new / delete    
[malloc](https://so.csdn.net/so/search?q=malloc&spm=1001.2101.3001.7020) / free     
CoTaskMemAlloc / CoTaskMemFree     
IMalloc::alloc / IMalloc/free     
GlobalAlloc / GlobalFree     
LocalAlloc / LocalFree     
HeapAlloc / HeapFree     
VirtualAlloc / VirtualFree

VirtualAlloc以页面为单位(4K)进行分配，是操纵虚拟[内存](https://so.csdn.net/so/search?q=%E5%86%85%E5%AD%98&spm=1001.2101.3001.7020)的底层函数。

HeapAlloc在指定堆上分配内存。    
Windows为每个进程设置了一个缺省堆，应用程序也可以用HeapCreate创建更多的堆。

GlobalAlloc和LocalAlloc原本为16位环境下的API，Win32为了保持兼容性而保留了这两个函数，但在Win32下已不存在全局堆(所有的堆都是进程私有)，所以 GlobalAlloc 和 LocalAlloc 在Win32下意义完全相同，它们都在进程缺省堆中进行内存分配。需注意的是，在Win32下，GlobalAlloc的字面意思已失效，它并不能在进程间共享数据。微软强调，GlobalAlloc/LocalAlloc 比 HeapAlloc 要慢，已不再推荐使用，但由于GlobalAlloc/LocalAlloc 具有简单的使用接口，所以即使在微软所提供的源码中，它们仍被大量使用。

malloc是CRT函数，实现方式取决于具体的CRT版本。VC++的malloc系对 HeapAlloc 作简单的包装，而Borland C++则选择自己实现malloc。在应用程序启动时，CRT创建自己的私有堆，驻留在Win32堆的顶部。

C++中的new先调用 operator new，再调用构造函数生成类对象。而 operator new 在缺省情况下调用 malloc 进行内存分配。应用程序可以重载 operator new，选择其他的内存分配方式。

CoTaskMemAlloc用于COM对象，它在进程的缺省堆中分配内存。    
IMalloc接口是对 CoTaskMemAlloc/CoTaskMemFree 的再次封装。

调用关系：    
msvcrt.malloc                 => kernel32.HeapAlloc(ntdll.RtlAllocateHeap)    
kernel32.LocalAlloc           => ntdll.RtlAllocateHeap    
kernel32.GlobleAlloc          => ntdll.RtlAllocateHeap    
kernel32.HeapAlloc            == ntdll.RtlAllocateHeap (映射)     
kernel32.VirtualAlloc         => kernel32.VirtualAllocEx    
kernel32.VirtualAllocEx       => ntdll.NtAllocateVirtualMemory    
ntdll.RtlAllocateHeap         => ntdll.NtAllocateVirtualMemory    
ntdll.NtAllocateVirtualMemory => ntdll.KiFastSystemCall    
ntdll.KiFastSystemCall        => sysenter指令 (0F34)      
即:     
new -> malloc -> HeapAlloc -> VirtualAlloc -> 驱动程序的\_PageAlloc

关于什么时候使用CoTaskMemAlloc/CoTaskMemFree来申请/释放内存

从网上看来的说法：  
The thing is that COM (actually the RPC runtime library, but it's easier to blame COM) has an additional requirement for [out] pointers.  This requirement is that if the type of an [out] parameter isn't a scalar quantity (in other words if it's a structure or anything more complicated than a int or float), then the memory pointed to by the [out] parameter needs to be allocated either by MIDL\_user\_allocate (for RPC) or CoTaskMemAlloc (for COM).

也就是说，如果需要调用COM接口，并且接口中有[out]的参数，且类型不是整形或浮点数之类的基本数据类型，就需要调用CoTaskMemAlloc来申请这块内存，并使用完毕后调用CoTaskMemFree来释放。