LPVOID VirtualAllocEx(

HANDLE hProcess, //申请内存所在的进程句柄。

LPVOID lpAddress, //保留页面的内存地址;一般用NULL自动分配

SIZE\_T dwSize, //欲分配的内存大小，字节单位;注意实际分 配的内存大小是页内存大小的整数倍

DWORD flAllocationType, //

DWORD flProtect

);

flAllocationType 可取下列值:

****MEM\_COMMIT:为特定的页面区域分配内存中或磁盘的**[页面文件](http://baike.so.com/doc/5649710-5862353.html" \t "http://baike.so.com/doc/_blank)**中的物理存储****

MEM\_PHYSICAL :分配[物理内存](http://baike.so.com/doc/5677263-5889936.html" \t "http://baike.so.com/doc/_blank)(仅用于地址窗口[扩展内存](http://baike.so.com/doc/6282676-6496135.html" \t "http://baike.so.com/doc/_blank))

MEM\_RESERVE:保留进程的虚拟地址空间，而不分配任何物理存储。保留页面可通过继续调用VirtualAlloc()而被占用

MEM\_RESET :指明在内存中由参数lpAddress和dwSize指定的数据无效

MEM\_TOP\_DOWN:在尽可能高的地址上分配内存(Windows 98忽略此标志)

MEM\_WRITE\_WATCH:必须与MEM\_RESERVE一起指定，使系统跟踪那些被写入分配区域的页面(仅针对Windows 98)

flProtect可取下列值:

PAGE\_READONLY: 该区域为只读。如果应用程序试图访问区域中的页的时候，将会被拒绝访

PAGE\_READWRITE 区域可被应用程序读写

PAGE\_EXECUTE: 区域包含可被系统执行的代码。试图读写该区域的操作将被拒绝。

PAGE\_EXECUTE\_READ :区域包含可执行代码，应用程序可以读该区域。

PAGE\_EXECUTE\_READWRITE: 区域包含可执行代码，应用程序可以读写该区域。

PAGE\_GUARD: 区域第一次被访问时进入一个STATUS\_GUARD\_PAGE异常，这个标志要和其他保护标志合并使用，表明区域被第一次访问的权限

PAGE\_NOACCESS: 任何访问该区域的操作将被拒绝

PAGE\_NOCACHE: RAM中的页映射到该区域时将不会被微处理器缓存(cached)

返回值:

执行成功就返回分配内存的首地址，不成功就是NULL。

例

wchar\_t \*pszLibFileRemote;

//在远程进程的内存地址空间分配DLL文件名缓冲区

pszLibFileRemote = (wchar\_t \*)VirtualAllocEx(hRemoteProcess, NULL, lstrlen(DllFullPath) + 1,

MEM\_COMMIT, PAGE\_READWRITE);

if (pszLibFileRemote == NULL)

{

MessageBox(NULL, L"在远程进程的内存地址空间分配DLL文件名缓冲区失败! ", L"Error", MB\_OK);

return FALSE;

}