작성자(학번 이름) : 2019180052 한창우

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **주차** | 3주차 | **기간** | 2024.7. 8 ~ 14 | **지도교수** |  |
| 이번주 한일 요약 | Windows Via C/C++  챕터 3, 4 “커널 오브젝트, 프로세스” | | | | |

<상세 수행내용>

**챕터3. 커널 오브젝트**

**커널 오브젝트**

프로세스, 스레드, 파일 등과 같은 수많은 리소스를 관리하기 위해 커널 오브젝트를 사용한다.

커널 오브젝트에는 리소스를 다루는데 필요한 정보를 담고 있다, 하지만 커널 오브젝트에 담는 정보는 각 리소마다 다르다. 예를 들어 프로세스는 ID, 기본 우선순위와 종료 코드 같은 정보들을 가지고 있는 반면 파일 오브젝트의 경우 바이트 오프셋, 공유 모드, 오픈 모드와 같은 정보를 가지고 있다.

커널 오브젝트를 생성하면 반환값으로 커널 오브젝트를 구분하기 위한 핸들값을 준다. 그래서 사용자는 직접적으로 커널 오브젝트 구조체에 접근하는 것은 불가능하지만, 이렇게 반환 된 핸들을 이용하여 간접적으로 접근할 수 있다.

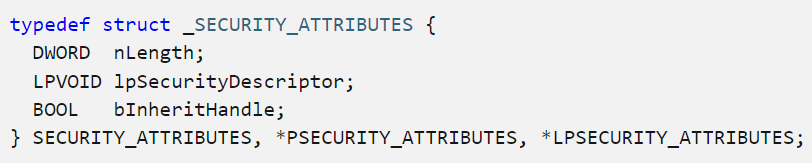
**사용 카운트**

커널 오브젝트는 프로세스가 아니라 커널에 의해 소유된다. 즉, 커널 오브젝트를 생성한 프로세스가 종료되어도 반드시 생성된 커널 오브젝트가 프로세스와 함께 삭제되는 것은 아니다.

그 이유는, 해당 커널 오브젝트를 참조하고 있는 다른 프로세스가 존재할 수 있기 때문이다.

참조 횟수가 바로 “사용 카운트” 이다. 그리고 커널은 이 “사용 카운트”를 참조하여 커널 오브젝트의 생명 주기를 관리하는 것이다.

**보안**

****

커널 오브젝트는 보안 디스크립터를 통해 보호될 수 있다. 보안 설정을 위해 위 구조체를 생성하고 커널 오브젝트 생성 함수 인자로 넘겨주면 된다. NULL을 인자로 넘기면 현재 프로세스의 보안 토큰을 근간으로 하는 기본 보안 디스크립터를 사용한다.

오브젝트에는 커널 오브젝트와 유저 오브젝트 또는 GUI 오브젝트들이 있다, 하지만 커널 오브젝트는 생성할 때, 보안 특성을 지정하는 매개변수를 요구한다.

**커널 오브젝트 핸들 테이블**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | 인덱스 |  |  | | 커널 오브젝트의 메모리 블록을 가리키는 포인터 | 엑세스 마스크 | |  | | --- | | 플래그 | |
|  |  |  |  |

커널 오브젝트를 생성하면 커널은 프로세스의 핸들 테이블을 조사하여 비어 있는 공간을 찾아내고, 해당 인덱스에 커널 오브젝트를 가리키는 포인터 정보와 그 외 정보들을 입력한다.

핸들 테이블은 서로 다른 프로세스라면 공유되는 것이 아니라서 같은 핸들값이라고 해도 서로 다른 커널 오브젝트를 가리킨다.

**커널 오브젝트 삭제**

CloseHandle 함수를 통해 커널 오브젝트를 삭제할 수 있다. 하지만 호출하지 않다면 누수가 발생할까?

