Creazione di una DLL in Win32Asm

Salve! In questo tutorial vedremo come creare in Win32Asm una semplicissima DLL che esporterà due funzioni che mostrano due message box. Dunque, prima di tutto partiamo con un po' (poca poca) di teoria. Allora dovete sapere, che una DLL è un comune file eseguibile di windows, che non fa altro che esportare funzioni, variabili e (in C++) classi, e le mette a disposizione di altre applicazioni qualora ne richiedano l'uso.
Una dll è divisa fondamentalmente in due parti, la prima è quella fatta dal codice vero e proprio, mentre la seconda si tratta di un file def, che contiene le funzioni che andranno esportate dalla dll.
Ok, iniziamo subito a creare la nostra DLL :) :

```
.386p
.model flat, stdcall
option casemap:none
Include windows.inc
Include kernel32.inc
Include user32.inc
IncludeLib kernel32.lib
IncludeLib user32.lib

.data
Message1 db "MessageBox chiamata nella prima funzione esportata", 0
Message2 db "MessageBox chiamata nella seconda funzione esportata", 0
Title db "FirstDll", 0

.code
DllEntry PROC hInstDLL:HINSTANCE, reason:DWORD, reserved1:DWORD ; entry point della dll
  mov eax, TRUE
  ret
DllEntry ENDP

ShowFirstMessage PROC ; prima funzione esportata
  invoke MessageBox, NULL, ADDR Message1, ADDR Title, MB_OK
  ret
ShowFirstMessage ENDP

ShowSecondMessage PROC ; seconda funziona esportata
  invoke MessageBox, NULL, ADDR Message2, ADDR Title, MB_OK
  ret
ShowSecondMessage ENDP

End DllEntry
```

Ok, ora ci serve il file def, che è fatto in questo modo:

```
LIBRARY FirstDll
EXPORTS
ShowFirstMessage @1
ShowSecondMessage @2
```

Il file def è un file che contiene il nome della libraria, seguito dalla parola EXPORTS e a capo le funzioni da esportare, seguite da @n, dove n è il numero della funzione.
Ok, ora salvate il file sorgente come FirstDll.asm e il file def come FirstDll.def, e quindi assemblate e linkate in questo modo:

```
ml /c /coff /Cp FirstDll.asm
link /dll /SUBSYSTEM:WINDOWS /DEF:FirstDll.def FirstDll.obj
```

Ed avrete la vostra bella DLL :). Ora però ci serve un programma per caricarla, eccolo qua (ho usato il metodo LoadLibrary/GetProcAddress, in questo modo la dll viene mappata in memoria SOLO quando lo vogliamo noi, e non quando windows carica il processo):

```
.386p
.model flat, stdcall

Include windows.inc
Include kernel32.inc
Include user32.inc
IncludeLib kernel32.lib
IncludeLib user32.lib
```

```
.data
DllName db "FirstDll.dll", 0 ; nome della DLL
Func1 db "ShowFirstMessage", 0 ; nome della prima funzione
Func2 db "ShowSecondMessage", 0 ; nome della seconda funzione

.data?
hInstance HINSTANCE ?
hDll dd ?
FuncAddr dd ?

.code
start:

  invoke GetModuleHandle, 0
  mov hInstance, eax
  invoke LoadLibrary, ADDR DllName ; mappa la DLL nella memoria del processo
  mov hDll, eax
  invoke GetProcAddress, eax, ADDR Func1 ; prende l'indirizzo della prima funzione esportata dalla dll
  call eax ; chiama la prima funzione
  invoke GetProcAddress, hDll, ADDR Func2 ; prende l'indirizzo della seconda funzione esportata dalla dll
  call eax ; chiama la seconda funzione
  invoke FreeLibrary, hDll ; scarica la libreria
  invoke ExitProcess, 0

end start
```

Ok, salvate questo file e chiamatelo loader.asm, e poi assemblate e linkate in questo modo:

```
ml /c /coff /Cp loader.asm
link /SUBSYSTEM:WINDOWS loader.obj
```

Eseguite il file loader.exe, ed ecco la prima message box, premete ok ed ecco la seconda! :)
Ok, per adesso è tutto, so che questo tutorial è molto corto, ma preferisco fare le cose semplici in modo che si possano capire facilmente.
Per informazioni, maledizioni, bestemmioni, parolacce, cose serie, ecc... mandate una mail a quake2th@libero.it :).

Ciao a tutti!

Quake2