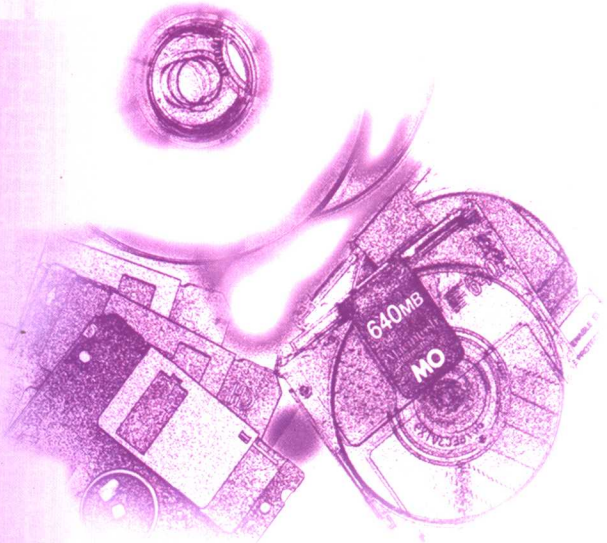
二十一世纪大学计算机应用系列教材 主编：金志农

JAVA与面向对象程序设计导论

北京希望电子出版社 总策划
刘建生 廖列法 吴南萍等 编 著

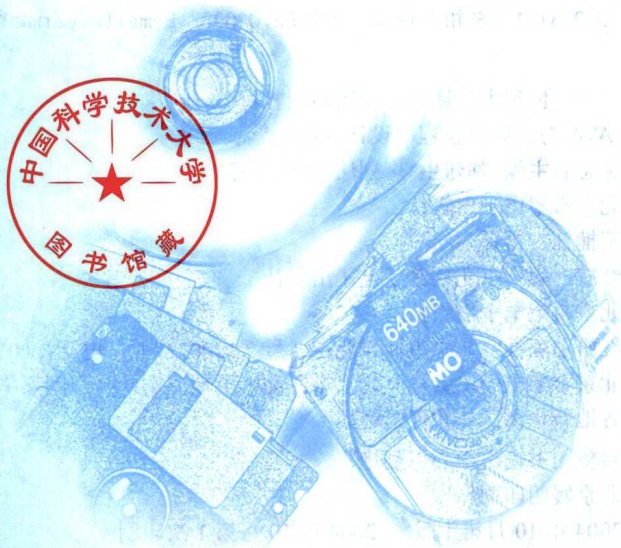


红旗出版社
北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

二十一世纪大学计算机应用系列教材 主编：金志农

JAVA与面向对象程序设计导论

北京希望电子出版社 总策划
刘建生 廖列法 吴南萍等 编 著



红旗出版社
北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

图书在版编目（CIP）数据

JAVA 与面向对象程序设计导论/金志农主编.

—北京：红旗出版社，2004.10

ISBN 7-5051-1001-2

Ⅰ.J… Ⅱ.金… Ⅲ.JAVA 语言—程序设计

Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2004）第 091564 号

内 容 简 介

本书是作者在多年从事面向对象程序设计教学和科研实践的基础上，进行归纳、总结、提高，并参考有关文献编写而成。全书共 10 章。第 1 章概述了面向对象程序设计基本概念；第 2 章讲述了 JAVA 基本语法；第 3 章和第 4 章是运用 JAVA 语言实现面向对象的程序设计；从第 5 章开始讲述 JAVA 的各种主要包和类库的使用；第 6 章运用 JAVA 实现了一些常用的算法和数据结构；第 7 章运用 AWT 包和 SWING 包实现基本的图形用户界面设计；第 8 章讲述了 JAVA 对数据库的访问和操作技术；第 9 章简要介绍了 JSP 技术，并和基本的 JAVA 程序设计技术相结合实现网络编程；并在最后一章实现一个较详细的实例，以便读者学习掌握 JAVA 程序设计方法。

