

RE: Turn off mouse pointer when no mouse is plugged in

Source:

<http://www.tech-archive.net/Archive/WinXP/microsoft.public.windowsxp.embedded/2008-03/msg00002.html>

- *From:* Schockal <Schockal@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Sat, 1 Mar 2008 18:53:00 -0800
-

Hi There,

I am looking for help with my touch screen driver. I am trying to use the touchscreen driver from Gunze, the zip files include inf, sys and exe files. I followed the method described by Microsoft on how to create a custom component. The component is getting built by target designer. The ufd.inf and ufd.sys are included in the C:/windows directory. FBA runs fine, setupai.log tells that the driver had been installed properly and the device did not start for some unknown reason.

My question is should I see the Gunze driver on the Add/Remove programs list ? . If I install it manually, I see the application on the Add/Remove program list. How come my application is not on the list ?.

Whats the normal procedure to add a third party component to XPE ?.

Thanks in advance,

SC

"Smedly Tonker" wrote:

Background:

My company builds an XPE product with a touch screen for pointer input & so we usually don't want a mouse pointer displayed. Even though our product is not fielded with a mouse I still wanted the capability to plug in a mouse for testing, development, etc. So I searched the internet for a registry setting or a utility that would hide the mouse pointer when no mouse was plugged in and cause it to reappear once a mouse is plugged in. Unfortunately I didn't find a utility that did this. So I built my own.

RE: Turn off mouse pointer when no mouse is plugged in

How my program works:

My program compiles to 6Kb (as long as you link to DLL runtime library and you use the following linker option: /OPT:NOWIN98). My program also relies on a free third party application that I found on the internet called 'nomousy.exe' (13Kb) which allows you to hide the mouse cursor. My program determines when a USB mouse is plugged in or not and then calls 'nomousy.exe' to hide or show the mouse cursor. Note that my program only recognizes when a USB mouse is plugged in (although you could easily fix this). This is because we don't have PS/2 connector on our device.

So if anyone now or in the future has a need for such a utility I have provided the source code to my program below:

```
#include "stdafx.h"

BOOL bMousedPluggedIn = FALSE;

BOOL FindProcess(TCHAR* _szName)
{
    try
    {
        char szName[MAX_PATH], szToTermUpper[MAX_PATH];

        int iLen, iLenP, indx;

        BOOL bResult;

        DWORD aiPID[1000], iCb = 1000, iNumProc, iV2000 = 0;

        DWORD iCbneeded, i;

        HANDLE hProc;

        HINSTANCE hInstLib;
```

RE: Turn off mouse pointer when no mouse is plugged in

```
HMODULE hMod;
```

```
if(!_szName == NULL))
```

```
return FALSE;
```

```
if((*_szName == NULL))
```

```
return FALSE;
```

```
// Transfer Process name into "szToTermUpper" and
```

```
// convert it to upper case
```

```
iLenP = strlen(_szName);
```

```
if((iLenP < 1) || (iLenP > MAX_PATH))
```

```
return FALSE;
```

```
for(indx=0; indx < iLenP; indx++)
```

```
szToTermUpper[indx] = toupper(_szName[indx]);
```

```
szToTermUpper[iLenP] = 0;
```

```
// PSAPI Function Pointers.
```

```
BOOL (WINAPI *lpfEnumProcesses)( DWORD*, DWORD cb, DWORD*);
```

```
BOOL (WINAPI *lpfEnumProcessModules)( HANDLE, HMODULE*, DWORD,  
LPDWORD );
```

```
DWORD (WINAPI *lpfGetModuleBaseName)( HANDLE, HMODULE, LPTSTR,  
DWORD );
```

```
if((hInstLib = LoadLibraryA("PSAPI.DLL")) == NULL)
```

RE: Turn off mouse pointer when no mouse is plugged in

RE: Turn off mouse pointer when no mouse is plugged in

```
return FALSE;

// Get procedure addresses.

lpfEnumProcesses = (BOOL (WINAPI*)(DWORD*, DWORD, DWORD*))
GetProcAddress(hInstLib, "EnumProcesses");

lpfEnumProcessModules = (BOOL (WINAPI*)(HANDLE, HMODULE*, DWORD,
LPDWORD)) GetProcAddress(hInstLib, "EnumProcessModules");

lpfGetModuleBaseName = (DWORD (WINAPI*)(HANDLE, HMODULE, LPTSTR,
DWORD )) GetProcAddress(hInstLib, "GetModuleBaseNameA");

if((lpfEnumProcesses == NULL) || (lpfEnumProcessModules == NULL) ||
(lpfGetModuleBaseName == NULL))

{

FreeLibrary(hInstLib);

return FALSE;

}

bResult = lpfEnumProcesses(aiPID, iCb, &iCbneeded);

if(!bResult)

{

// Unable to get process list, EnumProcesses failed

FreeLibrary(hInstLib);

return FALSE;

}

// How many processes are there?

iNumProc = iCbneeded / sizeof(DWORD);
```

RE: Turn off mouse pointer when no mouse is plugged in

RE: Turn off mouse pointer when no mouse is plugged in

```
// Get and match the name of each process

for(i = 0; i < iNumProc; i++)

{

// Get the (module) name for this process

strcpy(szName, "Unknown");

// First, get a handle to the process

hProc = OpenProcess(PROCESS_QUERY_INFORMATION | PROCESS_VM_READ,
FALSE, aiPID[i]);

// Now, get the process name

if(hProc)

if(lpEnumProcessModules(hProc, &hMod, sizeof(hMod),
&iCbneeded))

iLen = lpGetModuleBaseName(hProc, hMod, szName, MAX_PATH);

CloseHandle(hProc);

// We will match regardless of lower or upper case

if(stricmp(szName, szToTermUpper) == 0)

{

FreeLibrary(hInstLib);

return TRUE;

}

}
```

RE: Turn off mouse pointer when no mouse is plugged in

```
FreeLibrary(hInstLib);

}

catch(...)

{

#ifdef _DEBUG

OutputDebugString("Uncaught exception!! – FindProcess");

#endif

}

return FALSE;

}

void Execute(LPCTSTR lpszApp, LPCTSTR lpszArgs)

{

SHELLEXECUTEINFO sei;

TCHAR szPath[MAX_PATH];

TCHAR* pStrPos;

lstrcpy(szPath, lpszApp);

if((pStrPos = _tcsrchr(szPath, '\\'))

*pStrPos = NULL;

else

*szPath = NULL;

memset(&sei, 0, sizeof(sei));
```

RE: Turn off mouse pointer when no mouse is plugged in

RE: Turn off mouse pointer when no mouse is plugged in

```
sei.cbSize = sizeof(sei);  
  
sei.fMask = SEE_MASK_NOCLOSEPROCESS | SEE_MASK_FLAG_NO_UI;  
  
sei.hwnd = NULL;  
  
sei.lpFile = lpzApp;  
  
sei.lpParameters = lpzArgs;  
  
sei.lpDirectory = szPath;  
  
sei.nShow = SW_HIDE;
```

```
ShellExecuteEx(&sei);
```

```
}
```

```
void MousePointer()
```

```
{
```

```
static TCHAR szNoMousyExe[] = "nomousy.exe";
```

```
static BOOL bFirst = TRUE;
```

```
static BOOL bNoMousy = FALSE;
```

```
if(bFirst)
```