中国传媒大学“十一五”规划教材

应用软件设计教程

徐品 李绍彬 蓝善祯 编著

中国广播电视出版社
CHINA RADIO & TELEVISION PUBLISHING HOUSE

中国传媒大学“十一五”规划教材

应用软件设计教程

徐 品
李绍彬
蓝善祯
编著

中国广播电视出版社
CHINA RADIO & TELEVISION PUBLISHING HOUSE

图书在版编目（CIP）数据

应用软件设计教程／徐品，李绍彬，蓝善祯编著．—北京：中国广播电视出版社，2009.6
中国传媒大学“十一五”规划教材
ISBN 978-7-5043-5791-5

Ⅰ.应… Ⅱ.①徐…②李…③蓝… Ⅲ.软件设计—高等学校—教材Ⅳ.TP311.5

中国版本图书馆CIP数据核字（2009）第027051号

应用软件设计教程
徐品　李绍彬　蓝善祯　编著

责任编辑	王本玉
封面设计	郭运娟
版式设计	张智勇
责任校对	张　哲
出版发行	中国广播电视出版社
电　　话	010-86093580　010-86093583
社　　址	北京市西城区真武庙二条9号
邮政编码	100045
网　　址	www.crtp.com.cn
电子信箱	crtp8@sina.com
经　　销	全国各地新华书店
印　　刷	高碑店市鑫宏源印刷包装有限责任公司
开　　本	787毫米×1092毫米　1/16
字　　数	384（千）字
印　　张	17
版　　次	2009年6月第1版　2009年6月第1次印刷
印　　数	5000册
书　　号	ISBN 978-7-5043-5791-5
定　　价	31.00元

前 言

“软件设计”是学习如何设计一个软件，是“软件工程”中的重要一环。但是要说清楚如何设计一个软件却不是一件很容易的事。在应用软件方面软件的类型实在太多了，按开发规模分类，有个人、3 人左右的小团队、5～10 人的中等团队、企业式团队（几十人）等；按网络方式分类，有单机、C/S（客户机/服务器）、B/S（浏览器/服务器）等类型；按项目来源分类，则有自创自用型、科研型、内部使用型、商业型等；按适用范围可分为独家订做、通用型、行业型，等等。由于各种类型软件的要求不同，实现的目标不同，对软件开发的质量不同，当然，编写方式也是不同的。软件设计课程通常会告诉你编制软件所要遵循的原则，软件开发需要经历哪些工程。但不能期望学完了软件设计，什么软件都能设计了。应该说，通常软件设计这个课程属于方法论范畴，而不是如同电路设计、程序语言设计那样让你学会一项技能。

很多读者期望通过软件设计课程的学习，就能学会如何去设计一个软件，但事实上并不那么理想。现在，很多软件设计的论著都致力于对软件开发过程的总结，力求在方法论上找出软件设计的规律，用于指导学习者进行软件设计。这方面最重要的成果应该是 UML（统一建模语言）。这些抽象的法则和设计方法无疑是非常重要的，在本书中也有专门的介绍。但如果对一个没有太多的软件开发经验的大学生或研究生，尤其是非计算机专业的学生来说，这些理论实在是太抽象了，以至于很多学完软件设计的学生感觉没有实质性的收获。这些书本上的软件设计的知识也许要等到从事软件开发两三年以后才有体会，如果那时还记得这些知识的话。

但是，要求学习者一定要等到积累了一定经验以后再来学习软件设计的方法是不现实的。根据作者多年的软件开发经验，并通过几年的教学实践，我们认为，对于没有足够编程经验的读者来说，软件设计课程也是可以学习的；但在学习抽象的软件设计方法之前还是要学一些基本的软件设计技巧，让学习者积累一定的经验，然后再去理解抽象的方法论。

本书的主要对象是学过C++ 语言但没有太多开发经验的学生，特别是非计算机专业的学生。可作为大学四年级或新入学的研究生学习软件设计的教材。

本书分四个部分，共 11 章。本书采用以实例为主，力求将抽象的设计方法融入到具体程序实现中，让读者从实例中学习软件设计的方法。本书所采用的程序语言是 C++，开发工具是VC++，软件设计实例是单机运行的绘图软件。

第一部分是“程序设计基础知识”,是由第1、2两章组成。主要是C++语言和数据结构的知识提要,供那些C++基础不太好,或学得不够深入的学生复习和深入学习用。其中着重介绍运算符重载、多态性、模板、线性表等概念。如果对这方面很熟的学生,则可以跳过或粗略浏览一遍即可。

