# 学习大纲

|  |
| --- |
|  |

## 所谓的 虚拟内存,其实就是电脑硬盘上面的一个文件,这个文件非常大,window把它当作内存来使用,他可以在一定程度上解决电脑内存不足的问题.例如,一台32为的window电脑的所有内存地址空间是4G,假如此时它的物理内存注意2G,windows就会在磁盘上面创建一个2G大小的虚拟内存文件来当作内存使用.

## MEMORYSTATUS结构体

|  |  |  |
| --- | --- | --- |
| 包含有关物理内存和虚拟内存的当前状态的信息。 [GlobalMemoryStatus](https://learn.microsoft.com/zh-cn/windows/desktop/api/winbase/nf-winbase-globalmemorystatus) 函数将信息存储在 **MEMORYSTATUS** 结构中。 语法 C++复制  typedef struct \_MEMORYSTATUS {  DWORD dwLength;  DWORD dwMemoryLoad;  SIZE\_T dwTotalPhys;  SIZE\_T dwAvailPhys;  SIZE\_T dwTotalPageFile;  SIZE\_T dwAvailPageFile;  SIZE\_T dwTotalVirtual;  SIZE\_T dwAvailVirtual;  } MEMORYSTATUS, \*LPMEMORYSTATUS; 成员 dwLength  **MEMORYSTATUS** 数据结构的大小（以字节为单位）。 在调用 [GlobalMemoryStatus](https://learn.microsoft.com/zh-cn/windows/desktop/api/winbase/nf-winbase-globalmemorystatus) 函数之前，无需设置此成员;函数设置它。  dwMemoryLoad  一个介于 0 和 100 之间的数字，指定正在使用的物理内存的近似百分比 (0 表示不使用内存，100 表示) 已满内存使用。  dwTotalPhys  实际物理内存量（以字节为单位）。  dwAvailPhys  当前可用的物理内存量（以字节为单位）。 这是可以立即重复使用的物理内存量，而无需先将其内容写入磁盘。 它是备用列表、可用列表和零列表的大小之和。  dwTotalPageFile  已提交的内存限制的当前大小（以字节为单位）。 这是物理内存加上页面文件的大小，减去少量开销。  dwAvailPageFile  当前进程可以提交的最大内存量（以字节为单位）。 此值应小于系统范围的可用提交。 若要计算此值，请调用 [GetPerformanceInfo](https://learn.microsoft.com/zh-cn/windows/desktop/api/psapi/nf-psapi-getperformanceinfo) 并从 **CommitLimit** 中减去 **CommitTotal** 的值。  dwTotalVirtual  调用进程的虚拟地址空间的用户模式部分的大小（以字节为单位）。 此值取决于进程类型、处理器类型和操作系统的配置。 例如，对于 x86 处理器上的大多数 32 位进程，此值约为 2 GB，对于在启用了 4 GT RAM 优化的系统上运行的可识别大型地址的 32 位进程，此值约为 3 GB。  dwAvailVirtual  当前位于调用进程的虚拟地址空间的用户模式部分中的未保留和未提交的内存量（以字节为单位）。 注解 **MEMORYSTATUS** 反映调用时内存的状态。 它还反映当时分页文件的大小。 操作系统可以将分页文件放大到管理员设置的最大大小。  在内存超过 4 GB 的计算机上， **MEMORYSTATUS** 结构可能会返回错误的信息，报告值 –1 以指示溢出。 如果应用程序存在此行为的风险，请使用 [GlobalMemoryStatusEx](https://learn.microsoft.com/zh-cn/windows/desktop/api/sysinfoapi/nf-sysinfoapi-globalmemorystatusex) 函数而不是 [GlobalMemoryStatus](https://learn.microsoft.com/zh-cn/windows/desktop/api/winbase/nf-winbase-globalmemorystatus) 函数。 示例 有关示例，请参阅 [GlobalMemoryStatus](https://learn.microsoft.com/zh-cn/windows/desktop/api/winbase/nf-winbase-globalmemorystatus) 函数。 要求  |  |  | | --- | --- | | **标头** | winbase.h (包括 Windows.h) |  另请参阅 [GlobalMemoryStatus](https://learn.microsoft.com/zh-cn/windows/desktop/api/winbase/nf-winbase-globalmemorystatus)  [GlobalMemoryStatusEx](https://learn.microsoft.com/zh-cn/windows/desktop/api/sysinfoapi/nf-sysinfoapi-globalmemorystatusex)  [内存性能信息](https://learn.microsoft.com/zh-cn/previous-versions/windows/desktop/legacy/aa965225(v=vs.85)) |

