

与君共勉

未曾清贫难成人  
不经打击老天真  
自古英雄出炼狱  
从来富贵入凡尘！

欢迎关注左丞的微博  
个人邮箱：  
morewindows#126.com

个人资料



MoreWindows



访问： 2864721次

积分： 24201

等级： BLOG > ?

排名： 第86名

原创： 155篇 转载： 0篇

译文： 0篇 评论： 4015条

博客专栏



白话经典算法

文章： 17篇

阅读： 676887

秒杀多线程面试题系列

[从零开始掌握iOS8开发技术（Swift版）](#) [那些年我们追过的Wrox精品红皮计算机图书](#) [CSDN学院--学习礼包大派送](#) [CSDN JOB](#)  
[带你坐飞机回家过年](#)

## C++ 类的静态成员详细讲解

分类： [C/C++/C#基础](#)

2011-08-26 18:19

18570人阅读

[评论\(7\)](#)

[收藏](#)

[举报](#)

[c++](#)

[output](#)

[linker](#)

[reference](#)

[class](#)

[function](#)

在C++中，静态成员是属于整个类的而不是某个对象，静态成员变量只存储一份供所有对象共用。所以在所有对象中都可以共享它。使用静态成员变量实现多个对象之间的数据共享不会破坏隐藏的原则，保证了安全性还可以节省内存。

静态成员的定义或声明要加个关键**static**。静态成员可以通过双冒号来使用即<类名>::<静态成员名>。

在C++中类的静态成员变量和静态成员函数是个容易出错的地方，本文先通过几个例子来总结静态成员变量和成员函数使用规则，再给出一个实例来加深印象。希望阅读本文可以使读者对类的静态成员变量和成员函数有更为深刻的认识。

第一个例子，通过类名调用静态成员函数和非静态成员函数

```
[cpp]
01. class Point
02. {
03. public:
04.     void init()
05.     {
06.     }
07.     static void output()
08.     {
09.     }
10. };
11. void main()
```



文章: 15篇  
阅读: 661596



Windows  
C/C++/C# 编程  
文章: 130篇  
阅读: 2628171



C++ STL  
文章: 11篇  
阅读: 245685

阅读排行

- 白话经典算法系列之七 堆 (117828)
- 白话经典算法系列之六 树 (99827)
- 白话经典算法系列之五 串 (98946)
- 【OpenCV入门指南】第 (98313)
- 秒杀多线程第一篇 多线程 (88813)
- 秒杀多线程第二篇 多线程 (87143)
- 白话经典算法系列之一 冒 (83813)
- 秒杀多线程第四篇 一个结 (72499)
- STL系列之十 全排列(百万 (57391)
- 秒杀多线程第三篇 原子操 (51315)

MoreWindows微博

```
12. {  
13.     Point::init();  
14.     Point::output();  
15. }
```

编译出错: error C2352: 'Point::init' : illegal call of non-static member function

结论1: 不能通过类名来调用类的非静态成员函数。

第二个例子, 通过类的对象调用静态成员函数和非静态成员函数

将上例的main()改为:

```
[cpp]  
01. void main()  
02. {  
03.     Point pt;  
04.     pt.init();  
05.     pt.output();  
06. }
```

编译通过。

结论2: 类的对象可以使用静态成员函数和非静态成员函数。

第三个例子, 在类的静态成员函数中使用类的非静态成员

```
[cpp]  
01. #include <stdio.h>  
02. class Point  
03. {  
04. public:  
05.     void init()  
06.     {  
07.     }  
08.     static void output()  
09.     {  
10.         printf("%d\n", m_x);  
11.     }  
12. private:  
13.     int m_x;  
14. };  
15. void main()  
16. {  
17.     Point pt;  
18.     pt.output();  
19. }
```

编译出错: error C2597: illegal reference to data member 'Point::m\_x' in a static member