第二部分是“MFC编程技术”,是由第3、4、5三章组成。主要介绍MFC程序调试方法、基本原理及基本的开发技术。使读者对MFC程序有一个比较深入的了解。MFC程序结构提供了一个很好的软件设计范本,我们在了解它的编程技术的同时,也会对MFC程序结构有一个比较深刻的印象,这对软件设计的学习是非常有好处的。

第三部分是“软件开发实例”,是由第6、7、8三章组成。这部分将引导读者开发一个绘图软件系统。该软件虽然很小,但其中包含的数据结构、数据的管理、程序流程等对软件设计的学习者都非常具有借鉴作用。

第四部分是“软件工程与软件设计”,是由第9、10、11三章组成。这部分将结合前面程序的例子,介绍软件工程的概念和软件设计的方法。后面还着重介绍了UML统一建模语言。最后还结合开发实例给出了UML的设计方法。从一个具体的开发实例中获得软件设计的思路,从而能够理解软件设计的理论。这就是本书所要追求的目标。其中第一和第二部分(1至5章)是由徐品老师完成;第三部分(6至8章)是由蓝善祯老师完成,第四部分(9至11章)是由李绍彬老师完成。

本书在编写过程中得到了中国传媒大学南广学院的周晓梅和段洪秀老师的帮助,他们投入了大量的精力参加了本书部分资料搜集、整理工作,在此向他们表示衷心的感谢。中国传媒大学信息工程学院的研究生袁大钧、匡红梅同学对其中部分书稿核对和代码验证工作,在此表示感谢。由于编者水平有限,时间紧张,加上新技术不断涌现,书中难免存在错误或不妥之处,恳请广大读者批评指正。

本书有关参考程序请到以下网址下载:http://www.crtp.com.cn

徐 品

2008年12月于北京

目 录

第一部分 程序设计基础知识

第二部分 MFC 编程技术

第四部分　软件工程与软件设计

第一部分

程序设计基础知识

程序设计

第一部分

海洋设计基础知识

第1章

C++语言提要

1.1 概述

让我们简单回顾一下计算机语言的发展历程。第一个阶段是汇编语言,这个面向机器的语言。汇编语言解决了用便于理解的缩写字母来代替二进制的机器码的问题。汇编的每一条语句对应机器的一条指令,便于执行但不便于编写和阅读。第二阶段是面向过程的语言。这时候所解决的问题是将指令过程函数化,以及运算过程高级化(采用人类容易理解的四则运算等运算方式,并引进了与具体机器无关的+、-、*、/等运算符)。这种改进使得程序过程清晰明了,便于阅读和编写。但当程序过于庞大时,代码的重复率提高,数据和函数得不到有效的保护,代码再利用不方便等缺点也暴露出来,于是出现了第三阶段的面向对象的语言。面向对象的语言将面向过程语言中杂乱的数据和函数,整齐地包装成一个个自我完备的对象,提高了代码的可读性、程序的健壮性以及代码的可重用性。时至今日,尽管计算机语言还在不断地发展,面向对象的程序设计方法一直是现代计算机语言的一个基本理念之一。从20世纪90年代发展起来的面向组件的技术就是基于面向对象技术发展出来的软件设计的新方法。

面向对象语言的最主要的特点是封装、继承和多态性,称为面向对象思想的三基石。作为面向对象语言的代表,C++语言一方面以完全兼容的方式牵手C语言,另一方面,又全方位地实现了面向对象的思想。当应用软件发展到一定规模,如果没有面向对象的语言是很难构建程序的。当回顾C++的全貌,我们应该认识到,从程序设计的角度来看,其实C++与C语言是完全不同的两种语言。C++更注重设计,而不是过程。因此,在程序设计思路上,C++语言与C语言有着很大的不同。比如编写一个图书管理程序,用C语言的思路,针对每一个功能(图书录入、检索、借阅等)编制函数。如果系统庞大、功能复杂,会导致程序可读性差、代码重复性高、不易修改等问题。C++按照对象的方式进行管理,整体上可读性好,又具有代码重用、升级方便等特点,可使软件设计人员在更高的层次上管理程序。

本章不打算全面介绍C++的知识,只是用一些实例来复习一下C++的主要概念,并对C++某些难点进行梳理。对于从未学过C++的读者,我们建议先学一遍C++再看本章。

1.2 类与对象

C++是面向对象的语言,所以它最重要的概念就是对象。对象是什么?对象就是一种封装,是对属性和行为的封装。封装的目的是要保证对象的健壮性。要说明这个问题,先从C语言的结构说起。

1.2.1 类的构造函数

我们来看一个结构(struct)的定义。例1-1中结构PERSON有一个成员变量age,这个结构就是一种封装。但这个封装是不完整的,因为它无法保证结构内数据age的合法性,调用者可以对其随意赋值。

