文章编号:1006-2475(2005)02-0014-02

C++中构造函数的深入分析

张 永,叶水生,张小锋

(南昌航空工业学院计算机系,江西 南昌 330034)

摘要:深入分析了拷贝构造函数和转换构造函数的作用和实现方法,并指出了程序设计中出现的问题和解决方法。

关键词:拷贝构造函数;转换构造函数;资源

中图分类号:TP312 **文献标识码**:A

Deep Analysis of Constructor Function in C++

ZHANG Yong, YE Shui-sheng, ZHANG Xiao-feng

(Department of Computer, Nanchang Institute of Aeronautical Technology, Nanchang 330034, China)

Abstract: On the basis of deep analyzing the effect of copy constructor function and change constructor function, this article points out the problems in programming design and gives out the resolvents.

Key words: copy constructor function; change constructor function; resource

0 引 言

C++程序设计语言既可进行过程化程序设计,也可进行面向对象程序设计。C++实现了类的封装、数据隐藏、继承及多态,使得其代码容易维护及高度可重用。在类定义时,如果用户没有显示定义构造函数,则系统会提供一个默认的构造函数,仅负责创建对象,而不做任何初始化工作;用户可以显示定义构造函数,用来初始化数据成员和资源,包括定义不带参数的和带参数的构造函数,还可以定义转换构造函数和拷贝构造函数;转换构造函数能够实现将平台提供的基本数据类型变量赋给类型[1]对象,而拷贝构造函数解决了类对象之间的拷贝和资源冲突的问题。

1 默认构造函数与用户定义的构造函数

C++规定每个类必须有一个构造函数,没有构造函数,就不能创建任何对象,若用户未提供一个类的构造函数(一个都未提供),则C++提供一个默认的构造函数,该默认构造函数是个无参构造函数,它仅负责创建对象,而不做任何初始化工作;若用户对一个类定义了一个构造函数(不一定是无参构造函数),C++就不再提供默认的构造函数,也就是说,如果为类定义了一个带参数的构造函数,还想要无参构造函数,则必须自己定义。在用默认构造函数创建对象时,如果创建的是全局对象或静态对象,则对象的位模式全为0,否则,对象值是随机的[2]。

例如,下面的代码试图定义一个带参数的构造函数,若要创建无参对象,将不能正确地编译:

```
#include "string.h"
class Test
{public:
   Test(char *pName){strcpy(name,pName);}
protected:  char name[20];
};
void main()
{Test noName;  //无匹配的构造函数}
```

另外一个值得注意的是,如果用户要定义构造函数时,必须在public区定义,类定义时不能对其数据成员初始化,且结尾处一定要有一个分号。

为了解决上面这个问题,我们可以重载构造函数,再定义一个不带参数的构造函数,问题就解决了。如:Test(){strcpy(name,"zhangsan");}

收稿日期:2004-05-08

作者简介:张永(1977-),男,辽宁铁岭人,南昌航空工业学院计算机系助教,研究方向:面向对象;叶水生(1957-)男,江西南康人,计算机系教授,系副主任,硕士生导师,研究方向:计算机网络、计算机软件工程;张小锋(1963-),男,江西高安人,计算机系教授,系教研室主任,研究方向:计算机及应用。

2　拷贝构造函数与转换构造函数

可用一个对象去构造另一个对象,或者说,用另一个对象值初始化一个新构造的对象;函数的参数是对象时,都会涉及拷贝构造函数。拷贝构造函数的形参是本类的引用。而转换构造函数一般定义为含一个参数的构造函数(或相当于一个参数的构造函数),它可以实现将基本数据类型转化为类类型,在这两类构造函数使用过程中,经常涉及到临时对象和无名对象。当函数返回一个对象时,要创建一个临时对象以存放返回的对象,而无名对象一般是调用转化构造函数产生的。看下面的程序代码:

```
//hello.h
#include "iostream.h"
class stu
{public: stu(){cout << "Hello" << endl;}
stu(int m){kk = m;  cout << "structure" << kk << endl;}
//定义转化构造函数
stu(stu &s){  kk = s.kk + 4;cout << "copy" << kk << endl;} //定义拷贝构造函数
protected: int kk;};
```

