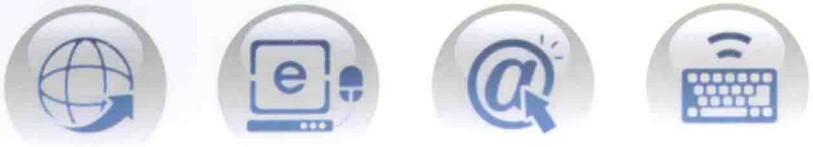
◎计算机

“卓越工程师计划”应用型教材



面向对象程序设计（C#.NET）

Object Oriented Programming (C#.NET)

◎ 王文琴　费贤举　李亦飞　等编著

中国工信出版集团

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

计算机“卓越工程师计划”应用型教材

面向对象程序设计（C#.NET）

王文琴　费贤举　李亦飞　唐学忠　编著

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内 容 简 介

C#语言是微软公司专门为使用.NET 平台而创建的，是一种现代的面向对象的程序开发语言，它使得程序员能够在新的微软.NET 平台上快速开发种类丰富的应用程序。本书以读者不具备面向对象概念为前提，由易到难地全面讲解了 C#相关知识。全书共分为 9 章，主要包括软件开发方法与面向对象概述、.NET 程序设计基础、面向对象程序设计初级篇、面向对象程序设计高级篇、界面设计、文件操作、多线程、图形和数据库程序设计。

本书由多年从事一线程序设计与开发的软件项目开发人员和教学经验丰富的教师根据教学实践和开发心得进行组织编写，充分考虑学生在学习中可能遇到的各种问题和学习效率的提高，以初学者的角度讲述了学习和设计中遇到的所有重点和难点。本书适合作为应用型高等院校计算机及相关专业教材和教学参考书，也可供 C#编程入门者学习使用。

图书在版编目（CIP）数据

面向对象程序设计：C#.NET / 王文琴等编著. —北京：电子工业出版社，2015.7
计算机“卓越工程师计划”应用型教材
ISBN 978-7-121-25685-1

Ⅰ. ①面… Ⅱ. ① 王… Ⅲ. ①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 047741 号

责任编辑：刘海艳
印　　刷：三河市兴达印务有限公司
装　　订：三河市兴达印务有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编　100036
开　　本：787×1 092　1/16　印张：17.25　字数：498.4 千字
版　　次：2015 年 7 月第 1 版
印　　次：2015 年 7 月第 1 次印刷
印　　数：2 500 册　　定价：42.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

计算机“卓越工程师计划”应用型教材
编委会

丛书序言

党的十八大提出要“努力办好人民满意的教育”，要“推动高等教育内涵式发展”，“全面实施素质教育，深化教育领域综合改革，着力提高教育质量，培养学生社会责任感、创新精神、实践能力。”这对高等教育提出了新的要求，明确了人才培养的目标和标准。

十八大明确指出“坚持走中国特色新型工业化、信息化、城镇化、农业现代化道路，推动信息化和工业化深度融合、工业化和城镇化良性互动、城镇化和农业现代化相互协调，促进工业化、信息化、城镇化、农业现代化同步发展。”“推动信息化和工业化深度融合”对高等工程教育改革发展提出了迫切要求。

遵照《国家中长期教育改革和发展规划纲要（2010—2020 年）》和《国家中长期人才发展规划纲要（2010—2020 年）》，为贯彻落实教育部“卓越工程师教育培养计划”，促进我国由工程教育大国迈向工程教育强国，培养造就一批创新能力强、适应经济社会发展需要的高质量计算机工程技术人才，电子工业出版社决定组织相关实施和计划实施卓越计划以及江浙两省实施软件服务外包人才培养试点的地方高校的相关教师，在以往实践校企合作人才培养的基础上编写一套适合地方高校的计算机“卓越工程师计划”人才培养系列教材。

我们将秉承“行业指导、校企合作、分类实施、形式多样”的“卓越工程师教育培养计划”四原则，坚持“学科规范、本科平台、行业应用”，以“具备较为扎实的专业基础知识、拥有良好的职业道德素质、具有创新的计算机应用能力”为目标，探索“校企一体化”产学研结合人才培养模式改革，强化“岗位目标、职业培养”，努力实现计算机工程型技术人才（应用型）培养目标：