本书面向对象是大学计算机专业或非计算机专业的学生以及其他自学人员。

需要本书或技术支持的读者，请与北京中关村 083 信箱（邮编：100080）发行部联系，电话：010-82702660，62978181（总机）传真：010-82702698 E-mail：yanmc@bhp.com.cn。

系 列 名	二十一世纪大学计算机应用系列教材
书 名	JAVA 与面向对象程序设计导论
编 者	金志农主编 刘建生 廖列法 吴南萍等编著
总 策 划	北京希望电子出版社
责 任 编 辑	王楠楠 雷铎
出 版	红旗出版社 北京希望电子出版社
发 行	北京希望电子出版社
地 址	红旗出版社 北京市沙滩北街 2 号（1000727） 电话：（010）64037138 北京希望电子出版社 北京市海淀区上地三街 9 号金隅嘉华大厦 C 座 610
经 销	各地新华书店 软件连锁店
排 版	希望图书输出中心 娄艳
印 刷	北京媛明印刷厂
版次/印次	2004 年 10 月第 1 版 2004 年 10 月第 1 次印刷
开本/印张	787 毫米×1092 毫米 1/16 18.75 印张
字 数	428 千字
印 数	1～4000 册
书 号	ISBN 7-5051-1001-2
定 价	35.00 元

总　序

有人曾经说过：计算机技术的理念发源于东方，而兴起于西方。在经济全球一体化的今天，国家的竞争实质就是技术的竞争，而技术竞争的根本在于人才的竞争。因此，在中华民族伟大复兴的过程中，造就大批的懂技术的实用性人才就成为我们刻不容缓的一个世纪课题。计算机技术的迅速发展以前所未有的深度和广度深刻地改变着人们的工作、生活、学习和交流方式，对社会的政治、经济、军事、科技和文化等领域也产生着越来越深刻的影响。而掌握计算机技术，让计算机为人们的生活和事业服务，已成为现代人们所必需的最重要的技能之一。因此，培养一大批掌握和运用计算机技术的实用性人才，在全球化知识经济竞争中占据主动地位，不仅是国家经济和社会发展的需要，也是每一个计算机和信息技术工作者责无旁贷的历史责任。

为了适应新世纪计算机技术发展的要求，满足各大专院校师生及社会各阶层人事从事计算机技术和渴望掌握计算机知识的需要，我们组织编写了这套《二十一世纪计算机教育系列教材》，目的是让更多的人迅速地掌握与运用计算机实用性技术，为实现我国经济腾飞和民族复兴的伟大事业尽绵薄之力。

本系列教材包括《计算机文化基础》、《Java 与面向对象程序设计导论》、《数据库技术与应用》、《网络与网络编程》、《Matlab 在理工课程中的应用》等等。随着计算机技术的发展以及读者的要求，我们还将不断地对该系列教材进行补充和完善。本系列教材既可作为大专院校计算机专业或非计算机专业的教材，也可作为社会各类计算机技术培训教材。

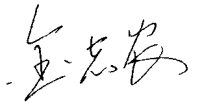
本系列教材的主要特点为：

1．实用性强。本系列教材的编著者均来自各高校和科研院所长期从事计算机课程教学的一线教师，他们有着丰富的教学实践经验和极强的敬业精神，比较真实的了解和掌握各阶层读者对计算机知识的需求。因此，在选材方面，做到了针对性与实用性相结合，尽量回避了一些与软件应用无关的纯理论解说；在内容方面，每个知识点都安排了详尽的实例，以提高读者在实践中运用知识的能力，使读者能真正做到为应用而学习，而不是为理论而学习。

2．适合于不同阶层和层次的读者选用。本系列教材涵盖计算机技术入门知识、程序设计、系统开发以及网络应用等各个方面，可以满足不同领域和不同层次读者的需要。每个读者都可以根据自己的实际情况，找到一本适合自身的教材。

3．写作风格通俗易懂。在写作风格上，编著者力求做到简明、扼要、重点突出。每章的开篇有本章内容简介，结束安排有小结，以便读者能掌握每章的重点与难点。同时，每章附有习题，既方便教学也适合于自学。