事例一、

```
#include "hello.h"
stu tt()
{ stu ms = 4;cout << "clever" << endl;return ms;}
void main()
{stu d = 7;d = tt();}
```

运行结果:

```
structure7
structure4
clever
copy8
destroy4
destroy8
destroy8
```

由于函数 stu tt()的返回类型是对象,所以要调用一次拷贝构造函数来实现将 ms 赋给 tt 函数的临时对象,所以有第四行的输出(构造一个临时对象)和第六行的输出(析构临时对象),tt 函数中的局部对象 ms 通过拷贝构造函数赋给临时对象后才析构自己,然后临时对象把自己再赋给对象 d 后,再析构自己,所以对象 d 中的数据成员 kk 得到临时对象的数据成员 kk 的值 8,即析构时 d 的数据成员 kk 是 8(最后一行输出),过程如图 1 所示。

事例二、

```
#include "hello.h"
stu &tt()
{ stu ms = 4;cout << "clever" << endl;return ms;}
void main()
{stu d = 7;d = tt();}
```

运行结果:

```
structure7
structure4
clever
destroy4
destroy4
```

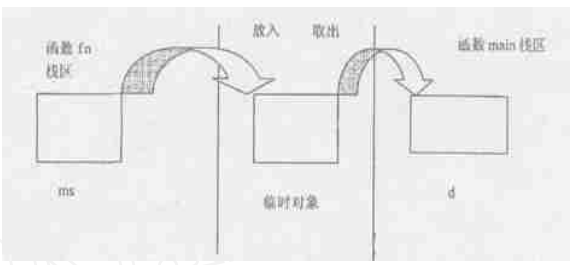

图 1　拷贝构造函烽过程

函数 tt 的返回类型是引用,不是对象,所以不会有临时对象生成,也就是说,对象 d 的数据成员 kk 直接得到对象 ms 数据成员 kk 的值 4,所以析构对象 d 时,其数据成员 kk 是 4(最后一行输出),而不是 8 了,过程如图 2 所示。

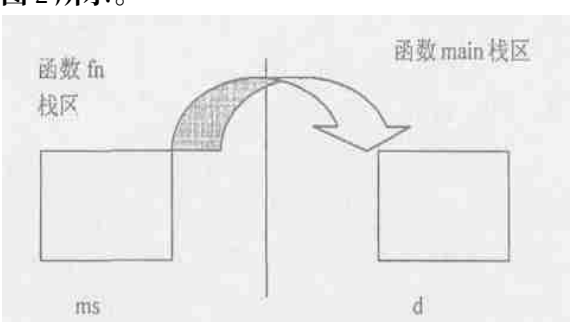

图 2　转换构造函数过程

3　拷贝构造函数与资源

类定义中,如果未提供自己的拷贝构造函数,则 C++提供一个默认拷贝构造函数,完成一个成员一个成员的拷贝。但是,一个类可能会拥有资源,如果拷贝构造函数简单地复制了一个该资源的拷贝,而不对它本身分配,就得面临一个错误:两个对象都拥有同一个资源,当对象析构时,该资源将经历两次资源返回,导致错误[3]。例如:

```
#include "iostream.h"
#include "string.h"
class person
{public:
  person(char *pn){cout << "constructing " << pn << endl;
pname = new char[strlen(pn) + 1];strcpy(pname,pn);}
person()
```

(下转第 21 页)

很大的灵活性,而 FrontController 应该谨慎使用,因为它无论在复杂度还是性能上,都需要付出较大的代价。顺便值得一提的是,Asp.Net 架构是极其灵活的,高效率地构建简单系统或是处理复杂的企业级应用都可以胜任,而 Struts 架构可以说天生就是为复杂应用服务的,它使复杂如 FrontController 模式的实现也变得轻而易举。

参考文献:

[1] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Patterns Elements of Reusable Object-oriented Software [M]. 北京:机械工业出版社. 2003.