MoreWindows

粉丝2938人

欢迎各位报名参加2015微软社区大课堂 Community Camp。详情 & 报名地址：<http://t.cn/Rz1um0a> 课堂地点在北京市东城区王府井天伦王朝酒店，免费听课还包午餐。果断行动吧。@微软中国MVP项目组



1月23日 10:50

转发了[月光博客](#) 的微博：  
【红旗文稿：敌对势力千方百计研究“翻墙”技术并推广】《红旗文稿》刊文坦

[登录](#) | [注册](#)

文章分类

- [白话经典算法系列](#) (16)
- [Windows多线程](#) (15)
- [STL 他山之石](#) (11)
- [Windows界面编程](#) (13)
- [MoreWindows工作笔记](#) (12)
- [Windows编程](#) (87)
- [C/C++/C#基础](#) (18)
- [VC6.0及VS2008使用技巧](#) (7)
- [OpenCV入门指南](#) (13)
- [HTML/javascript/PHP](#) (12)
- [Linux编程](#) (1)

评论排行

- [白话经典算法系列之七 并](#) (142)
- [秒杀多线程第六篇 经典归](#) (135)
- [白话经典算法系列之六 树](#) (130)
- [位操作基础篇之位操作全](#) (128)
- [秒杀多线程第五篇 经典归](#) (117)
- [秒杀多线程第三篇 原子操](#) (116)

function

因为静态成员函数属于整个类，在类实例化对象之前就已经分配空间了，而类的非静态成员必须在类实例化对象后才有内存空间，所以这个调用就出错了，就好比没有声明一个变量却提前使用它一样。

结论3：静态成员函数中不能引用非静态成员。

第四个例子，在类的非静态成员函数中使用类的静态成员

```
[cpp]
01. class Point
02. {
03. public:
04.     void init()
05.     {
06.         output();
07.     }
08.     static void output()
09.     {
10.     }
11. };
12. void main()
13. {
14.     Point pt;
15.     pt.output();
16. }
```

编译通过。

结论4：类的非静态成员函数可以调用用静态成员函数，但反之不能。

第五个例子，使用类的静态成员变量

```
[cpp]
01. #include <stdio.h>
02. class Point
03. {
04. public:
05.     Point()
06.     {
07.         m_nPointCount++;
08.     }
09.     ~Point()
10.     {
11.         m_nPointCount--;
12.     }
13.     static void output()
14.     {
```

【白话经典算法系列之十】 (115)

STL系列之十 全排列(百月) (109)

秒杀多线程第十篇 生产者 (109)

白话经典算法系列之五 非 (107)

文章搜索

最新评论

【OpenCV入门指南】第十篇 彩张欣-数字图像处理: 如楼上，会导致色彩失真啊

秒杀多线程第十篇 生产者消费者MRain\_: 感谢楼主的多线程解析真的是非常好

进程通信之二 管道技术第二篇 匿PoorGeek: WaitForSingleObject没必多大要吧？

【白话经典算法系列之十三】随hellofuturecyj: set不是本来就是有序的么……

【OpenCV入门指南】第二篇 缩paresly: 楼主，这段代码有内存泄漏，你找到原因了吗，我昨天按你说的，发现有这个问题。调试中

【白话经典算法系列之十二】数!iicup: 确实很有趣,也很巧妙.不过这种条件在实际中基本不会用到,真要遇到这种面试题, 直接BS.

进程通信之二 管道技术第二篇 匿mike\_1990: 管道和共享内存有何不同，博主能介绍下吗。

秒杀多线程第七篇 经典线程同步xinpo66: 楼主写的很好，这个系列让我对线程入门了。最后的那个遗弃的处理，有些时候是我们忘记ReleaseMut...