（1）尝试以“知识保障、能力渐进、素质为本，重视技术应用能力培养为主线”，坚持以“素质教育，能力培养”为导向，体现本科平台、能力定位、应用背景构建课程体系。

（2）尝试“以学生工程意识、创新精神和工程实践能力培养”为核心，坚持以“培养学生的工程化开发能力和职业素质”为原则，校企合作构建实践教学体系。

本系列教材基于“以德为先、能力为重、全面发展”的人才培养观念，在内容选择、知识点覆盖、课程体系安排、实践环节构建、企业强化训练上按照能力培养和满足职业需求为本进行了有益的、初步的探索。

然而，由于社会对计算机人才的需求广泛而多样，各领域的人才规格和标准既有共性又有特殊性，同时各相关高校在计算机相关专业设置以及人才培养的探索上各有特点，我们编写的本套系列教材目前只能部分满足计算机相关专业人才培养的需要。我们力争建立一个体系，以模块构建的增量方式实现教材编写的滚动、增加和淘汰，逐步建设可供地方高校计算机不同专业、针对不同领域培养计算机工程技术人才选择的教材库：①所有专业的公共基础课相对统一，不同专业的专业基础课按模块划分、各自专业的专业课按领域整合、拓展课紧跟技术和行业发展；②公共基础课、专业基础课以经典知识为主，专业课、拓展课与国际主流技术接轨；③实践环节或实践课程必须接纳企业文化、优选企业实际工程项目，体现校企合作、重视企业导师的参与。

“卓越工程师教育培养计划”的实施具有三个特点：一是行业企业深度参与培养过程；二是学校按通用标准和行业标准培养工程人才；三是强化培养学生的工程能力和创新能力。

本系列教材的编写得到了中软国际、苏州花桥国际商务区（及所属企业）、常州创意产业基地（及所属企业）等热心和关注计算机类人才培养的国家重点企业、园区的大力支持。我们曾以“目标明确、责任共担、实现共赢”为原则探索了多种人才培养合作途经：从师资培养到校企共建实训基

地，到建立校内软件学院，再到学生进企业强化、顶岗实训……取得了一定的经验。在“卓越工程师教育培养计划”的实施中，企业和学校签订了全面合作协议，共同确定人才培养标准、制订人才培养方案、参与人才培养过程，提供企业学习课程和项目案例，确保学生在企业的学习时间。

同样，本系列教材的编写总结了参编高校和支撑企业在校企合作人才培养过程中共同取得的经验和教训，并涵盖了我们已经做的、想要做的实施卓越计划的理念和努力。这仅是初步的尝试，会存在许多不足和缺陷，但希望由此能起到抛砖引玉的作用。在卓越计划的实施探索中，我们衷心地希望能有更多的地方高校计算机院系、更多的行业企业加入团队，面对企业必须参与的国际化产业竞争，为培养优秀的、具有应用创新精神的计算机工程技术（包括软件）人才，企业和学校能深度合作、各尽职责；每一位教育工作者都能贡献自己的聪明才智，尽一份绵薄之力。

对给予本套丛书编审大力支持的江苏计算机学会、中国矿业大学计算机学院以及参与编写教材的高校、单位表示由衷的感谢！

计算机“卓越工程师计划”应用型教材编委会

前　言

2000 年 6 月 22 日，微软公司正式推出了其下一代的计算计划：Microsoft.NET。.NET 作为新一代互联网软件和服务战略，将使微软现有的软件不仅适用于传统的个人计算机，而且能够满足在网络时代呈现强劲增长的新设备的需要。微软官员把对.NET 定义为代表一个集合，一个环境，一个可以作为平台支持下一代 Internet 的可编程架构。.NET 平台提供了大量的工具和服务，能够最大限度地发掘和使用计算及通信能力。