만약 프로세스가 실행 중이라면 누수가 일어나지만, 프로세스가 종료된다면 시스템이 모든 오브젝트 핸들을 정리해 주는 것을 보장하기 때문에 커널 오브젝트 누수문제는 발생하지 않게 된다.

**프로세스간 커널 오브젝트의 공유**

서로 다른 프로세스에서 동일 커널 오브젝트를 공유해야하는 경우들이 있다.

* 파일-매핑 오브젝트를 통해 데이터 블록을 공유
* 뮤텍스, 세마포어, 이벤트와 같은 오브젝트를 공유하여 스레드간에 동기화를 수행할 수 있다.

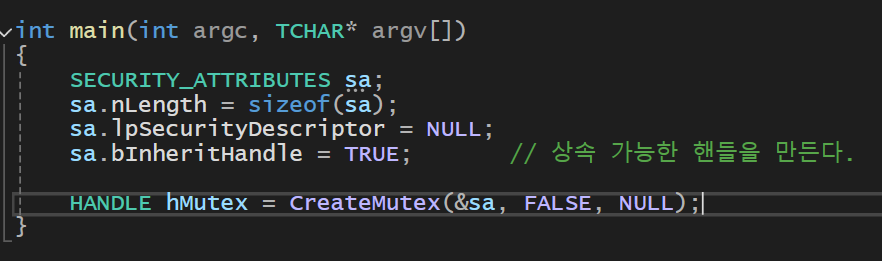
커널 오브젝트를 공유하는 방법으로 3가지 방법이 있다.

* 오브젝트 핸들의 상속을 이용하는 방법
* 명명된 오브젝트를 사용하는 방법
* 오브젝트 핸들의 복사를 이용하는 방법

**오브젝트 핸들의 상속을 이용하는 방법**

오브젝트 핸들의 상속을 이용하려면 부모-자식 관계의 프로세스에서만 가능하다.

SECURITY\_ATRRIBUTES 구조체를 상속가능하게 만들고 Create함수를 호출할 때 인자로 전달해야한다.



위 뮤텍스는 SECURITY\_ATRRIBUTES 구조체를 이용하여 생성했기 때문에 상속가능한 핸들이 된다, 하지만 구조체 대신에 NULL을 인자로 보내면 상속 불가능한 핸들이라는 뜻이다.

**CreateProcess 함수**



bInheritHandles 매겨변수로 TRUE를 전달하면 자식 프로세스는 부모 프로세스의 상속 가능한 핸들 값들을 상속하게 된다. 그러면 커널에서 프로세스를 생성한 후에 부모 프로세스의 핸들 테이블을 참조하여 상속 가능한 핸들을을 자식 프로세스의 핸들 테이블에 복사 한다. 그리고 커널 오브젝트 내의 사용 카운트는 증가하게 된다.

