

简易任务管理器

姓名：高圣迪

0. Introduction

项目名称**简易任务管理器的实现**。需要实现展示：进程名，进程PID，用户名，CPU占用（以百分比展示），内存占用（取峰值内存量），IO读字节数，IO写字节数，页错误数，线程数，句柄数，GDI对象数。

其中大部分是理解的名词，页错误数，句柄数，GDI对象数需要理解。

页错误数：页面错误是指软件在读取或写入标记为“不存在”的虚拟内存位置时发生的中断。页错误数记录了一个进程必须从磁盘上恢复的次数。

句柄数：由进程打开的当前句柄数。句柄是Windows用来标识被应用程序所建立或使用的对象的唯一整数。比如应用程序的实例，控制，位图，GDI对象等

GDI对象数：GDI（Graphics Device Interface），用于应用程序和设备驱动程序的交互。

1. Data Structures

我用了一个Map来存所有进程，key设了PID，value是我定义的一个结构体，具体含义如注释所示！

```
struct Process {
    WCHAR name[MAX_PATH];           //进程名
    WCHAR user_name[MAX_PATH];      //用户名
    float CPU;                       //CPU占用
    int64_t last_time = 0;          //用于记录第一次的时间，求差算CPU占用
    int64_t last_system_time = 0;
    SIZE_T memory;                  //内存占用
    DWORD IOread;                   //IO读字节数
    DWORD IOwrite;                  //IO写字节数
    DWORD pagefault;                //页错误数
    DWORD thread_cnt;               //线程数
    DWORD handle_cnt;               //句柄数
    DWORD GDI_cnt;                  //GDI对象数
};
```

2. Functions

我一共构建了8个函数

```

map<DWORD, Process> GetPidAndProcessNameAndThreadCnt(HANDLE hSnapshot);
void GetProcessUser(map<DWORD, Process>& list, HANDLE hSnapshot);
uint64_t convert_time_format(const FILETIME* ftime);
void GetCpu(map<DWORD, Process>& list);
void GetMemoryAndPageFault(map<DWORD, Process>& list);
void GetIOReadAndWrite(map<DWORD, Process>& list);
void GetHandleCnt(map<DWORD, Process>& list);
void GetGdiCnt(map<DWORD, Process>& list);

```

在main函数里，我利用 CreateToolhelp32Snapshot 得到一个快照，其中包括的进程信息，线程信息等。利用这个快照，遍历next，我可以构建一个map，并通过看 PROCESSENTRY32 的源码往 Process里写入一些信息（进程名，所持线程数）。

借助 WTSEnumerateProcesses 遍历之前的快照，可以得到每个进程的SID，然后根据SID得到用户名。

为了得到CPU的占比，一开始我的想法是像网站一样对同一个Process打开两次然后取delta求占比，但是这样需要 sleep() 一段时间（否则会-nan(ind)）。为了不影响效率，我选择在struct里加入 last_time和last_system_time然后遍历Process两次求delta。这样的好处是我可以先得到一次time，遍历其他Process占用的时间等效替代了sleep的时间。

剩下的内存占用，IO读写字节数，页错误数，句柄数，GDI对象数都有 GetProcess 方法可以直接调用，比较简单，不再赘述。

3. Result

Microsoft Visual Studio 调试控制台

There are 244 processes!

Process Name	ProcessID	User Name	CPU (%)	Memory (B)	IO Read	IO Write	Page Fault	Threads	Handles	GDI
(System Process)	0	SYSTEM	84.935059	0	0	0	0	8	0	0
System	4	SYSTEM	0.000000	0	0	0	0	266	0	0
Registry	124	SYSTEM	0.000000	0	0	0	0	4	0	0
Lsf.exe	296	40280	0.000000	17387520	5012	3922	4867	1	238	37
services.exe	412	SYSTEM	0.000000	0	0	0	0	9	0	0
smss.exe	508	SYSTEM	0.000000	0	0	0	0	2	0	0
lsass.exe	548	SYSTEM	0.000000	28327936	9846563	1245926	85701	11	1991	0

任务管理器

名称	PID	用户名	CPU	峰值工作集...	页面错误	句柄	线程	GDI 对象	I/O 读取字节	I/O 写入字节
系统中断	-	SYSTEM	00	0 K	0	-	-	0	0	0
系统空闲进程	0	SYSTEM	96	12 K	9	-	8	0	0	0
System	4	SYSTEM	00	8,852 K	29,367	6,221	261	0	120,022,132	4,007,739,593
Registry	124	SYSTEM	00	152,724 K	5,838,382	-	4	0	1,626,112	339,738,624
Lsf.exe	296	40280	00	16,980 K	4,867	238	1	37	5,012	3,922
services.exe	412	SYSTEM	00	12,888 K	144,784	821	10	0	229,684	160

可以看到以Lsf.exe为例，数据都与Windows的任务管理器是对得上的！

4. Reference

PROCESSENTRY32 structure (tlhelp32.h)

<https://docs.microsoft.com/en-us/windows/win32/api/tlhelp32/ns-tlhelp32-processentry32>

C++获取windows中所有进程，使用pid杀死进程，并用map保存进程id与name以备后用

https://blog.csdn.net/liunan199481/article/details/89381802?utm_medium=distribute.pc_relevant.none-task-blog-baidujs_baidulandingword-0&spm=1001.2101.3001.4242

老生常谈---通过PID获取用户名的方法。我先写一个

<https://bbs.csdn.net/topics/360187493>

C++ 获取进程CPU占用率

<https://www.jb51.net/article/159205.htm>

收集进程的内存使用情况信息

<https://docs.microsoft.com/zh-cn/windows/win32/psapi/collecting-memory-usage-information-for-a-process?redirectedfrom=MSDN>

GetProcessHandleCount() -- 获取属于指定进程的已打开句柄数

<https://blog.csdn.net/nemo2011/article/details/7920532>

Windows下C++获取进程cpu使用率，内存使用，IO情况

https://blog.csdn.net/qg_32619837/article/details/94555100

【API】获取当前进程的用户对象和GDI对象

https://blog.csdn.net/shen_001/article/details/79231098