## MEMORYSTATUSEX结构体

|  |  |  |
| --- | --- | --- |
| 包含有关物理内存和虚拟内存（包括扩展内存）的当前状态的信息。 [GlobalMemoryStatusEx](https://learn.microsoft.com/zh-cn/windows/desktop/api/sysinfoapi/nf-sysinfoapi-globalmemorystatusex) 函数在此结构中存储信息。 语法 C++复制  typedef struct \_MEMORYSTATUSEX {  DWORD dwLength;  DWORD dwMemoryLoad;  DWORDLONG ullTotalPhys;  DWORDLONG ullAvailPhys;  DWORDLONG ullTotalPageFile;  DWORDLONG ullAvailPageFile;  DWORDLONG ullTotalVirtual;  DWORDLONG ullAvailVirtual;  DWORDLONG ullAvailExtendedVirtual;  } MEMORYSTATUSEX, \*LPMEMORYSTATUSEX; 成员 dwLength  结构大小（以字节为单位）。 在调用之前，必须设置此成员 [GlobalMemoryStatusEx](https://learn.microsoft.com/zh-cn/windows/desktop/api/sysinfoapi/nf-sysinfoapi-globalmemorystatusex).  dwMemoryLoad  一个介于 0 和 100 之间的数字，指定正在使用的物理内存的近似百分比 (0 表示不使用内存，100 表示) 已满内存使用。  ullTotalPhys  实际物理内存量（以字节为单位）。  ullAvailPhys  当前可用的物理内存量（以字节为单位）。 这是可以立即重复使用的物理内存量，而无需先将其内容写入磁盘。 它是备用列表、可用列表和零列表的大小之和。  ullTotalPageFile  系统或当前进程的当前已提交内存限制，以字节为单位，以较小者为准。 若要获取系统范围的已提交内存限制，请调用 [GetPerformanceInfo](https://learn.microsoft.com/zh-cn/windows/desktop/api/psapi/nf-psapi-getperformanceinfo)。  ullAvailPageFile  当前进程可以提交的最大内存量（以字节为单位）。 此值等于或小于系统范围的可用提交值。 若要计算系统范围的可用提交值，请调用 [GetPerformanceInfo](https://learn.microsoft.com/zh-cn/windows/desktop/api/psapi/nf-psapi-getperformanceinfo)，并从 **CommitLimit** 的值中减去 **CommitTotal** 的值。  ullTotalVirtual  调用进程的虚拟地址空间的用户模式部分的大小（以字节为单位）。 此值取决于进程类型、处理器类型和操作系统的配置。 例如，对于 x86 处理器上的大多数 32 位进程，此值约为 2 GB，对于在启用了 [4 GB 优化](https://learn.microsoft.com/zh-cn/windows/desktop/Memory/4-gigabyte-tuning) 的系统上运行的大地址感知的 32 位进程，此值约为 3 GB。  ullAvailVirtual  当前位于调用进程的虚拟地址空间的用户模式部分中的未保留和未提交的内存量（以字节为单位）。  ullAvailExtendedVirtual  保留。 此值始终为 0。 注解 **MEMORYSTATUSEX** 反映调用时内存的状态。 它还反映当时分页文件的大小。 操作系统可以将分页文件放大到管理员设置的最大大小。  返回的物理内存大小包括来自所有节点的内存。 示例 有关示例，请参阅 [GlobalMemoryStatusEx](https://learn.microsoft.com/zh-cn/windows/desktop/api/sysinfoapi/nf-sysinfoapi-globalmemorystatusex) 函数。 要求  |  |  | | --- | --- | | **标头** | sysinfoapi.h (包括 Windows.h) |  另请参阅 [GlobalMemoryStatusEx](https://learn.microsoft.com/zh-cn/windows/desktop/api/sysinfoapi/nf-sysinfoapi-globalmemorystatusex)  [内存性能信息](https://learn.microsoft.com/zh-cn/previous-versions/windows/desktop/legacy/aa965225(v=vs.85)) |