-자식 프로세스에게 핸들을 알려주는 방법들

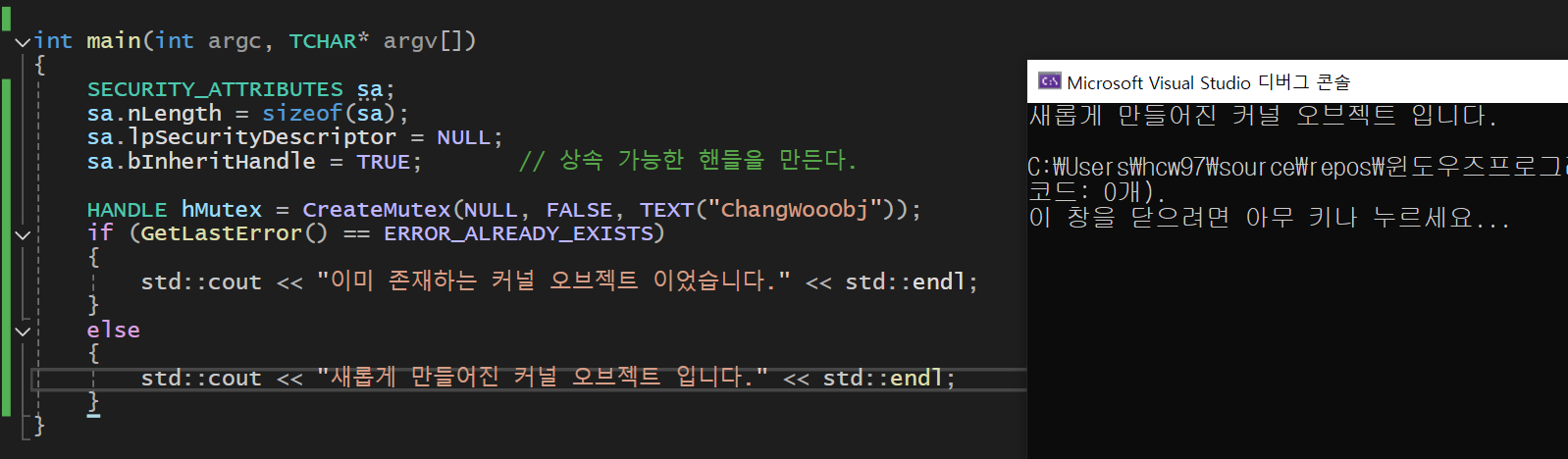
* 자식 프로세스에게 핸들값을 전달하는 가장 일반적인 방법은 명령행 인자로 전달
* 부모 프로세스가 환경변수 블록에 상속할 커널 오브젝트의 핸들 값을 환경변수에 추가하여 전달 (GetEnvironmentVariable 함수 사용)
* 프로세스간 통신 방법을 이용 (추후에 알아볼 내용)

**명명된 오브젝트를 사용하는 방법**



커널 오브젝트를 생성하는 Create 함수 중에서 인자로 이름을 받는 함수들이 있다.

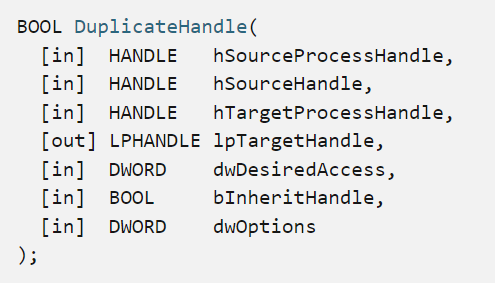
이러한 함수들은 이름을 통하여 프로세스간에 커널 오브젝트를 공유할 수 있다.

위 그림에서는 첫번째 인자로 NULL이 전달되어 상속 불가능한 핸들을 뜻하지만, 인자로 전달된 이름을 이용하여 두 프로세스간에 커널 오브젝트를 공유할 수 있다.

명명된 오브젝트를 사용하여 공유를 하기 위해서는, 일단 이름이 겹쳐서는 안된다.

그래서 위 그림처럼 내가 만든 커널 오브젝트가 이미 만들어진 오브젝트를 참조하는건지, 지금 생성된 오브젝트인지 확인할 수 있으며, 유일한 이름으로 오브젝트를 생성하는 방법이다.

**오브젝트 핸들의 복사를 이용하는 방법**



위 함수는 특정 프로세스 핸들 테이블 내의 항목을 다른 프로세스 핸들 테이블로 복사하는 함수이다.

자식 프로세스의 핸들 테이블에 상속할 커널 오브젝트의 핸들을 복사는 하지만, 실제로 자식 프로세스가 그 커널오브젝트의 핸들값을 알려면 IPC 통신을 통해 알려야한다.

# 챕터4. 프로세스

## 프로세스 & 애플리케이션 작성

프로세는 일반적으로 수행 중인 프로그램의 인스턴스이다. 프로세스도 프로세스의 정보를 관리하는 커널 오브젝트가 있다.