再看例1-2中C++中类(class)的封装。在这个类CPerson中,将变量Age放在private保护区,使外部只能通过Get和Set函数访问变量,保证了成员变量Age在设置过程中不小于0。这与前面定义的结构PERSON相比提高了封装性,使得外部调用者不得随意改变Age的值。

然而,要使得该类的对象在任何时候都能够使Age不小于0,这个类还是有一个问题,那就是初始化!如果我们不对这些变量进行初始化,系统会随机分配一个值给成员变量,则该类可能一开始Age的值就是不合法的。所以就有了构造函数(constructor)的概念。CPerson加上构造函数如例1-3。

例1-1 结构PERSON的定义

```
struct PERSON
{
    int age; // 成员变量为public
};
```

例1-2 类CPerson的定义

```
class CPerson
{
    int Age; // 缺省成员变量为private
public:
    int GetAge() { return Age; }
    void SetAge(int age) { Age = (age < 0) ? 0 : age; }
};
```

例1-3 具有构造函数的类 CPerson

```
class CPerson
{
public:
    CPerson() { Age = 0; }
    int GetAge() { return Age; }
    void SetAge(int age) { Age = (age < 0) ? 0 : age; }
private:
    int Age;
};
```

构造函数是在这个类创建对象时自动执行的,其目的是在对象创建时可以有机会对成员变量进行初始化,以保证对象各成员的合法性。

现在我们可以说这个类在任何时候都是健壮的,因为无论对该类怎么操作,Age 的值都是合法的,即 Age≥0。

构造函数有多种形式,如带默认形参的构造函数、拷贝构造函数等,如例1-4所示。

值得提出的是,拷贝构造函数如果不写,编译系统会给一个默认的拷贝构造函数,其行为就是逐个成员变量的拷贝(称为浅拷贝)。所以,例1-4中拷贝构造函数完全可以不写。

有一个问题必须提醒读者,在类的声明中是不能赋值的。如例1-3中类 CPerson 的成员变量 int Age;不能写成 int Age=20;,这是因为类的声明在编译时只生成一个数据类型,而不可以带有可执行代码。也许有的读者会问,类 CPerson 的构造函数、成员函数(如 GetAge()和 SetAge())不都是有可执行语句吗?不是也能赋值吗?其实,它们与 int Age=20 的语句是根本不同的。例如 int GetAge() { return Age; },其后面大括号中的语句实质上是要嵌入需要调用该函数的语句中,就像 inline 函数一样,可认为是 inline 函数的简化的写法。

例1-4 具有多种构造函数的类 CPerson

```
class CPerson
{
public:
    CPerson(int age=0) // 带默认形参的构造函数
        { Age = age; }
    CPerson(CPerson& person) // 拷贝构造函数
        { Age = person.age; }
    int GetAge() { return Age; }
```

```
    void SetAge(int age) { Age = (age < 0) ? 0 : age; }
private:
    int Age;
};
```

1.2.2 拷贝构造函数被调用的场合

拷贝构造函数是构造函数中一个比较不好理解的概念。关键的问题是拷贝构造函数是在什么时候被调用的？我们以例 1-4 为例说明拷贝构造函数的三个被调用场合。

① 拷贝构造函数被调用的第一个场合是用已有对象构造新的对象，如：

```
main()
{
    CPerson person1(18);
    CPerson person2(person1); // 调用拷贝构造函数
    cout << person2.Age << endl;
}
```

这个例子是以 CPerson 的第 1 个对象 person1 来构造它的第 2 个对象时会调用拷贝构造函数，其结果是使 person2.Age 也等于 18。

② 拷贝构造函数的第二个调用场合是作为函数的参数调用时，如：

```
void Function(CPerson person)
{  // 参数 person 的传递过程实际就是重新构造新的对象的过程。
    int age = person.Age;
    cout << age << endl;
}
```

当这个函数被调用时，类 CPerson 的对象 person 作为实参传递进来（称为传值调用），实际上就重新构造了一个新的对象。在构造过程中就自动调用了拷贝构造函数。

③ 还有一个拷贝构造函数的调用场合就是在函数返回的时候。见下面代码：

```
CPerson Function()
{
    CPerson person(18);
    return person;
} // 函数返回一个类的对象，实际也是重新构造新的对象的过程。
```

在函数 Function 内部的对象 person 称为自动变量，它在函数结束时就自动析构了，函数返回时实际上是用拷贝构造函数构造了一个新的对象。

1.2.3 带有指针变量的类

有构造函数就有析构函数（destructor）。析构函数是在对象结束时被自动调用的。如果这个类有指针形式的变量，并且该变量用 new 方式申请了内存空间，则本着谁申请谁释放的原则，该变量申请的内存空间一般在对象结束时释放。例如例 1-5 所示：