由于水平所限，书中难免存在错误和不足之处，敬请广大读者批评指正。



前　言

欢迎阅读本书，这是一本集综合性、实用性为一体的学习 JAVA 语言的教材，本书最显著的特点就是提供了大量实用的编程技巧，本书所有例题和程序都在 JAVA 环境中编译通过并运行成功。

JAVA 是一种面向对象程序设计语言。面向对象程序设计是一种更好的程序设计方法，已经替代了传统的基于过程的程序设计技术。面向对象语言使用抽象、封装、继承和多态，为软件开发提供了极大的灵活性、模块性和可重用性。诞生于 1995 年的 JAVA 以其平台无关性、安全机制、高可靠性和内嵌的网络支持功能使之成为当前编写网络应用程序的首选工具之一。本书以 JAVA 语言为载体，在分析了面向对象基本概念和讲解了 JAVA 基本知识的基础上，运用 JAVA 实现面向对象的设计和实现的方法和技术。

对于初次接触 JAVA 的程序设计人员来说，学习它和学习其他高级设计语言一样，学习程序设计的基本原则是训练准确叙述程序求解步骤的基本技能，并且使用选择语言、循环和方法将求解步骤翻译为程序。新接触面向对象程序设计的读者，要多花一些时间熟悉类和对象等基本概念，掌握面向对象的一些基本原理。在掌握这些基本原理的基础上，就能较快地掌握 JAVA 程序设计技术。本书取材先进、科学、内容丰富、实用，简明易懂、可读性强，例题、习题精心设计，理论与实践紧密结合，是面向计算机科学与技术、自动化、电子信息工程、应用数学、计算数学、系统工程等专业的本、专科生及对面向对象编程技术感兴趣的读者编写的，具有一定的深度和广度。本书是目前国内最新出版的、较为适合教学使用的 JAVA 语言及其应用教材之一。

本书是作者在多年从事面向对象程序设计教学和科研实践的基础上，进行归纳、总结、提高，并参考有关文献编写而成。全书共 10 章。第 1 章概述了面向对象程序设计基本概念，介绍了面向对象与传统面向过程开发方法的区别及多种面向对象开发方法，并简单分析了面向对象方法的开发过程；第 2 章讲述了 JAVA 基本语法，从 JAVA 的编译环境、变量定义、表达式与运算符、控制流与数组，详细地介绍 JAVA 的基本语法知识；第 3 章和第 4 章是运用 JAVA 语言实现面向对象的程序设计，其中第 3 章主要介绍了在 JAVA 中实现类、对象、属性和方法等基本概念，第 4 章则深入讲述了类的组合、重载、接口、包和异常的概念与使用方法；从第 5 章开始讲述 JAVA 的各种主要包可类库的使用，其中第 5 章分析了 Lang 包、Util 包和 I/O 包的主要使用方法；第 6 章运用 JAVA 实现了一些常用的算法和数据结构，将面向对象的程序设计概念运用于数据结构的分析与设计；第 7 章运用 AWT 包和 SWING 包实现基本的图形用户界面设计；第 8 章讲述了 JAVA 对数据库的访问和操作技术；第 9 章简要介绍了 JSP 技术，并和基本的 JAVA 程序设计技术相结合实现网络编程；并在最后

一章实现一个较详细的实例，作为整本书的总结。

JAVA 语言是一门实践性很强的课程，本书的例子都调试通过，希望读者能够自己在阅读的过程中不断实践，上机实现这些例子，只有这样，才能真正理解面向对象程序设计技术，并运用 JAVA 语言实现。

