

# Re: RegisterWaitForSingleObject & memory leaks

---

*Source:*

<http://www.tech-archive.net/Archive/Development/microsoft.public.win32.programmer.kernel/2005-10/msg00321.htm>

---

- *From:* "Joe" <joe@xxxxxxxxxx>
  - *Date:* Wed, 19 Oct 2005 13:51:33 -0700
- 

You must use CloseHandle() on the hProcess and hThread handles in PROCESSINFO.

"stef" <stef.pellegrino@xxxxxxxx> wrote in message  
<news:1129751267.389681.147530@xx>

> Hello guys,  
>  
> I've "found" the pretty nice "RegisterWaitForSingleObject" to notify  
> (for example) a child death.  
>  
> But, with the example below, I don't know why and where, I noticed I  
> lost some significant handles after all children stops !?  
>  
> I think I'm not using the function correctly but where ?  
>  
>  
> Could you help me please...  
>  
>  
> note :  
> the "child.exe" is a simple puts("hello"); while(1);  
>  
> #include <windows.h>  
>  
>  
> BOOL (WINAPI \*\_RegisterWaitForSingleObject)(PHANDLE, HANDLE, PVOID,  
> PVOID, ULONG, ULONG);  
> BOOL (WINAPI \*\_UnregisterWait)(HANDLE);  
>  
> typedef struct  
> {  
> DWORD ProcessId;  
> HANDLE hProcess;  
> HANDLE hThread;  
> HANDLE hNew;  
> } S\_PI;  
>  
>

## Re: RegisterWaitForSingleObject & memory leaks

```
>
>
>
> /**
> *
> */
> main()
> {
> HINSTANCE h;
> int i;
> STARTUPINFO si;
> PROCESS_INFORMATION pi;
> S_PI *pInfo;
> BOOL b;
>
> h = LoadLibrary("kernel32.dll");
>
> _RegisterWaitForSingleObject = GetProcAddress(h,
> "RegisterWaitForSingleObject");
> _UnregisterWait = GetProcAddress(h, "UnregisterWait");
>
>
> ZeroMemory(&si, sizeof(STARTUPINFO));
> si.cb= sizeof(STARTUPINFO);
> si.dwFlags|= STARTF_USESTDHANDLES;
>
> // at this moment, ProcessExplorer said 10 handles used
> //
> for(i=0; i<12; i++)
> {
> pInfo = (S_PI *) malloc(sizeof(S_PI));
>
> if (CreateProcess(NULL, "c:/child.exe", NULL, NULL, FALSE, 0, NULL,
> NULL, &si, &pi)==0)
> {
> puts("error");
> exit(1);
> }
>
> pInfo->ProcessId = pi.dwProcessId;
> pInfo->hProcess = pi.hProcess;
> pInfo->hThread = pi.hThread;
>
> b = _RegisterWaitForSingleObject(&pInfo->hNew, pi.hProcess, fct,
> (PVOID) pInfo, INFINITE, 0x00000008);
> }
>
>
> while(1);
> scanf(">>%d", &i);
> FreeLibrary(h);
```

## Re: RegisterWaitForSingleObject & memory leaks

```
> }
>
>
>
>
>
> /**
> * when the 12 children are stopped, I "stay" between 15/22 handles used
> and
> * never down to 10 handles !!
> *
> */
> void NTAPI fct(void *ptr, BOOLEAN TimerOrWaitFired)
> {
> S_PI *p = (S_PI *) ptr;
>
> printf("child %d stops\n", p->ProcessId);
> CloseHandle(p->hProcess);
> CloseHandle(p->hThread);
>
> _UnregisterWait(p->hNew);
>
> free(ptr);
> CloseHandle(p->hNew);
> }
>
```

---

- **Follow-Ups:**

- ◆ **[Re: RegisterWaitForSingleObject & memory leaks](#)**

- ◇ From: stef

- **References:**

- ◆ **[RegisterWaitForSingleObject & memory leaks](#)**

- ◇ From: stef

- Prev by Date: **[RegisterWaitForSingleObject & memory leaks](#)**
- Next by Date: **[Re: Reasonable values for CRITICAL\\_SECTION spin-count](#)**
- Previous by thread: **[RegisterWaitForSingleObject & memory leaks](#)**
- Next by thread: **[Re: RegisterWaitForSingleObject & memory leaks](#)**
- Index(es):
  - ◆ **[Date](#)**
  - ◆ **[Thread](#)**