位操作基础篇之位操作全面总结清qing: 楼主大牛，写的很好，非常感谢

热门智力题 过桥问题和倒水问题daeba: 我只思考出：回来的时候一定是过了桥的人之中最快的那个回来

```
15.         printf("%d\n", m_nPointCount);
16.     }
17.     private:
18.         static int m_nPointCount;
19.     };
20.     void main()
21.     {
22.         Point pt;
23.         pt.output();
24.     }
```

按Ctrl+F7编译无错误，按F7生成EXE程序时报链接错误

error LNK2001: unresolved external symbol "private: static int Point::m\_nPointCount" (?m\_nPointCount@Point@@@0HA)

这是因为类的静态成员变量在使用前必须先初始化。

在main()函数前加上int Point::m\_nPointCount = 0;

再编译链接无错误，运行程序将输出1。

结论5：类的静态成员变量必须先初始化再使用。

结合上面的五个例子，对类的静态成员变量和成员函数作个总结：

一。静态成员函数中不能调用非静态成员。

二。非静态成员函数中可以调用静态成员。因为静态成员属于类本身，在类的对象产生之前就已经存在了，所以在非静态成员函数中是可以调用静态成员的。

三。静态成员变量使用前必须先初始化(如int MyClass::m\_nNumber = 0;)，否则会在linker时出错。

再给一个利用类的静态成员变量和函数的例子以加深理解，这个例子建立一个学生类，每个学生类的对象将组成一个双向链表，用一个静态成员变量记录这个双向链表的表头，一个静态成员函数输出这个双向链表。

```
[cpp]

01. #include <stdio.h>
02. #include <string.h>
03. const int MAX_NAME_SIZE = 30;
04.
05. class Student
06. {
07. public:
08.     Student(char *pszName);
09.     ~Student();
```

```

10.     public:
11.         static void PrintfAllStudents();
12.     private:
13.         char    m_name[MAX_NAME_SIZE];
14.         Student *next;
15.         Student *prev;
16.         static Student *m_head;
17.     };
18.
19. Student::Student(char *pszName)
20. {
21.     strcpy(this->m_name, pszName);
22.
23.     //建立双向链表，新数据从链表头部插入。
24.     this->next = m_head;
25.     this->prev = NULL;
26.     if (m_head != NULL)
27.         m_head->prev = this;
28.     m_head = this;
29. }
30.
31. Student::~~Student ()//析构过程就是节点的脱离过程
32. {
33.     if (this == m_head) //该节点就是头节点。
34.     {
35.         m_head = this->next;
36.     }
37.     else
38.     {
39.         this->prev->next = this->next;
40.         this->next->prev = this->prev;
41.     }
42. }
43.
44. void Student::PrintfAllStudents()
45. {
46.     for (Student *p = m_head; p != NULL; p = p->next)
47.         printf("%s\n", p->m_name);
48. }
49.
50. Student* Student::m_head = NULL;
51.
52. void main()
53. {
54.     Student studentA("AAA");
55.     Student studentB("BBB");
56.     Student studentC("CCC");
57.     Student studentD("DDD");
58.     Student student("MoreWindows");
59.     Student::PrintfAllStudents();
60. }

```

程序将输出：

```
MoreWindows
DDD
CCC
BBB
AAA
Press any key to continue
```

当然在本例还可以增加个静态成员变量来表示链表中学生个数，如果读者有兴趣，就将这个作为小练习吧。

转载请标明出处，原文地址：<http://blog.csdn.net/morewindows/article/details/6721430>

上一篇 白话经典算法系列之七 堆与堆排序

下一篇 VC 编译参数介绍

顶 23

踩 0

主题推荐

c++ 类 链表 实例 function

猜你在找

|   |                                 |
|---|---------------------------------|
| C++学习之深入理解虚函数一虚函数表解析                      | string的size和length              |
| 类静态指针的申请和释放                               | 学习OpenCV滤镜系列3颜色变幻               |
| Qt树形控件QTreeView使用2复选框的设置                  | GDB 查看死锁                        |
| Cocos2d-x 30开发六使用cocoStudio创建一个骨骼动画       | vim显示行号语法高亮自动缩进的设置              |
| mfc中自动生成的CViewOnLButtonDownnFlags point代码 | cocos2d-x-33rc2-001-hello-world |

