**导致“无法解析的外部符号”的可能情况**

2016年10月17日 16:28:43

阅读数：19894

之前写代码时一直有遇到过这种问题:

无法解析的外部符号"XXX::XXXX"，该符号在函数"XXX::XXXX" 中被引用......

其大部分都是由于

1.只声明了函数但并没有实现它。

这种一般都是大意忘记了,第一次编译就会发现。

另外声明了静态变量，没有初始化它也会导致这样的情况发生。

2.声明了函数也定义了函数但其函数参数对不上。

比如项目A中引用了项目B的类,后来修改了项目B中类的成员函数等但A中没改过,导致编译的时候出错。

3.未在被引用的函数或类中加\_\_declspec(dllexport)和\_\_declspec(dllimport)修饰符。

想要让函数在其他项目中被引用就要在源文件处加上\_\_declspec(dllexport),在引用处加上\_declspec(dllimport);

一般的写法是

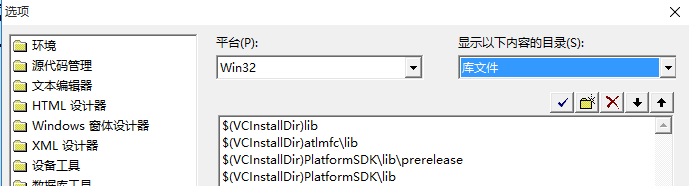
**[cpp]** [view plain](https://blog.csdn.net/vis_stu/article/details/52839438) [copy](https://blog.csdn.net/vis_stu/article/details/52839438)

1. //B项目中:
2. //一般在stdafx.h中定义PRJ\_B
3. //在导出的XXXH.h中写入对宏定义的判断
4. #ifdefine PRJ\_B
5. #define BDLLEXPIMPORT \_\_declspec(dllexport)
6. #else
7. #define BDLLEXPIMPORT \_\_declspec(dllimport)
8. #endif
9. //然后在类中使用
10. //比如b.h头文件中
11. **class** BDLLEXPIMPORT b{
12. //....
13. };

于是,在B项目中因为定义了PRJ\_B所以其为导出模式,而在A项目中因为没有定义过PRJ\_B所以其为导入模式;

4.项目中包含类(lib,dll)的设置路径不正确。

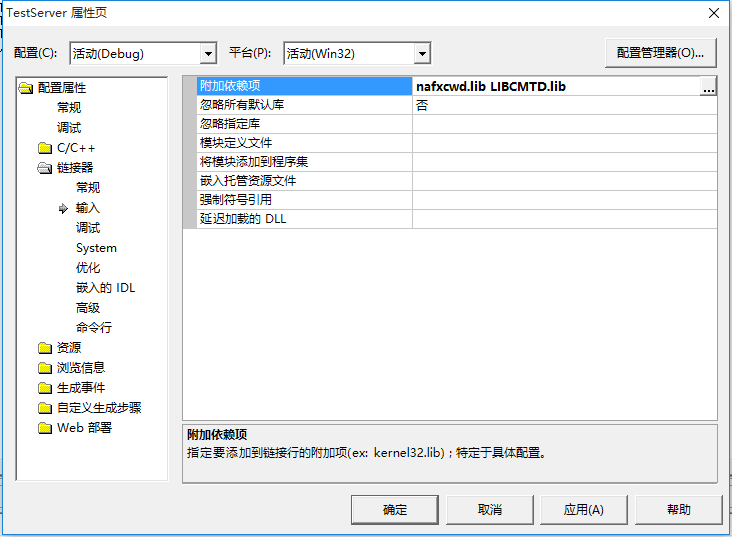
检查工程中路径的设置是否有误。



5.确定需要的库都正确指定。

一种方式是通过#pragma comment(lib,"../XXX.lib")方式指定

一种是在工程属性中设置附加依赖项：



6.编译模式方式不同 , (比如win32位下编译和64位下编译)也有可能造成编译不过的现象。

https://img-blog.csdn.net/20161027150840694?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQv/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center

7.debug或release的编译方式。

之前在本地编译一个关于sqlite3.c的项目时, 总提醒无法解析的外部符号\_ReadWriteBarrier，但服务器端一直有在编译，应该不会编不过。msdn上查了一下，应该是要包含<intrin.h>头文件，而后发现原文件中include <intrin.h>这句被人注释掉了 , 原因是包含这个文件vs2002环境下会编译不过。

那服务器端在不包含该头文件的情况下为什么能编译过呢? 原因是服务器上编译是release版本的。而本地调整成release版本后也能进行编译了。

结论是release和debug版本使用的库可能存在区别，这个请自行确定使用release版本编译或者重新找一个debug的库。

f