## GlobaMemoryStatus函数的语法

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| [**GlobalMemoryStatus** 可以返回不正确的信息。 请改用 [GlobalMemoryStatusEx](https://learn.microsoft.com/zh-cn/windows/desktop/api/sysinfoapi/nf-sysinfoapi-globalmemorystatusex) 函数。]  检索有关系统当前使用物理内存和虚拟内存的信息。 语法 C++复制  void GlobalMemoryStatus(  [out] LPMEMORYSTATUS lpBuffer  ); 参数 [out] lpBuffer  指向 [MEMORYSTATUS](https://learn.microsoft.com/zh-cn/windows/desktop/api/winbase/ns-winbase-memorystatus) 结构的指针。 **GlobalMemoryStatus** 函数将有关当前内存可用性的信息存储到此结构中。 返回值 无 备注 在内存超过 4 GB 的计算机上， **GlobalMemoryStatus** 函数可能会返回错误的信息，报告值 –1 以指示溢出。 因此，应用程序应改用 [GlobalMemoryStatusEx](https://learn.microsoft.com/zh-cn/windows/desktop/api/sysinfoapi/nf-sysinfoapi-globalmemorystatusex) 函数。  在内存超过 2 GB 且内存小于 4 GB 的 Intel x86 计算机上，**GlobalMemoryStatus** 函数将始终在 [MEMORYSTATUS](https://learn.microsoft.com/zh-cn/windows/desktop/api/winbase/ns-winbase-memorystatus) 结构的 **dwTotalPhys** 成员中返回 2 GB。 同样，如果总可用内存在 2 到 4 GB 之间，则 **MEMORYSTATUS** 结构的 **dwAvailPhys** 成员将向下舍入为 2 GB。 如果使用 **/LARGEADDRESSAWARE** 链接器选项链接可执行文件，则 **GlobalMemoryStatus** 函数将在两个成员中返回正确的物理内存量。  **GlobalMemoryStatus** 函数返回的信息是可变的。 不能保证对此函数的两次连续调用将返回相同的信息。 要求  |  |  | | --- | --- | | **标头** | winbase.h (包括 Windows.h) | | **Library** | Kernel32.lib | | **DLL** | Kernel32.dll |  另请参阅 [GlobalMemoryStatusEx](https://learn.microsoft.com/zh-cn/windows/desktop/api/sysinfoapi/nf-sysinfoapi-globalmemorystatusex)  [MEMORYSTATUS](https://learn.microsoft.com/zh-cn/windows/desktop/api/winbase/ns-winbase-memorystatus)  [内存管理函数](https://learn.microsoft.com/zh-cn/windows/desktop/Memory/memory-management-functions)  [内存性能信息](https://learn.microsoft.com/zh-cn/previous-versions/windows/desktop/legacy/aa965225(v=vs.85))  [虚拟地址空间和物理存储](https://learn.microsoft.com/zh-cn/windows/desktop/Memory/virtual-address-space-and-physical-storage) |