프로세스는 자력으로 실행될 수 없으며 하나의 실행 단위인 스레드가 있어야 한다. 시스템은 프로그램이 처음 실행되면 첫 번째 스레드를 생성해 주는데, 이를 “주 스레드” 라고 부른다.

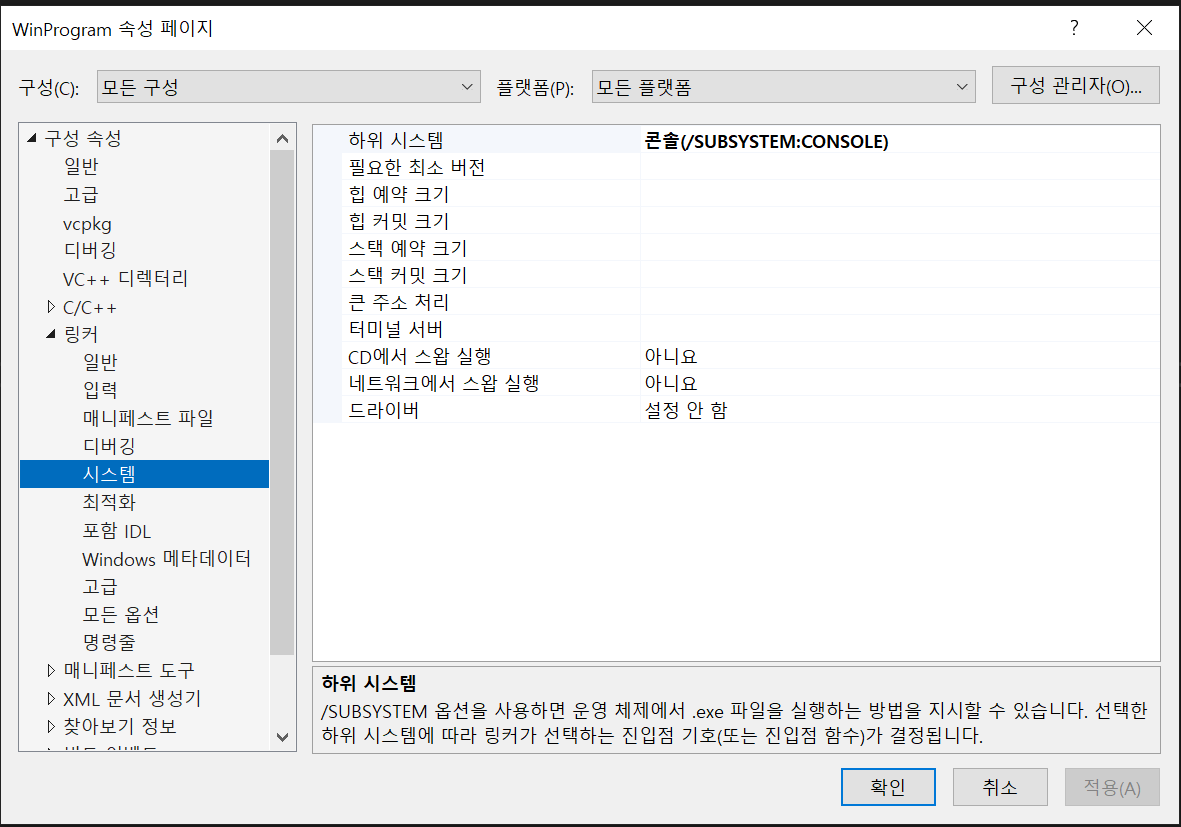
Visual Studio는 실행파일의 형태에 알맞는 서브시스템 값을 포함하여 생성한다.

그래서 우리가 프로그램을 실행하면, 운영체제의 로더는 프로그램의 헤더를 확인하여 CUI기반인지 GUI기반인지 확인을 하여 거기에 맟는 서브시스템 값을 가져오게 되는 것이다.

