些私有数据(如成绩,班长)就可以得到他需要的成绩,当然,他不需要的信息仍然封装在类里,没有必要公开。

又如另一个对象封装的例子,如图 1-1 所示,公司里的一个部门就是一个对象,三个对象有其封装好的数据及对数据的操作(方法)。以财务部为例,财务数据属于隐蔽信息,如果销售部门想看财务数据,一般是不允许的,但是如果经上级批准,也可以通过公有的接口方法"管理财务"操作来实现。图 1-1 中椭圆形表示数据(属性),矩形表示操作(方法)。

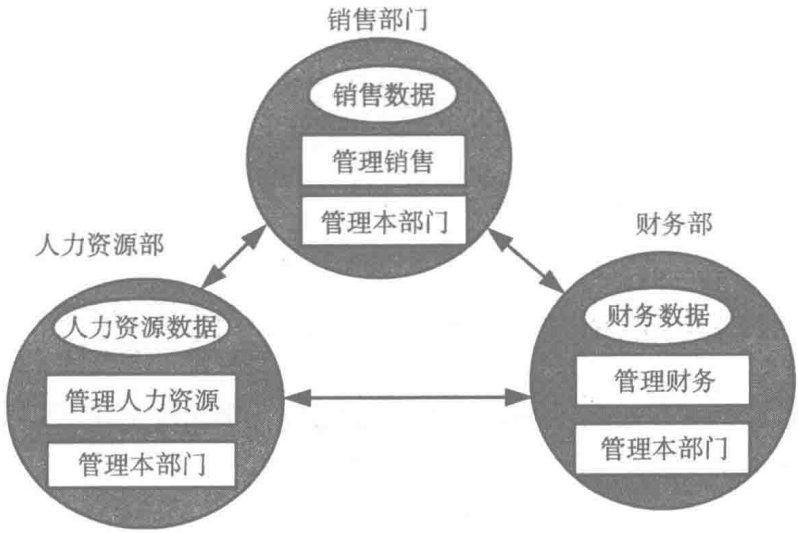

图 1-1 对象的封装示意图

2. 继承

继承,也就是在现有类的基础上创建新类,并在其中添加新的属性和功能。现有类与新类之间是一种一般性与特殊性的关系,现有类具有该类及其新类的共同特征,而新类还具有一些特殊的特征。

类的封装引出了继承的概念。日常生活中的类通常可以再派生出新类:人类可以分为下列子类,如白种人、黄种人、黑种人;交通工具类可分为下列子类,如轿车、卡车、公共汽车等。可以说

就是面向对象最大的优点，而代码重用的主要方式就是继承（后面我们还会提到代码重用的其他方法）。继承是在封装的基础上实现的，前面提到最好把一个封装好的类放在一个.java 文件里，这么做是出于方便类的重用的考虑，可以直接在这个类的基础上派生出子类。该子类可以使用现有类的所有功能，并在无须重新编写现有类的情况下对这些功能进行扩展。试想你要设计一个关于狗的卡通（Cartoon）程序，首先想到的是定义一个 CartoonDog 类，原先已定义的 Dog 类已经具有了普通狗的特征，就没有必要再从头写代码，可以直接继承 Dog 类，在它的基础上添加一些卡通的特性，生成一个新的 CartoonDog 类，既省时又省力。

Java 语言中通过继承创建的新类称为“子类”或“派生类”，被继承的类称为“基类”、“父类”或“超类”。继承的过程就是从一般到特殊的过程。在继承过程中子类继承了父类的特性，包括方法和变量；子类也可修改继承的方法或增加新的方法，使之更适应特殊的要求。继承使代码可以重用，使得数据、方法的大量重复定义在一定程度上得以避免，使系统的可重用性得到了保证，促进了系统的可扩充性，同时也使程序结构清晰，易于维护，提高了编程效率。

下面我们以“愤怒的小鸟”为例。假如你是“愤怒的小鸟”游戏的开发者，当游戏版本不断更新，如 Angry Bird、Space Angry Bird、Crazy Bird……如果游戏的每一个版本都从头开始开发，重复劳动、开发时间加长、效率低下这些都是避免不了的，而采用面向对象的设计就是一种比较好的解决方案。

图 1-2 展现了小鸟类的继承关系，设计 Bird 类为一个基类，具有普通鸟的特征，在 Bird 的基础上派生出了 Angry Bird、Crazy Bird、Kind Bird。这些派生出的子类继承了 Bird 类的所有功能，同时又具备各自的一些新特性。把原有的 Bird 代码拿过来重用，不失为一种提高效率的好方法。