配合.NET，微软推出了一种新的程序语言——C#。它是从 C/C++演变而来的，是一种现代的面向对象的程序开发语言，使得程序员能够在新的微软.NET 平台上快速开发种类丰富的应用程序。由于其一流的面向对象的设计，从构建组件形式的高层商业对象到构造系统级应用程序，你都会发现 C#将是最合适的选择。在.NET 运行库的支持下，.NET 框架的各种优点在 C#中表现得淋漓尽致。

作为一门高质量的开发语言，很多高校纷纷开设了相关课程，但是很少有 C#结合面向对象、WinForm、ADO.NET、LINQ TO SQL、文件、多线程等内容开发 Windows 应用，并且符合高校课程设置的图书。

本书由多年从事一线程序设计与开发的软件项目开发人员和教学经验丰富的教师根据教学实践和开发心得进行组织编写，从软件开发方法、基础的编程语言知识开始，逐步介绍面向对象的初级、高级特性，并结合.NET 提供的类库和控件介绍使用 C#开发各种应用程序。通过本书的学习，读者将对 C#、面向对象的基本概念和开发方法有一个比较全面的认识和了解，并能应用 C#开发应用程序。

全书分为 9 章。

第 1 章简要介绍了目前常用的软件开发方法、面向对象技术基本概念及面向对象的分析与设计过程。

第 2 章首先简述了.NET Framework，对其特性做了简要概述，介绍了开发环境 Visual Studio.NET；然后对 C#语言基础进行了介绍，主要内容包括 C#基本类型、变量和常量、数组、操作符、表达式、流程控制语句、函数等编程语言的基本要素。

第 3 章首先简要介绍了窗体、标签、按钮及文本框简单控件对象；然后介绍面向对象的初级特性，包括如何使用 C#定义类、创建对象、销毁对象、C#方法的创建和调用，C#的静态成员和静态类，在 C#中实现类的继承和多态，其中包括抽象类、密封类等特殊类；接下来对 C#接口进行了介绍。

第 4 章给出了面向对象的高级特性，包括命名空间、委托、泛型，事件原理、定义和使用过程，常用集合的使用。

第 5 章讨论了如何创建传统的 Windows 应用程序，同时介绍了 Windows Form 的各种可用资源，如控件、菜单和对话框等。

第 6 章介绍如何使用文件，包括文件和目录操作的类，文本文件和二进制文件的创建、读写。同时介绍了序列化对象及其应用，XML 文档编程。

第 7 章介绍多线程的概念、线程的创建和控制以及同步等知识。

第 8 章介绍图形图像编程（GDI+）、Graphics 类、Pen 类和 Brush 类的使用。

第 9 章首先探讨了如何使用 ADO.NET 进行数据访问，介绍了 ADO.NET 中的数据提供程序

和 DataSet 对象，以及如何利用这些对象访问数据；介绍了如何使用多层架构来搭建数据访问应用程序，如何使用数据绑定技术简化数据的填充过程；最后介绍了 LINQ 编程，说明了如何使用 LINQ 处理对象，如何把 LINQ 应用于查询和数据处理。

本书适合作为学生的教材，主要面向那些希望学习 C#、没有面向对象概念而且缺乏开发经验的学生及初学者。与现有其他教材相比，本书从应用与工程实践的角度出发，重在激发学生的学习兴趣和将所学知识应用于程序开发实际能力的培养。通过本书的学习，可以达到以下目的。

（1）熟悉 Visual Studio.NET 开发、调试应用程序的步骤和方法。

（2）掌握 Windows 应用程序设计的方法和技巧。

（3）在项目中贯穿面向对象程序设计的思想，掌握使用 C#进行面向对象程序设计的方法。

（4）学会利用 I/O 流进行文件系统的访问和操作。

（5）掌握常见图形应用程序的编写方法。

（6）掌握多线程在应用程序设计中的重要性和方法。

（7）掌握 ADO.NET 对象模型体系、LINQ 编程以及学会使用多层架构搭建数据库应用程序的设计方法。

本书由王文琴、费贤举、李亦飞、唐学忠编著。第 1、2 章由费贤举编写；第 3、4 章由李亦飞编写；第 5、6 章由王文琴编写；第 7、8 章由唐学忠编写；第 9 章由王文琴、李亦飞共同完成。此外，史书明、蒋小莺、胡智喜等人也参与了本书的编写，为部分内容提出宝贵意见，在此一并感谢。同时在编写本书过程中参阅大量的文献资料和网站资料，在此对提供这些资料的作者也表示感谢。