本书的编写工作由江西理工大学多名教师共同完成。他们长期从事 JAVA 教学、面向对象设计与分析工作。这些教学与实践工作为本书的编写打下坚实的基础。刘建生、吴南萍负责全书的框架构思、统稿及第 1、2 章的编著。第 3、4 章由廖列法编著。第 5、6 章由陈优良编著。第 7、8、9、10 章由游安弼编著。我们希望本书的出版能为我国计算机教育事业的发展作出积极的贡献。计算机科学与技术发展日新月异，由于作者水平有限，编写时间较紧，书中如有错误和不足之处，敬请专家和读者提出宝贵意见和建议。

编者著

《二十一世纪大学计算机应用系列教材》

编写委员会

目 录

第 1 章　面向对象程序设计基本概念

[本章内容简介]本章介绍了面向对象与传统面向过程开发方法的区别，讲述了面向对象的基本概念，简要介绍了几种面向对象开发过程，并用一个简单的案例分析了面向对象开发方法的开发过程。

1.1　面向对象与面向过程

所有的计算机程序都由两类基本元素组成：代码和数据。此外，从概念上讲，程序还可以以它的代码或是数据为核心进行组织编写。也就是说，一些程序围绕“正在发生什么”编写，而另一些程序则围绕“谁将被影响”编写。这两种方法决定程序的构建方法。第一种方法被称为面向过程的模型，用它编写的程序都具有线性执行的特点。面向过程的模型可认为是代码作用于数据。第二种方式，面向对象的编程模型，面向对象的编程围绕它的数据（即对象）和为这个数据严格定义的接口来组织程序，实际上是用数据控制对代码的访问，可以简短的定义为：面向对象的编程是通过使用继承性、封装性和多态性来帮助组织复杂程序的编程方法。我们首先考虑几个问题。

普通的计算机系统是建立在冯诺伊曼基本硬件结构基础上，可以看做是一个函数或过程的集合以及单独的一批数据，图 1-1 说明这种静态的结构模型，同时表明在系统运行时，动态的运行过程可以被认为是某个函数 f（1）读数据 A，并在处理数据后写到 DATA1，另外一个函数 f（3）读数据，处理数据后写到 DATA2。

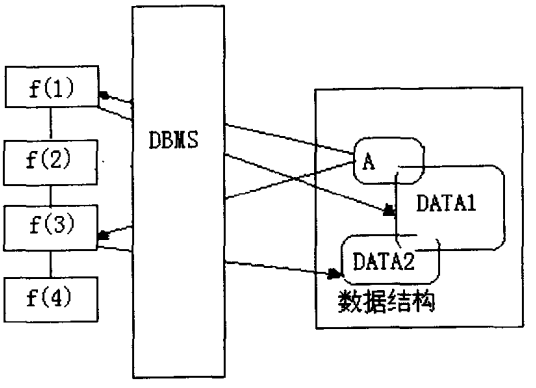


图 1-1　面向过程程序设计

以上的系统处理方法是典型的面向过程程序设计方法，当系统涉及并行性和完整性问题时可以通过利用数据库管理系统解决，但当数据结构的一部分改变时，系统开发人员必须检查每一个函数以确定数据结构的变化是否使函数受到干扰；而且，为了新结构而改动的函数可能对系统的其他部分有副作用。

图 1-2 用完全不同的系统体系结构实现。在此，一个函数和它需要存取的所有数据封装在称为对象（object）的包里，其他对象的函数不能访问这里的数据。可以将对象看成鸡蛋：蛋黄是数据结构，蛋清由访问数据的函数组成，蛋壳代表明显可见的操作标记。蛋壳接口隐藏了函数和数据结构的实现。现在程序维护人员只需检查这个特殊的鸡蛋的蛋清，维护被限制在局部范围内。如果实现变化了，其他对象不会受影响。这里就是封装（encapsulation）；数据和处理过程结合在（combined）一起并隐藏（hidden）的接口后面。