## GlobaMemoryStatusEx函数的语法

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 检索有关系统当前物理内存和虚拟内存使用情况的信息。 语法 C++复制  BOOL GlobalMemoryStatusEx(  [in, out] LPMEMORYSTATUSEX lpBuffer  ); 参数 [in, out] lpBuffer  指向 [MEMORYSTATUSEX](https://learn.microsoft.com/zh-cn/windows/desktop/api/sysinfoapi/ns-sysinfoapi-memorystatusex) 结构的指针，该结构接收有关当前内存可用性的信息。 返回值 如果该函数成功，则返回值为非零值。  如果函数失败，则返回值为零。 要获得更多的错误信息，请调用 GetLastError。 注解 可以使用 **GlobalMemoryStatusEx** 函数来确定应用程序可以分配多少内存，而不会严重影响其他应用程序。  **GlobalMemoryStatusEx** 函数返回的信息是可变的。 无法保证对此函数的两次顺序调用将返回相同的信息。  *lpBuffer* 中 [MEMORYSTATUSEX](https://learn.microsoft.com/zh-cn/windows/desktop/api/sysinfoapi/ns-sysinfoapi-memorystatusex) 结构的 **ullAvailPhys** 成员包括所有 NUMA 节点的内存。 示例 以下代码演示 **了 GlobalMemoryStatusEx** 函数的简单用法。  C++复制  // Sample output:  // There is 51 percent of memory in use.  // There are 2029968 total KB of physical memory.  // There are 987388 free KB of physical memory.  // There are 3884620 total KB of paging file.  // There are 2799776 free KB of paging file.  // There are 2097024 total KB of virtual memory.  // There are 2084876 free KB of virtual memory.  // There are 0 free KB of extended memory.  #include <windows.h>  #include <stdio.h>  #include <tchar.h>  // Use to convert bytes to KB  #define DIV 1024  // Specify the width of the field in which to print the numbers.  // The asterisk in the format specifier "%\*I64d" takes an integer  // argument and uses it to pad and right justify the number.  #define WIDTH 7  void \_tmain()  {  MEMORYSTATUSEX statex;  statex.dwLength = sizeof (statex);  GlobalMemoryStatusEx (&statex);  \_tprintf (TEXT("There is %\*ld percent of memory in use.\n"),  WIDTH, statex.dwMemoryLoad);  \_tprintf (TEXT("There are %\*I64d total KB of physical memory.\n"),  WIDTH, statex.ullTotalPhys/DIV);  \_tprintf (TEXT("There are %\*I64d free KB of physical memory.\n"),  WIDTH, statex.ullAvailPhys/DIV);  \_tprintf (TEXT("There are %\*I64d total KB of paging file.\n"),  WIDTH, statex.ullTotalPageFile/DIV);  \_tprintf (TEXT("There are %\*I64d free KB of paging file.\n"),  WIDTH, statex.ullAvailPageFile/DIV);  \_tprintf (TEXT("There are %\*I64d total KB of virtual memory.\n"),  WIDTH, statex.ullTotalVirtual/DIV);  \_tprintf (TEXT("There are %\*I64d free KB of virtual memory.\n"),  WIDTH, statex.ullAvailVirtual/DIV);  // Show the amount of extended memory available.  \_tprintf (TEXT("There are %\*I64d free KB of extended memory.\n"),  WIDTH, statex.ullAvailExtendedVirtual/DIV);  } 要求  |  |  | | --- | --- | | **标头** | sysinfoapi.h (包括 Windows.h) | | **Library** | Kernel32.lib | | **DLL** | Kernel32.dll |  另请参阅 [MEMORYSTATUSEX](https://learn.microsoft.com/zh-cn/windows/desktop/api/sysinfoapi/ns-sysinfoapi-memorystatusex)  [内存管理函数](https://learn.microsoft.com/zh-cn/windows/desktop/Memory/memory-management-functions)  [内存性能信息](https://learn.microsoft.com/zh-cn/previous-versions/windows/desktop/legacy/aa965225(v=vs.85))  [虚拟地址空间和物理存储](https://learn.microsoft.com/zh-cn/windows/desktop/Memory/virtual-address-space-and-physical-storage) |

# 演练

## 1.新建一个c++的常规空项目,取名Lesson78-Mem-status,然后新建一个cpp文件取名memstatus.cpp,然后添加一些基本代码

|  |
| --- |
|  |

## 2.获取64位windows内存状态的代码如下,注意:需要使用带有Ex的API和结构体

|  |
| --- |
| #include<Windows.h>  #include<stdio.h>  int main()  {  MEMORYSTATUSEX mse;  mse.dwLength = sizeof(mse);  if(!GlobalMemoryStatusEx(&mse))  {  printf("获取内存状态失败!错误码:%d\n",GetLastError());  }  int factor = 1024 \* 1024 \* 1024;  printf("物理内存总量:%dGB\n",mse.ullTotalPhys/factor);  printf("可用物理内存总:%dGB\n",mse.ullAvailPhys/factor);  printf("页面总容量:%dGB\n",mse.ullTotalPageFile/factor);  printf("可用页面容量:%dGB\n",mse.ullAvailPageFile/factor);  printf("虚拟内存总量:%dGB\n",mse.ullTotalVirtual/factor);  printf("可用虚拟内存总量:%dGB\n",mse.ullTotalVirtual/factor);  system("pause");  return 0;  } |

### 运行程序,效果如下

|  |
| --- |
|  |

# 这一节的学习到此为止