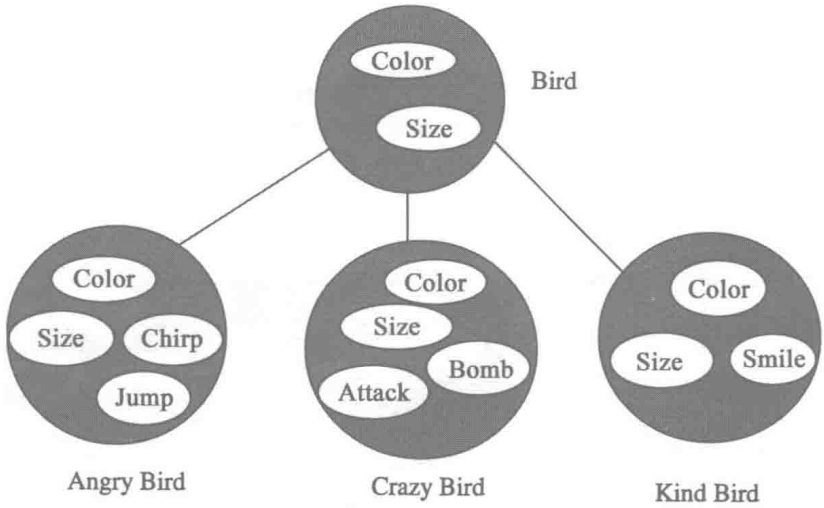

图 1-2 类的继承示意图

3. 多态

多态，即在一个程序中同名的不同方法可以共存，子类的对象可以响应同名的方法，但具体的实现方法不同，完成的功能也不同。

从字面上“多态”这个词理解起来稍有难度。事实上，“多态”起源于我们现实生活。生活中常提到“打”这个动词，“打”可以组词为“打的”、“打扫”、“打架”……这就是“多态”，同一个“打”的行为有不同的反映。多态就是同一个方法可以用来处理多种不同的情形。在程序中，“多态”可以理解为父类与子类都有的一个同名的方法，针对继承关系下不同对象可以采取不同的实现方式。我们还以“愤怒的小鸟”为例。

如图 1-3 所示，功能“Shoot”是三个类同时具有的，以射向隐藏好的小猪，Angry Bird 和 Crazy Bird 的对象，如红色鸟、小蓝鸟、白鸟，同样都具有“Shoot”功能。然而它们发射后执行的方式与效果是截然不同的，小蓝鸟弹出后分离出攻击力更强的三只小鸟，白鸟弹出后会像炸弹一样爆炸，这就是程序中的多态。需要注意的是，多态是以继承为基础的，只有继承关系下的类才具有多态的特性。

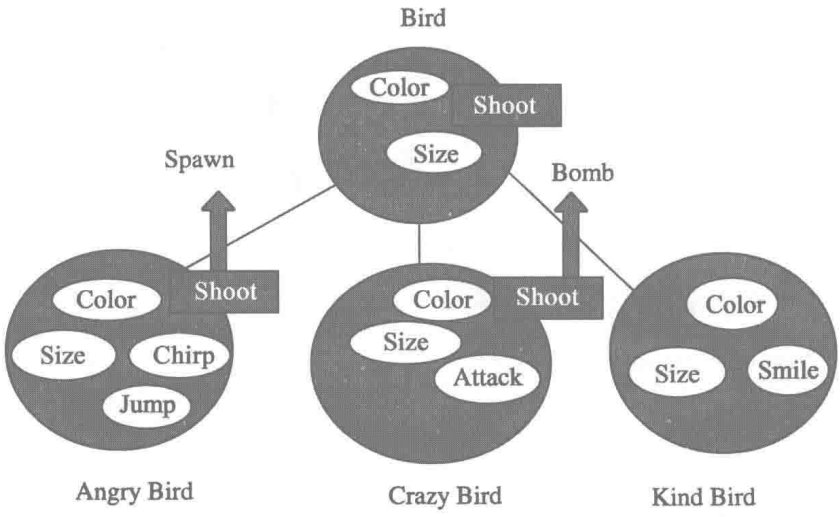

图 1-3　多态示意图

那么，多态到底有什么作用呢？我们知道，一些实现细节可通过封装隐藏起来，使得代码模块化；继承可以扩展已存在的代码模块(类)；它们的目的都是为了代码重用。继承虽然已经扩展了功能，但还不够丰富，多态的引入就是要在继承的基础上进一步实现变异的可能性，增强程序的可扩展性，简单地说，就是一种方法多种实现。

