

通过汇编代码理解成员函数指针并不是指针 PDF转换可能丢失图片或格式，建议阅读原文

https://www.100test.com/kao_ti2020/460/2021_2022__E9_80_9A_E8_BF_87_E6_B1_87_E7_c98_460695.htm

前言：在CSDN论坛经常会看到一些关于类成员函数指针的问题，起初我并不在意，以为成员函数指针和普通的函数指针是一样的，没有什么太多需要讨论的。当我找来相关书籍查阅了一番以后，突然意识到我以前对成员函数指针的理解太过于幼稚和肤浅了，它即不像我以前认为的那样简单，它也不像我以前认为的那样“默默无闻”。强烈的求知欲促使我对成员函数进行进一步

的学习并有了这篇文章。一。理论篇 在进行深入学习和分析之前，还是先看看书中是怎么介绍成员函数的。总结一下类成员函数指针的内容，应该包含以下几个知识点：1。成员函数指针并不是普通的函数指针。2。编译器提供了几个新的操作符来支持成员函数指针操作：1) 操作符“::*”用来声明一个类成员函数指针，例如：`typedef void`

`(Base::*PVVBASEMEMFUNC)(void).` //Base is a class 2) 操作符“->*”用来通过对象指针调用类成员函数指针，例如：

`//pBase is a Base pointer and well initialized //pVIBaseMemFunc is a member function pointer and well initialized`

`(pBase->*pVIBaseMemFunc)().` 3) 操作符“.”用来通过对象调用类成员函数指针，例如：`//baseObj is a Base object`

`//pVIBaseMemFunc is a member function pointer and well initialized (baseObj.*pVIBaseMemFunc)().` 3。成员函数指针是强类型的。

`typedef void (Base::*PVVBASEMEMFUNC)(void).
typedef void (Derived::*PVVDERIVEMEMFUNC)(void).`

PVVBASEMEMFUNC和PVVDERIVEMEMFUNC是两个不同类型的成员函数指针类型。4。由于成员函数指针并不是真真正正意义上的指针，所以成员函数指针的转化就受限制。具体的转化细节依赖于不同的编译器，甚至是同一个编译器的不同版本。不过，处于同一个继承链中的不同类之间override的不同函数和虚函数还是可以转化的。

```
void* pVoid = reinterpret_cast(pVIBaseMemFunc). //error
int* pInt = reinterpret_cast(pVIBaseMemFunc). //error
pVIDeriveMemFunc = static_cast(pVIBaseMemFunc). //OK
```

二。实践篇 有了上面的理论知识，我们对类成员函数指针有了大概的了解，但是我们对成员函数指针还存在太多的疑惑。既然说成员函数指针不是指针，那它到底是什么东东？编译器为什么要限制成员函数指针转化？老办法，我们还是分析汇编代码揭示其中的秘密。

首先，我写了这样两个具有继承关系的类：

```
class Base {
public: //ordinary member function void setValue(int iValue).
//virtual member function virtual void dumpMe(). virtual void
foobar(). protected: int m_iValue. }. class Derived:public Base{
public: //ordinary member function void setValue(int iValue).
//virtual member function virtual void dumpMe(). virtual void
foobar(). private: double m_fValue. }. 接着，我又定义了一些成员函数指针类型：
```

```
typedef void (Base::*PVVBASEMEMFUNC)(void). typedef void
(Derived::*PVVDERIVEMEMFUNC)(void). typedef void
(Base::*PVIBASEMEMFUNC)(int). typedef void
(Derived::*PVIDERIVEMEMFUNC)(int).
```

最后，在main函数写了一些测试代码：

```
int _tmain(int argc, _TCHAR* argv[]) {
```

```
PVIBASEMEMFUNC pVIBaseMemFunc = amp.Base::foobar.  
PVVDERIVEMEMFUNC pVVDeriveMemFunc =  
static_cast(pVVBaseMemFunc). Base baseObj.  
(baseObj.*pVIBaseMemFunc)(10).  
(baseObj.*pVVBaseMemFunc)(). Derived deriveObj.  
(deriveObj.*pVIDeriveMemFunc)(20).  
(deriveObj.*pVVDeriveMemFunc)(). return 0. } 100Test 下载频道  
开通，各类考试题目直接下载。详细请访问 www.100test.com
```