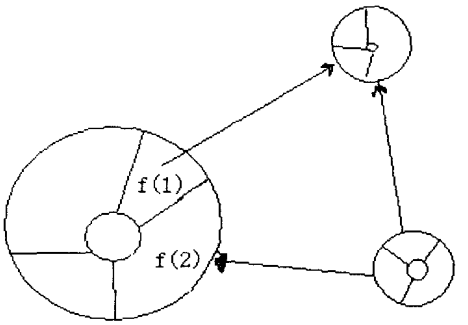


图 1-2 面向对象程序设计

以上的设计思想就是面向对象方法的基本设计思路；面向对象程序设计的基本特征为：

（1）万事万物都是对象：对象是一个奇特的变量类型，除了可以存储数据之外，还可以执行自身所具备的操作能力。

（2）程序是很多的对象，通过消息传递，实现对象之间的通信，从而完成任务。如果要对对象发出请求，必须传送消息到该对象；在程序中，是实现对某个特定对象的函数的调用。

（3）每个对象都可以拥有其他对象。可以封装既有对象的方式来产生新的对象。所以在面向对象程序设计方法中，可以在程序中建立复杂的体系，却将复杂的本质隐藏在对象的单纯性下。

（4）每个对象有所属的类型。每个对象是类（class）的一个实例（instance）。

（5）同一类的所有对象接受的消息相同。

1.1.1 面向对象技术的基本概念

面向对象方法是一种把面向对象的思想应用于软件开发过程中，指导开发活动的系统方法，简称 OO（Object-Oriented）方法。其特点是使用对象模型技术对客观世界进行抽象，分析出事物的本质特征，从而对问题进行求解。Java 语言是 90 年代出现的面向对象的程序设计语言。相对于 C++，Java 去除了其中为了兼容 C 语言而保留的非面向对象的内容。在面向对象技术中，主要的基本概念包括对象、类、继承、封装和多态性等。

1. 对象

对象是结构的基本单位，包括数据（属性）和操作（行为）两个部分。在面向对象技术中，对象是计算机系统的一个基本成分，有一个惟一的名字、一组状态（公共的和私有的数据）和一组操作。在 Java 程序设计中，对象的状态是静止的，用“变量”来描述和表

达；对象的行为是动态的，通过“方法”来实现。

Java 封装的基本单元是类。一个类（class）定义了将被一个对象集共享的结构和行为（数据和代码）。一个给定类的每个对象都包含这个类定义的行为和结构，对象有时被看作是类的实例。所以，类是一种逻辑结构，而对象是真正存在的物理实体。

当创建一个类时，你要指定组成那个类的代码和数据。从总体上讲，这些元素都被称为该类的成员。具体地说，类定义的数据称为成员变量或实例变量。操作数据的代码称为成员方法。一个类的行为和接口是通过方法来定义的，类这些方法对它的实例数据进行操作。

比如，一辆汽车在面向对象程序设计中就是一个对象，具有一定属性和当前状态；汽车还有一些方法，而这些行为方法改变特定汽车的状态和属性。表 1-1 描述了汽车的物理属性、状态和行为操作。

表 1-1　汽车类表

类别	表示	说明
汽车的物理属性	Engine（发动机）	描述发动机的类型、号码等信息
	Wheels（车轮）	4 个车轮，车轮还有自己的属性
	Doors（车门）	四扇车门
	Topspeed	描述车的最高速度
	…….	其他物理属性
汽车的状态属性	Runing	描述汽车的运行状态，是否运行中
	CurrentSpeed	描述汽车的当前速度
	Direction	描述当前汽车的方向
	……	其他状态
汽车的操作	Turnon	启动汽车
	Turnoff	关闭汽车
	Speedup	汽车加速
	Slowdown	汽车减速
	……	其他操作

