# Tool Help API的使用

## Tool Help API

Tool Help API分为几个部分：Heap Walking，Process Walking， Thread Walking，Module Walking，分别对应堆、进程、线程、模块操作，使用Tool Help API 可以方便的实现堆、进程、线程、模块的遍历。

## 系统快照

在遍历之前，先要获取系统的快照，方法是使用函数CreateToolhelp32Snapshot函数，该函数原型如下：

HANDLE WINAPI CreateToolhelp32Snapshot(DWORD dwFlags, DWORD th32ProcessID);

该函数第一个参数为快照标志，可以有为以下值：

|  |  |
| --- | --- |
| **Value** | **Description** |
| TH32CS\_SNAPALL | Equivalent to specifying TH32CS\_SNAPHEAPLIST, TH32CS\_SNAPMODULE, TH32CS\_SNAPPROCESS, and TH32CS\_SNAPTHREAD. |
| TH32CS\_SNAPHEAPLIST | Includes the heap list of the specified process in the snapshot. |
| TH32CS\_SNAPMODULE | Includes the module list of the specified process in the snapshot. |
| TH32CS\_SNAPPROCESS | Includes the process list in the snapshot. |
| TH32CS\_SNAPTHREAD | Includes the thread list in the snapshot. |
| TH32CS\_GETALLMODS | Includes all modules in the snapshot. |
| TH32CS\_SNAPNOHEAPS | By default, the process heap information is included when creating PROCESS snapshot.  For a more efficient way of receiving the basic information of the process, use this flag with TH32CS\_SNAPPROCESS. |

常用为值为红色部分。

函数的第二个参数为进程ID，可以由GetCurrentProcessId()函数或得。

函数返回值是一个快照句柄，可以用XXXX32First函数和XXXX32Next函数实现快照的遍历。

## 遍历函数

下面以遍历进程为例，说明Tool Help API的使用。

遍历进程用到的函数有：

BOOL WINAPI Process32First(HANDLE hSnapshot, LPPROCESSENTRY32 lppe);

BOOL WINAPI Process32Next(HANDLE hSnapshot, LPPROCESSENTRY32 lppe);

函数第一个参数为快照句柄，第二个参数为进程信息的结构体，定义如下：

typedef struct tagPROCESSENTRY32

{

DWORD dwSize;

DWORD cntUsage;

DWORD th32ProcessID; // this process

ULONG\_PTR th32DefaultHeapID;

DWORD th32ModuleID; // associated exe

DWORD cntThreads;

DWORD th32ParentProcessID; // this process's parent process

LONG pcPriClassBase; // Base priority of process's threads

DWORD dwFlags;

CHAR szExeFile[MAX\_PATH]; // Path

} PROCESSENTRY32;

下面就来解释 PROCESSENTRY32 中的结构成员的含义：

dwSize：                            PROCESSENTRY32 结构体的大小， 在使用 Process32First 之前需要将这个成员设置好。

cntUsage：                       这个值的结果必须为 1 。

th32ProcessID：              这个参数就很明显了，代表一个进程的 ID，即进程标识符。

th32DefaultHeapID：     这个参数的话代表了进程的默认堆标识符。

th32ModuleID：              这个参数代表了进程的模块标识符。

cntThreads：                    代表进程所启动的线程数目。

th32ParentProcessID：  代表该进程的父进程 ID。

pcPriClassBase：             该值代表由该进程所创建的线程所拥有的基本优先级。

dwFlags：                         保留，暂未使用。

szExeFile：                        返回该进程的可执行文件所在的路径。

th32MemoryBase：       该可执行文件所加载的基地址。

th32AccessKey：            访问标识符。

在使用 Process32First函数前，必须先为结构体的dwSize赋值。

## 示例程序

#include "stdafx.h"

#include <Windows.h>

#include <TlHelp32.h>

#include <iostream>

#include <stdlib.h>

using namespace std;

int \_tmain(int argc, \_TCHAR\* argv[])

{

HANDLE hSnapshot=CreateToolhelp32Snapshot(TH32CS\_SNAPPROCESS,GetCurrentProcessId());

if (hSnapshot!=INVALID\_HANDLE\_VALUE)

{

//遍历快照

PROCESSENTRY32 pe;

pe.dwSize=sizeof(PROCESSENTRY32);

if (ERROR\_NO\_MORE\_FILES==Process32First(hSnapshot,&pe))

{

CloseHandle(hSnapshot);

return 0;

}

do

{

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

printf("PROCESS NAME :%s\n",pe.szExeFile);

cout<<"PARENT PROCESS ID:"<<pe.th32ParentProcessID<<endl;

cout<<"PROCESS ID :"<<pe.th32ProcessID<<endl;

cout<<"THREADS COUNT :"<<pe.cntThreads<<endl;

cout<<"PRIORITY BASE :"<<pe.pcPriClassBase<<endl;

} while (Process32Next(hSnapshot,&pe));

}

CloseHandle(hSnapshot);

system("pause");

return 0;

}