本书文稿的录入，程序编写、运行以及插图的截取都是在 Windows 环境下同步进行的，所有程序都已在 Visual Studio.NET 2010 中文版环境中调试运行通过。

由于时间和水平的关系，书中错误和不当之处在所难免，敬请广大读者批评指正，并请将信息反馈给我们（wendycgy@163.com），不胜感激。

最后感谢电子工业出版社责任编辑刘海艳的辛勤工作。

编　者

2015 年 1 月

目　录

第1章

软件开发方法与面向对象概述

本章要点

- 了解目前常用的软件开发方法；
- 比较不同软件开发方法的适应场合；
- 理解面向对象技术基本概念及面向对象的分析与设计过程；
- 理解常见的面向对象方法与工具及异同。

1.1 软件开发方法概述

1.1.1 面向过程的开发方法

面向过程的开发方法是由 E.Yourdon 和 L.L.Constantine 在 1978 年提出的，即所谓 SASD 方法，也可称为面向功能的软件开发方法或面向数据流的软件开发方法，是 20 世纪 80 年代使用最广泛的软件开发方法。1979 年 TomDeMacro 对此方法作了进一步完善，提出了一组提高软件结构合理性的准则，如分解与抽象、模块独立性、信息隐蔽等。它首先用结构化分析（SA）对软件进行需求分析，然后用结构化设计（SD）方法进行总体设计，最后是结构化程序设计（SP）。这一方法不仅开发步骤明确，SA、SD、SP 相互承启，一气呵成，而且它给出了两类典型的软件结构（变换型和事务型）使软件开发的成功率大大提高。

结构化方法强调过程抽象和功能模块化，将现实问题域映射为数据流和加工，加工之间通过数据流来通信，数据流作为被动的实体被主动的操作所加工，以过程（操作）抽象为中心来构造系统和设计程序。结构化分析产生功能规约。它一般利用图形表达用户需求，使用的手段主要有数据流图、数据字典、结构化语言、判定表以及判定树等。结构化设计阶段用结构化程序设计语言实现分阶段表示出来的用户需求。

1.1.2 面向数据结构的开发方法

面向数据结构的开发方法适合求解算法取决于问题描述的数据结构的情况。这种开发方法最终的目标是得到对程序处理过程的描述，它并不明显地使用软件结构的概念，也不注重模块独立原则，模块只不过是设计过程的副产品。因此，这种方法最适合在完成了软件结构设计之后，用它来设计每个模块的处理过程。

使用面向数据结构的开发方法是根据问题的数据结构定义一组映射，把问题的数据结构转换为问题求解的程序结构。首先分析确定数据结构，并且用适当的工具清晰地描绘数据结构。常用的面向数据结构的开发方法有 Jackson 方法和 Warnier 方法。

1．Jackson 方法

1975 年，英国人 M.A.Jackson 提出的 Jackson 方法是最典型的面向数据结构的软件开发方法。Jackson 方法把问题分解为可由三种基本结构形式表示的各部分的层次结构。三种基本的结构形式是顺序、选择和重复。三种数据结构可以进行组合，形成复杂的结构体系。这一方法从目标系统的输入、输出数据结构入手，导出程序框架结构，再补充其他细节，就可得到完整的程序结构图。这一方法对输入、输出数据结构明确的中小型系统特别有效，也可与其他方法结合，用于模块的详细设计。

Jackson 方法在设计比较简单的数据处理系统时特别方便，当问题比较复杂时，常常遇到输入数据可能有错、条件不能预先测试、数据结构冲突等问题。为解决这些问题，还需要采取一系列比较复杂的辅助技术。