以上汽车类和对象在统一建模语言（Unified Modeling Language）中的表示为如图 1-3 所示。

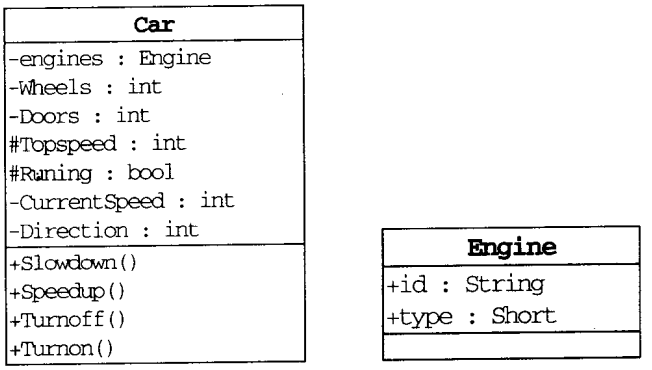


图 1-3　汽车类 UML 表示

其中发动机（engines）本身也是一个类，发动机与汽车的关系在下面的概念中说明。

2. 消息

对象通常在一个包含许多其他对象的系统中作为组件出现。通过这些对象之间的交互，程序员可以实现复杂的行为。为了命令类或对象执行一项任务，要为其发送一个消息。例如，发送 new 消息给 Car（汽车）类以创建一个实例（一个 Car 对象），然后可以为 Car 对象发送启动消息。在 Java 中的实现分别为：

```
Car mycar = new Car();//创建一个 Car 对象
mycar.turnon();//启动 mycar 对象
```

处理消息的类或对象必须相应编程，不能只是将消息随便发送给一个类或对象，而只能将消息发送给能够理解这个消息的类或对象。处理所收到消息的类或对象必须有对应的方法（method），也就是类或对象为完成一项任务而执行的指令序列。

一条消息由接受消息的对象、消息的操作（方法名）和消息带的变量三部分组成。

3. 封装

封装（Encapsulation）是将代码及其处理的数据绑定在一起的一种编程机制，该机制保证了程序和数据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。如果想与现实生活中的某个事物作对比，可考虑汽车上的自动传送。自动传送中包含了有关引擎的数百比特的信息，例如你正在以什么样的速度前进，你行驶路面的坡度如何，以及目前的档位。作为用户，你影响这个复杂封装的方法仅有一个：移动档位传动杆。例如，你不能通过使用拐弯信号或挡风玻璃擦拭器影响传动。所以档位传动杆是把你和传动连接起来的惟一接口。此外，传动对象内的任何操作都不会影响到外部对象，例如，档位传动装置不会打开车前灯！因为自动传动被封装起来了，所以任何一家汽车制造商都可以选择一种适合自己的方式来实现它。然而，从司机的观点来看，它们的用途都是一样的。与此相同的观点能被用于编程。封装代码的好处是每个人都知道怎么访问它，但却不必考虑它的内部实现细节，也不必害怕使用不当会带来负面影响。

既然类的目的是封装复杂性，在类的内部就应该有隐藏实现复杂性机制。类中的每个方法或变量都可以被标记为私有（private）或公共（public）。类的公共接口代表类的外部用户需要知道或可以知道的每件事情；私有方法和数据仅能被一个类的成员代码访问，其他任何不是类的成员的代码都不能访问私有的方法或变量。

在 UML 图中，公共的属性和方法的表示为：+；保护的属性和方法表示为：#；私有的属性和方法表示为：-；本章前面关于汽车的 UML 图中有这三种表示方法。

4. 继承

继承（Inheritance）是一个对象获得另一个对象的属性的过程。继承很重要，因为它支持了按层分类的概念。如前面提到的，大多数知识都可以按层级（即从上到下）分类管理。例如，猎犬是狗类的一部分，狗又是哺乳动物类的一部分，哺乳动物类又是动物类的一部分。如果不使用层级的概念，我们就不得不分别定义每个动物的所有属性。使用了继承，

一个对象就只需定义使它在所属类中独一无二的属性即可，因为它可以从它的父类那儿继承所有的通用属性。所以，可以这样说，正是继承机制使一个对象成为一个更具通用类的一个特定实例成为可能。下面让我们更具体地讨论这个过程。

