问题前导：之前跟大家讨论过关于DLL和EXE中堆内存的问题。问题就是，如果一个EXE调用一个DLL时，用new和delete分配和释放内存为什么应该放在同一个背景下的原因。得出的结论是，如果EXE和DLL有一个不是用动态链接CRT库(C runtime library)的方式使用CRT的话，或者是EXE和DLL动态链接的CRT库的版本不同时，EXE和DLL将会各自拥有各自的堆空间，所以在DLL中new的东西务必在DLL中delete。  
(对以前的讨论有问题的朋友可以参考我以前发的提问贴，搜 "DLL 内存")。  
  
//////////////////////////////////////////////////////////////////////////  
  
新问题描述：(环境：VC2005)  
  
某workspace下有子工程A、B、C三个，其中，A是静态库(\*.lib)，B是动态链接库(\*.dll)， C是应用程序(\*.exe)。  
他们的关系是，B(DLL)包含了A(LIB)，C(EXE)包含了A(LIB)，同时C(EXE)使用了B(DLL)。  
  
若其中A(LIB)中有代码如下：  
class CSampleA  
{  
public:  
    CSampleA(){}  
    ~CSampleA(){}  
    DestroySelf(){delete this;}  //标记1  
    //...other member function...  
};  
  
class CFactory  
{  
public:  
    static CSampleA\*  GetInstanceOfA(){return new CSampleA();}  
};  
  
  
其中B(DLL)中有代码如下：  
class CProxyA  
{  
public:  
    CProxyA(){m\_pA = CFactory::GetInstanceOfA();}  
    ~CProxyA(){if(m\_pA)m\_pA->DestroySelf();}  
    DestroySelf(){delete this;}  
    CSampleA\*  GetA(){return m\_pA;}  
public:  
    CSampleA\*  m\_pA;  
};  
  
class CProxyFactory  
{  
public:  
    static CProxyA\*  GetProxyInstanceOfA(){return new CProxyA();}  
};  
  
  
其中C(EXE)中有代码如下：  
CProxyA\* pProxyA = ...; //...use CProxyFactory get CProxyA's instance  
CSampleA\* pA = pProxyA->GetA();  
//...  
//...call some CSampleA's functions...  
//...  
pProxyA->DestroySelf();  //release memory, then, "标注1" will have problem.  
  
  
Problem:  
dbgheap.c,  1252 line, \_CrtIsValidHeapPointer()断言失败。  
  
不知道我的问题描述清楚了没有。。。  
  
//////////////////////////////////////////////////////////////////////////  
问题猜想：  
我感觉问题出在C和B同时都包含了A上。C和B同时有自己的堆内存空间，A分别静态链接到C和B中，那么C和B中分别包含一份A的代码段，这时，通过B的堆内存创建出来的CSampleA类，在C中调用起成员函数，是不是有问题？但为何是释放时才出问题？  
  
请高手指点！  
PS：基础不扎实或者概念比较模糊的朋友就不要误人子弟了，谢谢合作……