准备好了么？跳吧！

更多职位尽在 CSDN JOB


|                |          |                           |          |
|----------------|----------|---------------------------|----------|
| C++服务端开发工程师    | 我要跳槽     | C++开发工程师                  | 我要跳槽     |
| 欢聚时代（多玩YY）     | 面议       | 浙江大华技术股份有限公司              | 10-15K/月 |
| C++工程师         | 我要跳槽     | C++ / C#程序员(虚拟仿真平台系统软...) | 我要跳槽     |
| 北京和勤联创科技发展有限公司 | 12-24K/月 | 北京东方仿真软件按技术有限公司           |          |



|                          |                          |                           |                              |                            |
|--------------------------|--------------------------|---------------------------|------------------------------|----------------------------|
| <a href="#">1 启动器下载</a>  | <a href="#">5 语音广告制作</a> | <a href="#">9 个人主页模板</a>  | <a href="#">13 三维动画设计</a>    | <a href="#">17 代办工作居住证</a> |
| <a href="#">2 中维监控系统</a> | <a href="#">6 网银下载</a>   | <a href="#">10 苹果id密码</a> | <a href="#">14 网页制作下载</a>    | <a href="#">18 游戏制作软件</a>  |
| <a href="#">3 手机游戏制作</a> | <a href="#">7 led控制卡</a> | <a href="#">11 手写板下载</a>  | <a href="#">15 网银控件下载</a>    | <a href="#">19 数据恢复免费版</a> |
| <a href="#">4 老a电商学院</a> | <a href="#">8 流量表</a>    | <a href="#">12 sd卡修复器</a> | <a href="#">16 java架构师认证</a> | <a href="#">20 测试用例设计</a>  |

查看评论

5楼 [qyg1998](#) 2014-09-08 05:20发表



不知道为什么发不出去。第一次回复，哈哈。  
楼主写的非常好，对我有很多启发。  
除此之外，提个小瑕疵，就是析构函数还可以再严谨些。

```
01.         [cpp]
02.         Student::~Student ()//析构过程就是节点的脱离过程
03.         {
```

更多相关资源：[c](#)

```
05.             Student *ptmp = m_head;
06.             m_head = this->next;
07.             cout << "in destructor head " << endl;
08.         }
09.             else if(this->next == NULL)
10.             { this->pre->next =NULL ;}
11.             else {
12.                 this->prev->next = this->next;
13.                 this->next->prev = this->prev;
14.             }
15.     }
```

我增加了

```
else if(this->next == NULL)
{ this->pre->next =NULL ;}
```

否则 this->next->prev 将有问题，因为 this->next 都NULL了 还怎么能找prev域呢？

测试： new \*p1 = ...

new \*p2 = ...

new \*p3 = ...


//单独 delete p3 is fine

//单独 delete p2 is fine

单独 but delete p1 is wrong //这是最后一个，因为你是前插


修改后的析构函数没有这类问题了。均能正常调用析构函数析构。

4楼 [Nestler](#) 2014-03-10 15:40发表



例3: Point pt;  
这不算实例化对象吗？

Re: [Nestler](#) 2014-03-10 15:43发表



回复jy02326166：知道怎么回事了。建议把

```
void main()
{
    Point pt;
    pt.output();
}
```

里面的内容删掉，更能说明问题。

3楼 [CaptianSlow](#) 2014-02-16 21:34发表



`static`成员变量的定义必须在类外进行，并且一定并且永远不要依赖没有定义的变量

2楼 [lixiaojun9688](#) 2012-06-15 11:02发表



嗯不错，只是双向链表的构成有些难理解啊，不过还是想通了

Re: [MoreWindows](#) 2012-06-15 15:01发表



回复lixiaojun9688：STL中的list就是双向链表，有兴趣可以看下。

1楼 [Unix\\_Architect](#) 2012-05-16 18:26发表



DECLARE\_DYNMIC  
DECLARE\_CREATION  
DECLARE\_MESSAGE\_MAPPING

MFC那个机制，可以说是把静态变量用到了出神入化。

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

