**一、静态库与动态库的区别**

　　目前以lib后缀的库有两种，一种为静态链接库(Static Libary，以下简称“静态库”)，另一种为动态连接库(DLL，以下简称“动态库”)的导入库(Import Libary，以下简称“导入库”）。

　　静态库是一个或者多个obj文件的打包，所以有人干脆把从obj文件生成lib的过程称为Archive，即合并到一起。比如你链接一个静态库，如果其中有错，它会准确的找到是哪个obj有错，即静态lib只是壳子。

　　动态库一般会有对应的导入库，方便程序静态载入动态链接库，否则你可能就需要自己LoadLibary调入DLL文件，然后再手工GetProcAddress获得对应函数了。有了导入库，你只需要链接导入库后按照头文件函数接口的声明调用函数就可以了。

　　导入库和静态库的区别很大，他们实质是不一样的东西。静态库本身就包含了实际执行代码、符号表等等，而对于导入库而言，其实际的执行代码位于动态库中，导入库只包含了地址符号表等，确保程序找到对应函数的一些基本地址信息。

　　动态链接库 (DLL) 是作为共享函数库的可执行文件。动态链接提供了一种方法，使进程可以调用不属于其可执行代码的函数。函数的可执行代码位于一个 DLL 中，该 DLL 包含一个或多个已被编译、链接并与使用它们的进程分开存储的函数。DLL 还有助于共享数据和资源。多个应用程序可同时访问内存中单个 DLL 副本的内容。

　　动态链接与静态链接的不同之处在于它允许可执行模块（.dll 文件或 .exe 文件）仅包含在运行时定位 DLL 函数的可执行代码所需的信息。在静态链接中，链接器从静态链接库获取所有被引用的函数，并将库同代码一起放到可执行文件中。 使用动态链接代替静态链接有若干优点。DLL 节省内存，减少交换操作，节省磁盘空间，更易于升级，提供售后支持，提供扩展 MFC 库类的机制，支持多语言程序，并使国际版本的创建轻松完成。

**很多开源代码发布的惯用方式：**

　　1. 预编译的开发包：包含一些.dll文件和一些.lib文件。其中这里的.lib就是导入库，而不要错以为是静态库。但是引入方式和静态库一样，要在链接路径上添加找到这些.lib的路径。而.dll则最好放到最后产生的应用程序exe执行文件相同的目录。这样运行时，就会自动调入动态链接库。

　　2. 用户自己编译： 下载的是源代码，按照readme自己编译。生成很可能也是.dll + .lib(导入库)的库文件

　　3. 如果你只有dll，并且你知道dll中函数的函数原型，那么你可以直接在自己程序中使用LoadLibary调入DLL文件，GetProcAddress

**二、静态链接库**

　　静态链接库的创建和使用都比较的简单，我们直接看个理解就可以了。

　　在vs2010中新建一个Win32控制台的静态链接库工程static\_lib，在工程中创建文件lib.h和lib.cpp这两个文件，lib.h和lib.cpp的源代码如下：

[复制代码](javascript:void(0);)

//lib.h文件

#ifndef LIB\_H

#define LIB\_H

extern "C" int add(int a, int b); //声明为C调用方式的外部调用函数

#endif

//lib.cpp文件

#include "lib.h"

int add(int a, int b)

{

return a + b;

}

[复制代码](javascript:void(0);)

　　直接编译这个工程，在debug目标中就会自动生成static\_lib.lib。在static\_lib工程的工作区中新建libcall工程，它只有一个文件test.cpp，代码如下：

[复制代码](javascript:void(0);)

#include "../static\_lib/lib.h"

#include <stdio.h>

#pragma comment(lib, "..\\debug\\static\_lib.lib")

int main(int argc, char \*argv[])

