

xujianxin's CSDN Blog

目录视图

摘要视图

RSS 订阅

个人资料



sendy888

访问： 118479次

积分： 1466

等级： **BLOG > 4**

排名： 第12593名

原创： 39篇 转载： 13篇

译文： 1篇 评论： 21条

[博客Markdown编辑器上线啦](#) [那些年我们追过的Wrox精品红皮计算机图书](#) [PMBOK第五版精讲视频教程](#) [火星敏捷开发1001问](#)

C++成员函数的重载、覆盖与隐藏

分类： C/C++基础知识

2007-08-11 22:24

5078人阅读

[评论\(4\)](#)[收藏](#)[举报](#)

c++

float

behavior

class

编译器

object

成员函数的**重载**、**覆盖 (override)** 与**隐藏**很容易混淆，C++程序员必须要搞清楚概念，否则错误将防不胜防。

成员函数被**重载**的特征：

- (1) 相同的范围（在同一个类中）；
- (2) 函数名字相同；
- (3) 参数不同；
- (4) virtual 关键字可有可无。

覆盖：派生类函数覆盖基类函数，特征是：

- (1) 不同的范围（分别位于派生类与基类）；
- (2) 函数名字相同；

文章搜索

文章分类

[bookaa](#) (10)[C/C++基础知识](#) (8)[English 翻译](#) (0)[MFC Controls](#) (9)[MFC/SDK编程](#) (9)[VC网络编程](#) (4)[Xtreme Toolkit界面库](#) (6)[其它杂乱技术](#) (1)[加密与解密技术](#) (0)[反汇编技术](#) (1)[心情日记](#) (1)[数据库](#) (1)[汇编语言\(包括Win32汇编\)](#) (0)[算法与数据结构](#) (0)[网络转载](#) (1)[面试/测试题收集](#) (2)

文章存档

[2007年12月](#) (3)[2007年11月](#) (2)[2007年10月](#) (1)

(3) 参数相同;

(4) 基类函数必须有virtual 关键字。

函数Base::f(int)与Base::f(float)相互重载, 而Base::g(void)被Derived::g(void)覆盖。

```
#include <iostream.h>
```

```
class Base
```

```
{
```

```
public:
```

```
void f(int x){ cout << "Base::f(int) " << x << endl; }
```

```
void f(float x){ cout << "Base::f(float) " << x << endl; }
```

```
virtual void g(void){ cout << "Base::g(void)" << endl;}
```

```
};
```

```
class Derived : public Base
```

```
{
```

```
public:
```

```
virtual void g(void){ cout << "Derived::g(void)" << endl;}
```

```
};
```

```
void main(void)
```

```
{
```

```
Derived d;
```

```
Base *pb = &d;
```

```
pb->f(42); // Base::f(int) 42
```

```
pb->f(3.14f); // Base::f(float) 3.14
```

```
pb->g(); // Derived::g(void)
```

```
}
```

[2007年08月](#) (22)[2007年07月](#) (25)

阅读排行

[C/C++ 笔试、面试题目大](#)

(19729)

[HTTP断点续传的基本原](#)

(19618)

[RFC2326\(中文版\)－实时](#)

(7217)

[C++函数的高级特性－函](#)

(6861)

[MMS \(Microsoft Media S](#)

(5138)

[C++成员函数的重载、覆](#)

(5075)

[rtsp协议简介](#)

(4227)

[使用cout格式化输出字符](#)

(2953)

[Inno Setup使用教程](#)

(2659)

[几个算法题目](#)

(2618)

评论排行

[C/C++ 笔试、面试题目大](#)

(7)

[C++成员函数的重载、覆](#)

(4)

[几个算法题目](#)

(2)

[RFC2326\(中文版\)－实时](#)

(2)

[MMS \(Microsoft Media S](#)

(2)

[运算符重载](#)

(1)

[C++函数的高级特性－函](#)

(1)

[拖动N个文件,改变文件的](#)

(1)