你可能马上会问，如果不用多态，简单地把三个 Shoot 重新命名为“Birdshoot”、“Angrybirdshoot”、“Crazybirdshoot”同样也可以达到目的。没错，在继承关系层次简单的情况下多态的优越性是无法体现出来的。我们看看在继承关系复杂的情形下，假设愤怒的小鸟接二连三派生出不同的版本，如 PC 版、季节版、手机版、PC 版 2、季节版 2 等，如果同一个游戏开发小组的成员每人承担其中一个版本的工作，组长负责最后的版本整合，而每个组员都需要实现一个“Shoot”的功能。试想如果“Shoot”的命名不一样，组长就得牢牢记住每个人的“Shoot”方法名，如“Birdshoot”、“Angrybirdshoot”、“Crazybirdshoot”、“Angrybirdshoot2”、“Crazybirdshoot3”……以在其整合程序里调用。三四个还好说，如果数量更多呢？一旦记错了名字呢？而且组员们派生出的类还在不断增加，最终组长就崩溃了。所以，他最大的希望就是大家在

开发前定好规矩，都统一命名为“Shoot”，然后把“Shoot”的定义与具体实现分离开来。无论组里成员怎样去实现，如何扩展他们的程序，与他都没有关系，他只管关心“Shoot”这个公共接口，只记住“Shoot”这个词，无须考虑实现的细节，执行时就能够自动调用所有子类的“Shoot”功能。同时，组员们只管安心写自己的程序。这时，多态就是最好的解决方案。可见，多态使面向对象语言具有了灵活性、扩展性、代码共享的特征，把继承的优势发挥得淋漓尽致。

1.1.4 面向对象的程序设计

首先，我们必须知道面向对象的软件开发是一项巨大的工程，也是一门专业学科，仅仅依靠学习语法知识就立即进行代码的编写是无法达到开发要求的。在实际软件开发过程中，面向对象的软件开发需要经历一个从需求分析、设计、编程、测试到维护的生命周期。面向对象的思想渗透到软件开发的各个方面，这里简要介绍面向对象软件开发的几个基本流程。

第一阶段：面向对象需求分析（Object Oriented Analysis，OOA），需要系统分析员对用户的需求做出分析和明确的描述。包括从客观存在的事物和它们之间的关系归纳出有关的类及类之间的关系，并将具有相同属性和行为的对象用一个类来表示。

第二阶段：面向对象设计（Object Oriented Design，OOD），在需求分析的基础上，对每一部分进行具体的设计，首先是类的设计，可能包括多个层次，利用继承和组合等机制设计出类的层次关系，然后将程序设计的具体思路和方法一一提出。

第三阶段：面向对象编程（Object Oriented Programming，OOP），选用适当的面向对象编程语言，如C++、C#、Objective-C或Java，选用开发工具，设置开发环境进行代码的编写工作。

第四阶段：面向对象测试（Object Oriented Test，OOT），对程序进行严格的测试，单元测试、集成测试及系统测试等都包括在这个阶段里面。最后还要对程序进行维护管理。

面向对象的需求分析是一个比较繁琐、漫长的过程，主要是与用户交流、沟通、收集信息、整理需求的过程。相关的软件工程课程会有详细的讲解。

本节主要讲述如何根据面向对象需求分析结果，将类与类之间的联系找出来，用 UML 设计出类的层次结构，这部分设计是编程的基础，只有理解了类设计的主导思想，设计出清晰、合理、扩展性强的类结构，才能够完成好的面向对象程序的编写。在实际软件开发中，开发人员在编写代码之前，进行面向对象设计工作是必不可少的步骤。

1. 类的建模

通常使用 UML(Unified Modeling Language)来设计类的结构，即统一建模语言。它不是编程语言，而是为计算机程序建模的一种图形化“语言”，所谓“建模”就是勾画出工程的蓝图，就像盖房子需要首先设计出房子的模型。同样的道理，软件工程“建模”就是在考虑实际的代码细节之前，用 UML 图示将程序结构在很高的层次上表示出来。UML 除了能帮助进行程序的结构设计外，对程序具体工作流程的理解也是非常有帮助的。因为对于大型程序，仅仅看源代码想要弄清楚其各部分之间的联系还是有一定难度的，而 UML 提供了一种直观的方法让我们了解程序概貌，并能描述程序的主要部分、它们是如何一起工作的，以及工作的流程是怎样的。事实上，从文档管理、测试到维护，UML 在软件开发的所有阶段都是有用的。而且，从公司项目管理的角度考虑，UML 也是规范项目管理的一种行之有效的好方法。

IBM 公司的 Rational Rose、Together、MyEclipse 等都是比较常用的建模工具。当然用微软公司的 Visio 也能画出图，而且使用比较简单。UML 最重要的部分是 9 种类图，如表 1-1 所示。

表 1-1　UML

图　　名	说　明
类图(Class Diagram)	表示类之间的关系