|  |
| --- |
| * **DllMain** |
| DLL은 아래 와 같은 Entry함수를 가질 수 있다. DLL이 프로세스에 처음으로 Load될 때 DllMain이 호출된다. |
| **BOOL WINAPI** **DllMain(** **HINSTANCE** *hinstDLL***, DWORD** *fdwReason***, LPVOID** *lpvReserved* **);** |
| 다음은 DllMain의 각 인자에 대한 설명이다. |
| [in]***hinstDLL*** |
| Dll 의 핸들. DLL이 Load된 곳의 메모리 주소가 들어 온다. |
| [in] ***fdwReason*** |
| DllMain이 호출된 이유가 들어 있다. DLL\_PROCESS\_ATTACH, DLL\_PROCESS\_DETACH, DLL\_THREAD\_ATTACH, DLL\_THREAD\_DETACH 중의 1개의 값을 갖는다. |
| [in] ***lpvReserved*** |
| Dll이 Load된 방법. DLL이 명시적으로 Load된 경우(LoadLibrary, FreeLibrary 사용) NULL 이 들어 오고, 암시적으로 Load된 경우 NULL 이 아닌 값이 들어 온다. |
| * **언제 호출되는가 ?** |
| DllMain은 아래의 4가지 경우에 호출된다. |
| * DLL이 프로세스의 주소공간에 처음으로 Load될 때 호출된다.( DLL\_PROCESS\_ATTACH ) * DLL이 놓여 있는 프로세스에서 새로운 스레드가 생성 될 때 호출된다.(DLL\_THREAD\_ATTACH) * DLL이 놓여 있는 프로세스에서 실행 중이던 스레드가 종료 될 때 호출된다.(DLL\_THREAD\_DETACH) * DLL이 프로세스에서 Unload 될 때 호출된다.(DLL\_PROCESS\_DETACH) |

|  |
| --- |
| * **DLL 내의 전역변수** |
| A.exe와 B.exe 라는 2개의 실행 파일이 한 개의 DLL(XYZ.dll)을 사용한다고 가정해 보자. 또한 XYZ.dll안에는 share\_buffer는 버퍼가 있다고 가정하자. 이때 DLL을 사용하는 두 개의 프로세스가 DLL안에 있는 버퍼를 공유할수 있을까 ? |
| 안 된다. 일반적으로 exe나 DLL안의 전역변수들은 기본적으로 PAGE\_WRITECOPY 속성이 지정되기 때문에 쓰기전에 복사본이 만들어 진다.( Copy-On-Write) 그러므로 A와 B는 둘 다 다른 물리 공간을 사용하게 된다. |
|  |
| * **DLL 내의 공유 메모리** |
| 하지만 DLL내에 공유 메모리를 만들어 사용하면 서로 다른 2개의 프로세스가 DLL 내의 공유 메모리를 사용해서 통신할 수 있다. |
| 다음의 코드는 DLL 공유 섹션을 만드는 방법을 보여준다. |
| **#pragma data\_seg(“SHARED”)**  **char shared\_buffer[1024] = {0};**  **#pragma data\_seg()**  **#pragma comment( linker, “/subsystem:SHARED,RWS”)** |
| * **예제 – 공유 메모리를 사용한 프로세스간 통신** |
|  |
| 먼저 DLL 안에 Shared\_Buffer라는 공유 메모리를 만든다. 그리고 공유 메모리의 접근을 동기화하기 위해 뮤텍스 객체를 만든다. 마지막으로 A가 Data를 넣고 B에게 알려주기 위해 Event 객체를 사용한다. |
| **XYZ.h, XYZ.cpp : DLL의 소스와 헤더 파일.**  **Sender.h Sender.cpp : 공유 메모리에 Data를 넣는 프로그램.**  **Receiver.h Receiver.cpp : 공유 메모리에서 Data를 꺼내는 프로그램.** |

// XYZ.h

#ifndef \_XYZ\_H\_

#define \_XYZ\_H\_

#ifdef \_\_cplusplus

#define EXTERN\_C extern "C"

#else

#define EXTERN\_C

#endif

#ifdef DLL\_SOURCE

#define EXPORT \_\_declspec(dllexport)

#else

#define EXPORT \_\_declspec(dllimport)

#endif

EXTERN\_C EXPORT void SetSharedData( const char\* s );

EXTERN\_C EXPORT int GetSharedData( char\* s, int len);

#endif // \_XYZ\_H\_

// XYZ.c

#include <string.h>

#pragma data\_seg("SHARED")

char shared\_data[1024] = { 0 };

#pragma data\_seg()

#pragma comment(linker, "/section:SHARED,RWS")

void SetSharedData( const char\* s ){

strcpy( shared\_data, s );

}

int GetSharedData( char\* s, int len){

if ( len > 1024 ) return 0;

strcpy( s, shared\_data );

return 1;

}