隐藏：是指派生类的函数屏蔽了与其同名的基类函数，规则如下：

(1) **如果派生类的函数与基类的函数同名，但是参数不同。**此时，不论有无virtual关键字，基类的函数将被隐藏（注意别与重载混淆）。

(2) **如果派生类的函数与基类的函数同名，并且参数也相同，但是基类函数没有virtual关键字。**此时，基类的函数被隐藏（注意别与覆盖混淆）。

(1) 函数Derived::f(float)覆盖了Base::f(float)。

(2) 函数Derived::g(int)隐藏了Base::g(float)，而不是重载。

(3) 函数Derived::h(float)隐藏了Base::h(float)，而不是覆盖。

很多C++程序员没有意识到有“隐藏”这回事。由于认识不够深刻，“隐藏”的发生可谓神出鬼没，常常产生令人迷惑的结果。bp 和dp 指向同一地址，按理说运行结果应该是相同的，可事实并非这样。

```
#include <iostream.h>
```

```
class Base
```

```
{
```

```
public:
```

```
virtual void f(float x){ cout << "Base::f(float) " << x << endl; }
```

```
void g(float x){ cout << "Base::g(float) " << x << endl; }
```

```
void h(float x){ cout << "Base::h(float) " << x << endl; }
```

```
};
```

```
class Derived : public Base
```

```
{
```

```
public:
```

```
virtual void f(float x){ cout << "Derived::f(float) " << x << endl; }
```

```
void g(int x){ cout << "Derived::g(int) " << x << endl; }
```

托盘程序TrayIcon (1)
2007-8-11 (0)

推荐文章

- * CSS变量试玩儿
- * 【Android开发经验】兼容不同的屏幕大小
- * Cocoa Core Competencies_1_Accessibility
- * QtAndroid详解(3): startActivity 实战Android拍照功能
- * PHPer都应该关注的服务端性能问题-听云Server试用笔记

最新评论

C++成员函数的重载、覆盖与隐藏
l631068264: 尼玛，你抄书《高质量c++编程》还抄得一模一样还好意思说原创

C++成员函数的重载、覆盖与隐藏
kelikux: learning.....

C++成员函数的重载、覆盖与隐藏
vb748: 学习了

MMS (Microsoft Media Server p
微wx笑: 修改尝试一下，把
mms://pub1.qmoon.net/911pop?
cccode=cc1325改...

C++成员函数的重载、覆盖与隐藏
hujing1111: 英文注释太蹩脚了吧...

托盘程序TrayIcon
hnyztzh: 请教xtreme 托盘图标

```
void h(float x){ cout << "Derived::h(float) " << x << endl; }  
};
```

```
void main(void)  
{  
    Derived d;  
    Base *pb = &d;  
    Derived *pd = &d;  
    // Good : behavior depends solely on type of the object  
    pb->f(3.14f); // Derived::f(float) 3.14  
    pd->f(3.14f); // Derived::f(float) 3.14  
    // Bad : behavior depends on type of the pointer  
    pb->g(3.14f); // Base::g(float) 3.14  
    pd->g(3.14f); // Derived::g(int) 3 (surprise!)  
    // Bad : behavior depends on type of the pointer  
    pb->h(3.14f); // Base::h(float) 3.14 (surprise!)  
    pd->h(3.14f); // Derived::h(float) 3.14  
}
```

摆脱隐藏

隐藏规则引起了不少麻烦。示例程序中，语句pd->f(10)的本意是想调用函数Base::f(int)，但是Base::f(int)不幸被Derived::f(char *)隐藏了。由于数字10不能被隐式地转化为字符串，所以在编译时出错。

```
class Base  
{  
    public:
```

怎么响应鼠标双击事件，恢复窗口？

[C++函数的高级特性—函数重载](#)
ymy1248227142: 好文章，转载了哈

[C/C++ 笔试、面试题目大汇总](#)
lily7202502: 感觉归纳得挺好的，今天去面试的时候考了好多，可惜么有看啊！现在追悔莫及啊！

[C/C++ 笔试、面试题目大汇总](#)
wangbinyi: 真不错

[C/C++ 笔试、面试题目大汇总](#)
yanggang555:

```
void f(int x);  
};  
  
class Derived : public Base  
{  
public:  
void f(char *str);  
};  
  
void Test(void)  
{  
Derived *pd = new Derived;  
pd->f(10); // error  
}
```