2．Warnier 方法

1974 年，J.D.Warnier 提出的软件开发方法与 Jackson 方法类似。差别主要有三点：①使用的图形工具不同，分别使用 Warnier 图和 Jackson 图；②使用的伪码不同；③在构造程序框架时，Warnier 方法仅考虑输入数据结构，而 Jackson 方法不仅考虑输入数据结构，而且还考虑输出数据结构。

1.1.3 面向对象的开发方法

20 世纪 90 年代计算机业界最流行的几个单词就是分布式、并行和面向对象这几个术语，由此可以看出面向对象（Object Oriented，OO）这个概念在当前计算机业界的地位。现代软件工程对软件开发模式与方法产生了新的需求，面向对象的开发方法逐渐成为计算机软件界青睐的主流开发方法。

面向对象的开发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，它是建立在对象概念基础上的方法。面向对象技术是软件技术的一次革命，在软件开发史上具有里程碑的意义。面向对象的开发方法基本思想是：对问题空间进行自然分割，以更接近人类思维的方式建立问题域模型，以便对客观实体进行结构模拟和行为模拟，从而使设计出的软件尽可能直观地描述现实世界，构造出模块化的、可重用的、维护性好的软件，同时限定软件的复杂性和降低开发维护费用。

面向对象的软件开发方法是通过面向对象的分析（OOA）、面向对象的设计（OOD）以及面向对象的程序设计（OOP）等过程，将现实世界的问题空间平滑地过渡到软件空间的一种软件开发过程。面向对象的开发方法建立系统模型的基本思想是自底向上的归纳和自顶向下的分解相结合，以对象建模为基础，不仅考虑了输入、输出数据结构，也包含了所有对象的数据结构。此外，OO 技术在需求分析、可维护性和可靠性这三个软件开发的关键环节和质量指标上有了实质性的突破，彻底地解决了在这些方面存在的严重问题。体现在如下方面。

（1）面向对象的开发方法的基础是对象模型，每个对象类由数据结构（属性）和操作（行为）组成，相关的所有数据结构（包括输入、输出数据结构）都成了软件开发的依据。面向对象的开发方法解决了需求分析这一问题，因为需求分析过程与系统模型的形成过程一致，开发人员与用户的讨论是从用户熟悉的具体实例（实体）开始的。

（2）面向对象的开发方法关注的是目标系统的对象模型，而不是功能的分解。功能是对象的使用，它依赖于应用的细节，并在开发过程中不断变化。由于对象是客观存在的，因此当需求变化时对象的性质要比对象的使用更为稳定，从而使建立在对象结构上的软件系统也更为稳定。

（3）面向对象的开发方法解决了软件的可维护性。在 OO 语言中，子类不仅可以继承父类

的属性和行为，而且也可以重载父类的某个行为。利用这一特点，可以方便地进行功能修改，如引入某类的一个子类，对要修改的一些行为进行重载，也就是对它们重新定义。由于不再在原来的程序模块中进行修改，所以彻底解决了软件的可修改性，从而也彻底解决了软件的可维护性。

目前，典型的面向对象的开发方法是 UML 和统一开发过程（RUP）。

1.2　软件开发方法的评价与选择

1.2.1　软件开发方法的评价

传统的软件开发方法的本质，是在具体的软件开发工作开始之前，通过需求分析预先定义软件需求，然后一个阶段接着一个阶段有条不紊地开发用户所要求的软件，实现预先定义的软件需求。对某些类型的软件开发，传统的软件开发方法比较适用；但对那些大型复杂的软件系统和系统需求经常变化的项目，传统的软件开发方法有许多不足之处。面向对象的开发方法与传统的软件开发方法相比较，具有以下优点。

（1）面向对象的方法与人类习惯的思维方法一致

传统的结构化开发方法以算法为核心，把数据和过程作为相互独立的部分。功能与数据分离的软件设计结构必然导致对现实世界的认识与编程之间存在着一道很深的理解上的鸿沟。面向对象的开发方法以对象为核心，对象之间通过传递消息互相联系，以模拟现实世界中不同事物彼此之间的联系。