서브시스템 값에 따라 진입점 함수가 달라진다.

CUI 기반 : main

GUI 기반 : WinMain



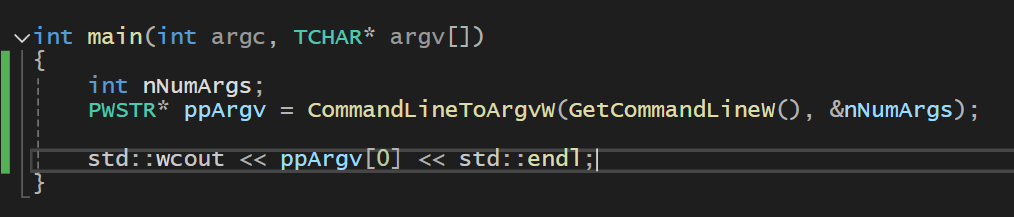
속성창에서 서브 시스템값을 확인하거나 변경할 수 있다.

다시 본론으로 돌아오면, 우리가 프로그램을 실행하면 진입점 함수를 실행하기 전에 운영체제는 C/C++ 런타임 시작함수를 호출한다. 런타임 시작함수가 실행하는 코드는 다음과 같다.

* 새로운 프로세스의 전체 명령행을 가리키는 포인터를 획득
* 새로운 프로세스의 환겨변수를 가리키는 포인터를 획득
* C/C++ 런타임 라이브러리의 전역변수를 초기화한다. 사용자 코드가 StdLib.h 파일을 include 하면 이 변수에 접근할 수 있다.
* C/C++ 런타임 라이브러리의 메모리 할당 함수와 저수준 입출력 루틴이 사용하는 힙을 초기화한다.
* 모든 전역 오브젝트와 static C++ 클래스 오브젝트의 생성자를 호출한다.

이러한 과정을 지나고나서야 C/C++ 시작함수는 비로소 진입점 함수(main 또는 WinMain) 함수를 호출한다.

### **프로세스 명령행**



운영 체제가 프로세스에게 명령행 인수를 전달한다.

위 그림은 명령행 블록을 가리키는 포인터를 CommandLineToArgvW 라는 함수의 인자로 전달해서

명령행에 무슨 내용이 있는지 출력하는 실습을 해봤다.

### **프로세스 환경변수**

모든 프로세스는 자기 자신과 연관된 환경블록을 가지고 있다. 환경블록이란 프로세스의 주소 공간에 할당된 메모리 블록을 말한다.

## 프로세스의 종료

프로세스는 다음과 같이 4가지 방법으로 종료된다

* 주 스레드의 진입점 함수가 반환된다. (추천)
* 프로세스 내의 어떤 스레드가 ExitProcess 함수를 호출한다. (권장x)
* 다른 프로세스의 스레드가 TerminateProcess 함수를 호출한다. (권장x)
* 프로세스 내의 모든 스레드가 각자 종료된다. (가끔 일어남)

### 1. 주 스레드 진입점 함수의 반환

### **프로세스가 종료되면**

프로세스에 의해 할당되었던 유저 오브젝트, 커널 오브젝트는 모두 파괴된다(만약 다른 프로세스가 참조하면 파괴는 하지 않음). 그리고 프로세스의 종료 코드는 STILL\_ACTVIE에서 설정한 값으로 변경 된다. 프로세스의 커널 오브젝트의 상태가 시그널 상태로 바뀌고 사용 카운트를 1감소 시키며 프로세스 내에 남아 있는 스레드는 종료된다.

진행 상황 유튜브 링크:

|  |  |  |  |
| --- | --- | --- | --- |
| **문제점 정리** |  | | |
| **해결방안** |  | | |
| **다음주차** | 4주차 | **다음기간** | 7. 15~ 7. 21 |
| **다음주 할일** | 4주차에 계획되어있는 내용들 | | |
| **지도 교수**  **Comment** |  | | |