[2] David Trowbridge, Dave Mancini, Dave Quick. 使用 Microsoft .NET 的企业解决方案模式 [J/OL]. http://www.microsoft.com/china/msdn/library/architecture/patterns/esp/esp.mspx,2004-04-21.

[3] Deepak Alur, John Crupi, Dan Malks. Core J2EE Patterns(第 1 版) [M]. Prentice Hall PTR, 2001.

[4] The Apache Struts Prouect Committee. Struts1.1 Users Guide [DB/OL]. http://jakarta.apache.org/struts/userGuide/index.html,2002-12-31.

(上接第 15 页)

```
{cout << "Destructing " << pname << endl;delete pname;}
protected:char *pname;
};
void main()
{person p1("Randy");person p2 = p1;}
```

运行结果: constructing Randy
Destructing Randy
Destructing …
Null pointer assignment

产生错误的原因是,当创建 p2 时,对象 p1 通过默认拷贝构造函数被复制给 p2,但资源并未复制,因此,p1 和 p2 指向同一个资源,因析构顺序与构造顺序相反,析构 p2 时,将堆中字符串清成空串,然后将堆空间返还给系统,当析构 p1 时,因为这时 pname 指向的是空串,所以有第三行输出,当执行"delete pname;"时,系统报错,显示第四行的信息。过程如图 3 所示。

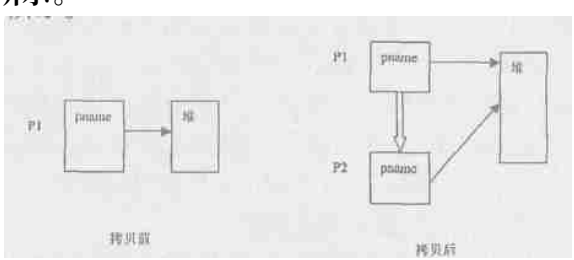

图 3　未提供自拷贝构造函数过程

解决方法:当一个对象创建时,分配了资源,这时,就需要定义自己的拷贝构造函数,使之不但拷贝成员,也拷贝资源。所以在上面的类定义中加入一个自己定义的拷贝构造函数,代码如下:

```
person(person &p)
{ cout << "copying" << p.pname << "into its own block \n";
pname = new char[strlen(p.pname) + 1];
strcpy(pname,p.pname);
}
```

解决后过程详见图 4。

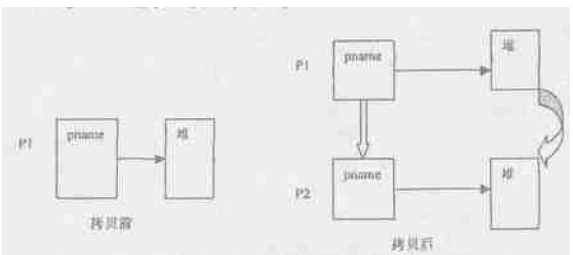

图 4　提供自定义的拷贝构造函数过程

4　结束语

在面向对象程序设计过程中,经常要用到类,而类的数据成员不能在类定义中初始化[4],所以经常要用构造函数来初始化成员,有必要弄清楚构造函数的不同用途,以便更好地运用以达到预期的目的,避免程序设计过程中不必要的错误。其中最重要的是拷贝构造函数很好地解决了对象之间的拷贝和资源冲突的问题,而转换构造函数实现了将基本数据类型变量赋给类类型对象,解决了基本数据类型向类类型转化的难题。

参考文献:

[1]Bjarne Stroustrup. The C++ Programming Language[M]. China Machine Press,2002.

[2]Harvey M deitel, Paul James Deitel. The Complete C++ Training Course(Second Edition)[M]. Publishing House of Electronics Industry,2002.

[3]范辉,刘惊雷,傅铭. Visual C++ 6.0 程序设计简明教程[M]. 北京:高等教育出版社,2001.

[4]钱能. C++程序设计教程[M]. 北京:清华大学出版社,1999.