（2）稳定性好

传统的结构化开发方法以算法为核心，开发过程基于功能分析和功能分解。软件系统的结构紧密依赖于系统所要完成的功能，当功能需求发生变化时将引起软件结构的整体修改。面向对象的开发方法以对象为中心构造软件系统，用对象模拟问题领域中的实体，以对象间的联系刻画实体间的联系。当系统的功能需求变化时，往往只需要作一些局部性的修改。这样的软件系统比较稳定。

（3）可重用性好

传统的结构化开发方法用标准函数库中的函数作为“预制件”来建造新的软件系统，但标准函数缺乏必要的“柔性”，不能适应不同的应用场合，不是理想的可重用的软件成分。面向对象的开发方法可以用以下两种方法重用一个对象类：①创建该类的实例，从而直接使用它；②从已有类派生出一个满足当前需要的新类。继承性机制使得子类不仅可以重用其父类的数据结构和程序代码，而且可以在父类代码的基础上方便地修改和扩充，这种修改并不影响对原有类的使用。

（4）较易开发大型软件产品

用传统的结构化开发方法开发涉及多种不同领域知识的大型软件系统，或开发需求模糊或需求动态变化的系统时，所开发出的软件系统往往不能真正满足用户的需要。主要表现在：①开发人员不能完全获得或不能彻底理解用户的需求，以致开发出的软件系统与用户预期的系统不一致，不能满足用户的需要；②开发出的系统不能适应用户需求经常变化的情况，软件系统往往不能真正满足用户需要。面向对象的开发方法可以将大系统分解成相互独立的小产品来处理，可以降低成本、提高质量。面向对象的软件稳定性好、比较容易理解和修改，易于测试和调试、易于维护。

1.2.2　软件开发方法的选择

那么在软件系统开发的时候是否应该放弃传统的软件开发方法而选择面向对象的开发方法呢？从前面的论述可以看到，面向过程的结构化方法更接近于计算机世界的物理实现，而面向对

象的方法更符合人类的认识习惯。相对传统的结构化方法来说，面向对象的方法具有更多的优势。但任何软件开发方法都不是十全十美的，结构化程序设计方法经过三十多年的发展和积累，已经有了一整套的工具和成熟的方法学，在发展其他软件方法的同时绝对不能全盘抛弃它。

从执行效率来说，结构化方法比面向对象方法产生的可执行代码更直接、更高，所以对于一些嵌入式的系统，结构化方法产生的系统更小、运行效率更高。从掌握难度来说，面向对象方法比结构化方法复杂，难于理解。面向对象的方法内容广、概念多，要经过长期的开发实践才能很好地理解、掌握。相比之下，结构化方法知识内容少，容易上手。从应用的范围看，结构化方法适用于数据少而操作多的问题。实践证明：对于操作系统这类以功能为主的系统，结构化方法比较适合；对于数据库、信息管理等以数据为主、操作较少的系统，用面向对象方法描述要好于结构化方法。

综上所述，开发者在开发实践中，应从实际出发，考虑执行效率、开发者的技术水平、系统规模、是否为需求易变化的系统等因素，尽量利用它们各自的优点，避免它们的缺点。例如，对于开发一些小型嵌入式实时监控系统或类似稳定小系统，可用结构化方法；对于开发入门者，使用结构化方法和面向对象方法相结合；对于大型系统或者需求易变系统，使用面向对象方法。系统开发的方法随着系统开发工具的不断改进，正在逐渐完善，特别在大型系统的开发中，常常不是采用一种开发方法，而是采用多种方法的组合。各种方法不是相互独立的相互排斥的，相反它们是相互促进相互补充的，它们经常可以混合使用以获得更好的效果。总之，根据实际出发，选取合适的软件开发方法，达到最佳的开发效益。

1.3 面向对象技术

1.3.1 面向对象方法的特点

面向对象的方法学可以概括为下列方程：

OO=Objects+Classes+Inheritance+Communication with Messages

也就是说，面向对象使用了对象、类和继承等机制，而且对象之间仅能通过传递消息实现彼此通信。面向对象方法以对象为中心，它有以下几个基本特点。

（1）客观世界是由各种对象组成的，任何事物都是对象，复杂的对象可以由比较简单的对象以某种方式组合而成。从问题领域的客观事物出发来构造软件系统，用对象作为对这些事物抽象的表示，并作为软件系统的基本构成。