从示例 看来，隐藏规则似乎很愚蠢。但是隐藏规则至少有两个存在的理由：写语句pd->f(10)的人可能真的想调用Derived::f(char *)函数，只是他误将参数写错了。有了隐藏规则，编译器就可以明确指出错误，这未必不是好事。否则，编译器会静悄悄地将错就错，程序员将很难发现这个错误，流下祸根。假如类Derived 有多个基类（多重继承），有时搞不清楚哪些基类定义了函数f。如果没有隐藏规则，那么pd->f(10)可能会调用一个出乎意料的基类函数f。尽管隐藏规则看起来不怎么有道理，但它的确能消灭这些意外。

如果语句pd->f(10)一定要调用函数Base::f(int)，那么将类Derived 修改为如下即可。

```
class Derived : public Base  
{  
public:  
void f(char *str);  
void f(int x) { Base::f(x); }  
};
```

上一篇 C++函数的高级特性－函数重载

下一篇 运算符重载

主题推荐 c++ 程序员 编译器 object 继承

猜你在找

- ACE笔记二ACE_Task框架
- 高质量c++const成员函数
- C++临时对象1
- elf 文件格式探秘程序运行背后的故事
- 图形图像处理－之一任意角度的高质量的快速的图像旋
- LuaPlus官方文档
- 什么情况下要有拷贝构造函数
- 定义自己的operator new和operator delete时注意点
- Visual C++MFC入门教程
- 原创六度拓扑www6dtopcom---超乎想像的人际关系网络

准备好了么？跳吧！

更多职位尽在 CSDN JOB

软件开发工程师（C++）	我要跳槽	C++服务器	我要跳槽
苏州敏行医学信息技术有限公司	6-10K/月	北京乐动卓越科技有限公司	15-20K/月
C++软件开发工程师	我要跳槽	软件工程师（C++/C#）	我要跳槽
天津市努思企业服务有限公司	7-10K/月	武汉海翼科技有限公司	4-6K/月



查看评论

4楼 [l631068264](#) 2014-01-27 12:20发表



尼玛，你抄书《高质量c++编程》还抄得一模一样还好意思说原创

3楼 [kelikux](#) 2013-05-25 17:35发表



learning.....

2楼 [vb748](#) 2013-05-05 21:19发表



学习了

1楼 [hujing1111](#) 2012-09-13 14:04发表



英文注释太蹩脚了吧...

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题 Hadoop AWS 移动游戏 Java Android iOS Swift 智能硬件 Docker OpenStack
VPN Spark ERP IE10 Eclipse CRM JavaScript 数据库 Ubuntu NFC WAP jQuery
BI HTML5 Spring Apache .NET API HTML SDK IIS Fedora XML LBS Unity
Splashtop UML components Windows Mobile Rails QEMU KDE Cassandra CloudStack
FTC coremail OPhone CouchBase 云计算 iOS6 Rackspace Web App SpringSide Maemo
Compuware 大数据 aptech Perl Tornado Ruby Hibernate ThinkPHP HBase Pure Solr

[Angular](#) [Cloud Foundry](#) [Redis](#) [Scala](#) [Django](#) [Bootstrap](#)

[公司简介](#) | [招贤纳士](#) | [广告服务](#) | [银行汇款帐号](#) | [联系方式](#) | [版权声明](#) | [法律顾问](#) | [问题报告](#) | [合作伙伴](#) | [论坛反馈](#)

[网站客服](#)

[杂志客服](#)

[微博客服](#)

webmaster@csdn.net

400-600-2320 | 北京创新乐知信息技术有限公司 版权所有 | 江苏乐知网络技术有限公司 提供商务支持

京 ICP 证 070598 号 | Copyright © 1999-2014, CSDN.NET, All Rights Reserved