世界是由对象组成的，而对象又是按动物、哺乳动物和狗这样的层级结构相互联系的。如果你想以一个抽象的方式描述动物，那么你可以通过大小、智力及骨胳系统的类型等属性进行描述。动物也具有确定的行为，它们也需要进食、呼吸，并且睡觉。这种对属性和行为的描述就是对动物类的定义。

描述一个更具体的动物类，比如哺乳动物，它们会有更具体的属性，比如牙齿类型、乳腺类型等。我们说哺乳类动物是动物的子类（subclass），而动物是哺乳动物的超类（superclass）。

由于哺乳动物类是需要更加精确定义的动物，所以它可以从动物类继承（inherit）所有的属性。一个深度继承的子类继承了类层级中它的每个祖先的所有属性。

继承性与封装性相互作用。如果一个给定的类封装了一些属性，那么它的任何子类将具有同样的属性，而且还添加了子类自己特有的属性。这是面向对象的程序在复杂性上线性而非几何性增长的一个关键概念。新的子类继承它的所有祖先的所有属性。它不与系统中其余的多数代码产生无法预料的相互作用。

上面关于动物类的关系用 UML 图表示为 1-4 所示。

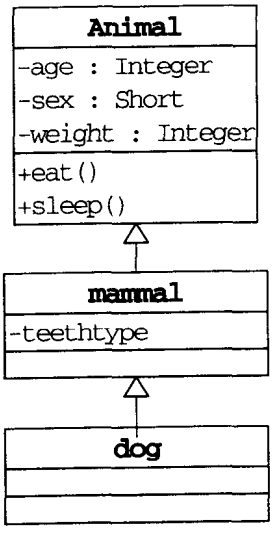


图 1-4　动物类层次图

在继承的类之间有一定的差异，主要有两种差异，第一种只要直接在派生类（子类）中增加新函数就可以。而这些新函数并不是父类接口的一部分。这意味着父类的函数不能满足需要，因此需要加入更多的函数。比如对于图形来说，图形包括多种形状：圆、正方形、三角形等；所有的图形都具备的函数包括：画图、擦除、移动、设置颜色、取得颜色等；但是一种特定的图形具备自己的函数功能；四边形可以有：设置长、设置宽等功能。这几个类的关系如图 1-5 所示。

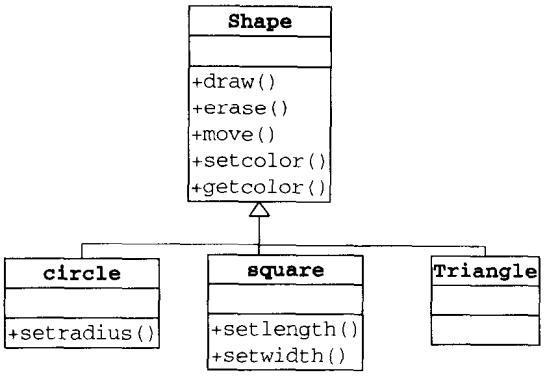


图 1-5　图形类层次图

第二种情况是子类改变父类的函数行为，这种行为称为“覆盖（overriding）”，在此表示：子类和父类有相同的接口函数，但是子类中的函数体与父类不同。这时候，在子类中也要显式的表示出来。如图1-6表示类图形类层次之间的关系。

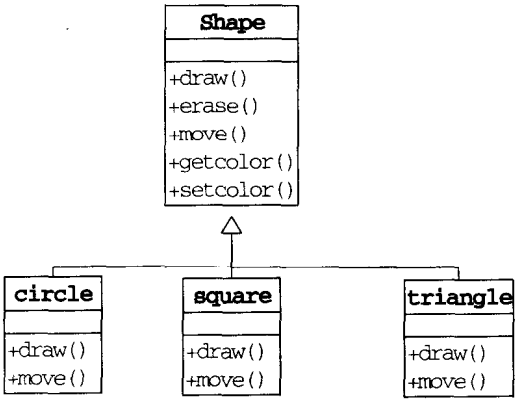


图 1-6　图形类方法“覆盖”UML 图

5. 组合

在面向对象程序设计中，类的复用性是很重要的目标。想要重复使用某个类，最简单的方式是直接使用其所产生的对象，也可以把一个类置于另一个类内；新的类可以由任意数目、任意类型的其他对象组成，这种概念为“组合（composition）”或“聚合（aggregation）”。组合通常被称为“has-a（拥有）”的关系。前面例子中汽车和发动机的关系就是组合关系，汽车拥有发动机，发动机是汽车的组成成分。它们关系的 UML 图表示为 1-7 所示。