（2）事物的静态特征（即数据的表达特征）用对象的属性表示，事物的动态特征（即事物的行为）用对象服务表示（即方法）。对象的属性与服务结合成一体，成为一个独立的实体，对外屏蔽其内部细节（称作封装）。对事物进行分类，把具有相同属性和相同服务的对象归成一类，类是这些对象的抽象描述，每个对象是它的类的一个实例。

（3）通过在不同程度上运用抽象的原则（较多较少忽略事物之间的差异）可以得到较一般（父类）和特殊（派生类）的类，特殊的类继承一般的类的属性与服务。面向对象方法支持对这种继承关系的描述与实现，从而简化系统的构造过程及其文档。但是，如果在派生类中对某些特性和服务操作又做了重新描述，则在派生类中的这些特性和服务操作将以新描述为准，也就是说，底层的特性将屏蔽高层的同名特性。

（4）复杂的对象可以用简单对象作为其构成部分（称作聚合），对象之间通过消息进行通信以实现对象之间的动态联系，通过关联表达对象之间的静态联系。对象与传统的数据有本质区别，它不是被动地等待外界对它施加操作。相反，它是进行处理的主体，必须发消息请求它执行

某个操作，处理它的私有数据，而不能直接从外界对私有数据进行操作，这就是封装性。这种灵活的消息传递方式，便于体现并行和分布式结构。

1.3.2　面向对象的基本概念

1．对象和类

（1）对象

面向对象方法学中的对象是由描述该对象属性的数据以及可以对这些数据施加的所有操作封装在一起构成的统一体。对象的操作表示它的动态行为，在面向对象分析和面向对象设计中，通常把对象的操作称为服务或方法。人们要进行研究（感兴趣）的任何事物，从最简单的整数到航天飞机都可以看成对象。对象可以是以下形式。

- 有形的实体，指一切看得见摸得着的实物。
- 作用，指人或组织所起的作用，如医生、公司、部门等。
- 事件，在特定时间发生的事，如飞行、演出、开会等。
- 性能说明，如机床厂对机床的性能说明。

对象的形象表示如图 1-1 所示。

实现对象操作的代码和数据是隐藏在对象内部的。一个对象好像是一个黑盒子，表示它内部状态的数据和实现各个操作的代码及局部数据，都被封装在这个黑盒子内部，在外面是看不见的，更不能从外面去访问或修改这些数据或代码。

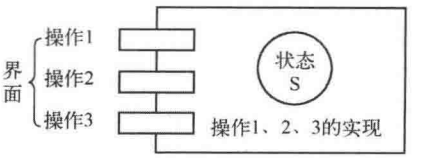


图 1-1　对象的形象表示

使用对象时只需知道它向外界提供的接口形式而无须知道它的内部实现算法，不仅使得对象的使用变得非常简单、方便，而且具有很高的安全性和可靠性。对象内部的数据只能通过对象的公有方法来访问或处理，这就保证了对这些数据的访问或处理，在任何时候都是使用统一的方法进行的。

从不同的角度可以给出对象的不同定义，目前比较常用的定义是：对象是封装了数据结构及可以施加在这些数据结构上的操作的封装体，这个封装体有可以唯一地标识它的名字，而且向外界提供一组服务（即公有的操作）。一个对象可以有多个属性和多项服务。

（2）类

人类习惯把有相似特征的事物归为一类，分类是人类认识客观世界的基本方法。在面向对象的软件技术中，将那些在程序执行期间除了状态之外其他方面都一样的对象归结在一起，构成对象的“类”，类描述了一组有相同属性和相同行为的对象。因此，定义一个类需要同时定义这个类的属性（说明该类对象的特性）和服务（说明该类对象的行为）。