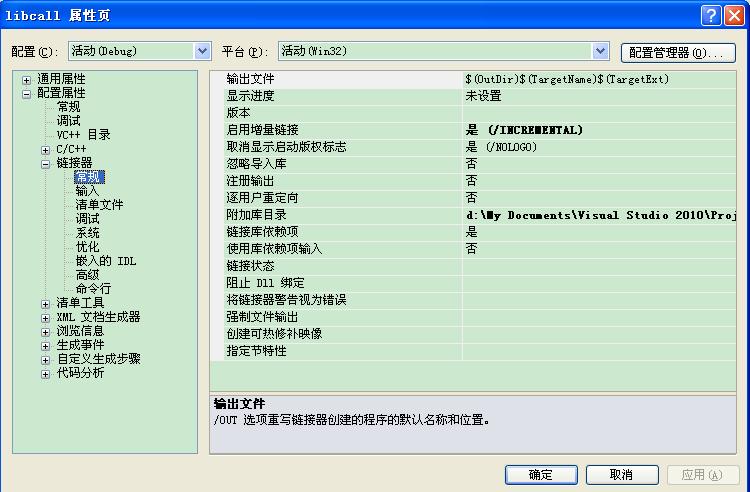
{

printf("4 + 5 = %d\n", add(4, 5));

}

[复制代码](javascript:void(0);)

　　静态链接库的调用就是这么简单，或许我们每天都在用，可是我们没有明白这个概念。代码中#pragma comment(lib, "..\\debug\\static\_lib.lib")的意思是指本文件生成的.obj文件应与tatic\_lib.lib一起链接。如果不用#pragma comment指定，则可以直接在vs中设置，如图1，在libcall工程的属性页面，通过链接器->常规->附加库目录添加tatic\_lib.lib所在的目录，通过链接器->输入->附加依赖项添加static\_lib.lib到项目中。



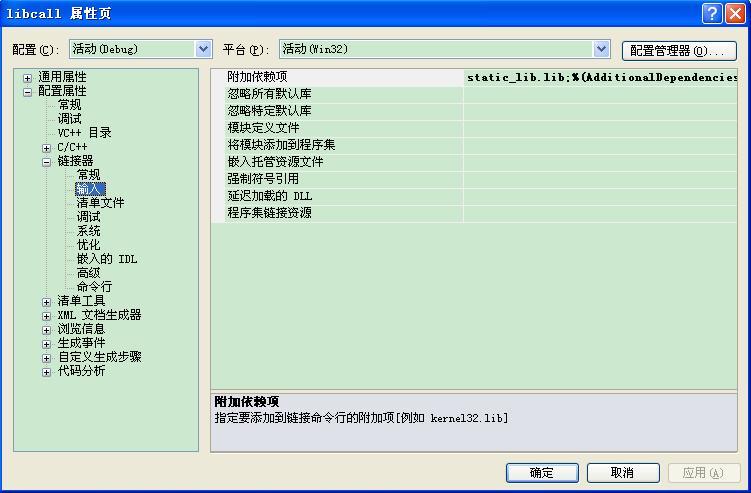


图1

　　直接运行就可以看到屏幕上输出结果。

　　现在我们来看看static\_lib.lib的反汇编结果：

[复制代码](javascript:void(0);)

Microsoft (R) COFF/PE Dumper Version 10.00.30319.01

Copyright (C) Microsoft Corporation. All rights reserved.

Dump of file D:\My Documents\Visual Studio 2010\Projects\static\_lib\Debug\static\_lib.lib

File Type: LIBRARY

\_add:

00000000: 55 push ebp

00000001: 8B EC mov ebp,esp

00000003: 81 EC C0 00 00 00 sub esp,0C0h

00000009: 53 push ebx

0000000A: 56 push esi

0000000B: 57 push edi

0000000C: 8D BD 40 FF FF FF lea edi,[ebp+FFFFFF40h]

00000012: B9 30 00 00 00 mov ecx,30h

00000017: B8 CC CC CC CC mov eax,0CCCCCCCCh

0000001C: F3 AB rep stos dword ptr es:[edi]

0000001E: 8B 45 08 mov eax,dword ptr [ebp+8]

00000021: 03 45 0C add eax,dword ptr [ebp+0Ch]

00000024: 5F pop edi

00000025: 5E pop esi

00000026: 5B pop ebx

00000027: 8B E5 mov esp,ebp

00000029: 5D pop ebp

0000002A: C3 ret

Summary

5FC .debug$S

70 .debug$T

41 .drectve

4 .rtc$IMZ

4 .rtc$TMZ

2B .text

[复制代码](javascript:void(0);)

　　从上面的汇编代码可以看到static\_lib.lib直接包含了add的函数代码，这也应证了前面说的代码直接放在静态库的lib文件中。