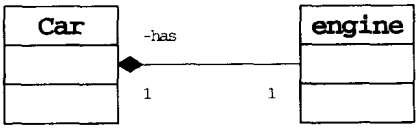


图 1-7　UML 组合图

其中，UML 图中以实心菱形指向汽车，代表组合关系。

另外一种类之间的关系为“聚合”，比如飞机场上有飞机，但是飞机不是飞机场的一部分，它们关系的 UML 图表示为 1-8 所示。

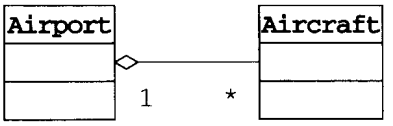


图 1-8　UML 聚合图

6. 多态性

多态性（Polymorphism）是允许一个接口被多个同类动作使用的特性，具体使用哪个动作与应用场合有关，下面以一个后进先出型堆栈为例进行说明。假设有一个程序，需要 3 种不同类型的堆栈。一个堆栈用于整数值，一个用于浮点数值，一个用于字符。尽管堆栈中存储的数据类型不同，但实现每个栈的算法是一样的。如果用一种非面向对象的语言，你就要创建 3 个不同的堆栈程序，每个程序一个名字。但是，如果使用 Java，由于它具有多态性，你就可以创建一个通用的堆栈程序集，它们共享相同的名称。

多态性的概念经常被说成是“一个接口，多种方法”。这意味着可以为一组相关的动作设计一个通用的接口。多态性允许同一个接口被必于同一类的多个动作使用，这样就降低了程序的复杂性。选择应用于每一种情形的特定的动作（即方法）是编译器的任务，程序员无需手工进行选择。只需记住并且使用通用接口即可。

例如：一个函数要求某个图形绘制自己，编译器在编译期无法精确知道应该执行哪一段程序代码；消息被发送时，程序设计者并不想知道哪一段程序代码会被执行；圆形、正方形、三角形的绘图函数没有什么区别；在面向对象程序设计中，面向对象程序设计语言采用“动态绑定”技术实现多态功能。下面的 UML 图中描述了 controll 类和 Shape 类的调用关系，如图 1-9 所示。

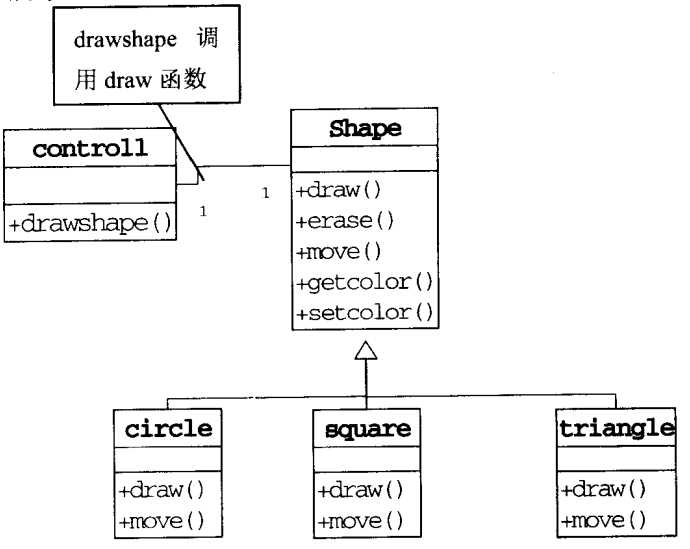


图 1-9　UML 类调用示意图