由某个特定的类所描述的一个具体的对象被称作实例。实际上类是建立对象时使用的“模板”，按照这个模板所建立的一个个具体的对象，就是类的实际例子，通常称为实例。类是对具有相同属性和行为的一组相似的对象的抽象，类在现实世界中并不能真正存在。类反映对象的共同性定义，而对象则是满足该定义的一个具体的实例化的个体。例如，“人”类是满足所有人的属性和行为的抽象，而把“人”类的所有属性赋予具体的数据就代表了一个具体的人物，如张三、李四和王五都是类“人”的实例。

2．封装

封装是面向对象的一个重要方面。它有两层含义：①把对象全部属性和全部服务结合在一起，形成一个不可分割的独立单位；②信息隐藏，即尽可能隐藏对象的内部细节，只保留有限的

对外接口使对象与外部发生联系。数据的表示方式和对数据的操作细节被隐藏起来，用户通过操作接口对数据进行操作，这就是数据的封装。封装体现了“信息隐藏和局部化”原则。使用一个对象的时候，只需知道它向外界提供的接口形式，无须知道它的数据结构细节和实现操作的算法。对象的封装性具有如下特点。

（1）有一个清晰的边界。所有私有数据和实现操作的代码都被封装在这个边界内，从外面看不见更不能直接访问。

（2）有确定的接口（即协议）。这些接口就是对象可以接受的消息，只能通过向对象发送消息来使用它。

（3）受保护的内部实现。实现对象功能的细节（私有数据和代码）不能在定义该对象的类的范围外访问。

封装具有很重要的意义，它使得软件的错误和修改可以局部化。举例来说，某个用面向对象概念封装良好的软件包被设计成为为上一层的程序员提供各种底层支持，这时上层的程序员不必去关心底层的库的实现细节，他只是简单地使用底层的软件包提供的对象和服务就可以了，而当底层的库因为硬件或者某些因素改变后，只要它表现出的外部特性不变，上层的程序员就不必去修改自己的代码。

3．继承

继承表示了基本类型和派生类型之间的相似性。在面向对象的软件技术中，继承是子类自动地共享父类中定义的属性和方法的机制。继承简化了对现实世界的认识和描述，在定义子类时不必重复定义那些已在父类中定义的属性和方法，只要声明自己是某个类的子类，把精力集中于定义本子类所特有的属性和方法上即可。例如在形状问题中，基本类型是形状，每个形状都有大小、位置、颜色等特性，每个形状都能被绘制、擦除、着色等。由此还可以派生出特殊类型的形状：圆、正方形、三角形等，它们每一个都有一些特殊的特性和行为。继承关系既体现了形状间的联系，又体现了它们之间的区别。

继承具有传递性。如果类 C 继承类 B，类 B 继承类 A，则类 C 继承类 A。一个类实际上继承了它所在的类等级中在它上层的全部基类的所有描述。也就是说，属于某类的对象除了具有该类所描述的性质外，还具有类等级中该类上层全部基类描述的一切性质。

另外还有一个重要的概念是多继承，也就是一个类可以是多个基类的派生类，它从多个基类中继承了属性和服务，这种继承模式叫做多继承。

继承性使得相似的对象可以共享程序代码和数据结构，从而大大减少了程序中的冗余信息。派生出新的子类的办法，使得对软件的修改变得更加容易。继承提高了软件的可复用性。用户在开发新的应用系统时不必完全从零开始，可以继承原有的相似系统的功能或从类库中选取需要的类，再派生出新的类以实现所需要的功能。

4．消息

现实世界是由许多不同的对象组成的，它们之间有各种联系和交互。对象之间的联系称为对象的交互；一个对象向另一个对象发出的请求称为消息。消息是要求对象进行动作的说明、命令或指导，是对象之间相互请求或相互协作的途径。把发送消息的对象称为发送者，接收消息的对象称为接收者。对象间的联系，只能通过传递消息来进行。对象也只有在收到消息后才能选用方法而被激活。

面向对象技术的封装机制使对象各自独立，对象之间通过消息互相联系、互发消息、响应消息、协同工作，实现系统的各种服务功能。消息具有三个性质：

（1）同一个对象可以接收不同形式的多个消息，做出不同的响应。