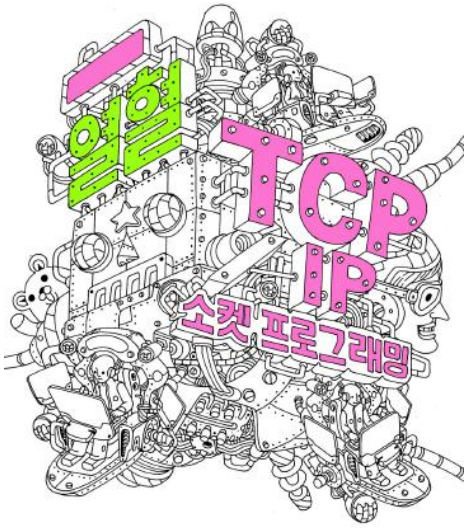
윤성우의 열혈 TCP/IP 소켓 프로그래밍

윤성우 저 열혈강의 TCP/IP 소켓 프로그래밍 개정판

Chapter 19. Windows에서의 쓰레드 사용



Chapter 19-1. 커널 오브젝트(Kernel Objects)

윤성우 저 열혈강의 TCP/IP 소켓 프로그래밍 개정판

커널 오브젝트의 이해

운영체제가 만드는 리소스의 유형

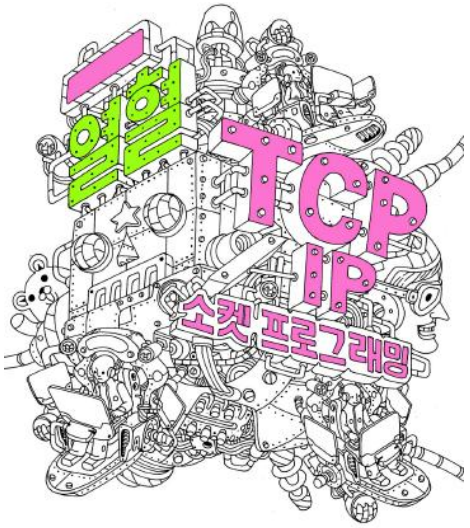
- 프로그램의 실행과 관련된 프로세스와 쓰레드
- 입출력의 도구가 되는 소켓과 파일
- 쓰레드간 동기화의 도구로 사용되는 세마포어 뮤텍스

리소스와 커널 오브젝트의 관계

- 리소스 관리를 위해서 운영체제가 만드는 데이터 블록이 커널 오브젝트이다.
- 커널 오브젝트에는 해당 리소스의 정보가 저장되어 있다.
- 리소스의 종류에 따라서 생성되는 커널 오브젝트의 형태에도 차이가 있다.

커널 오브젝트의 소유자

- 커널 오브젝트의 생성, 관리 및 소멸은 운영체제가 담당한다.
- 즉, 커널 오브젝트의 소유자는 운영체제이다.



Chapter 19-2. 윈도우 기반의 쓰레드 생성

윤성우 저 열혈강의 TCP/IP 소켓 프로그래밍 개정판

프로세스와 쓰레드의 관계

프로세스는 쓰레드가 담기는 상자로 이해

• 운영체제 레벨에서 쓰레드를 지원한다. 따라서 main 함수 호출은 쓰레드에 의해서 이뤄진다.
• 쓰레드를 추가로 생성하지 않으면, 하나의 프로세스 내에 하나의 쓰레드가 생성되어, 쓰레드에 의해서 실행된다.
• 과거 쓰레드를 운영체제 레벨에서 지원하지 않던 시절엔 프로세스가 단순히 쓰레드를 담는 상자로 묘사되지 않았다.

쓰레드에 따른 프로그램의 모델

• 단일 쓰레드 모델의 프로그램
  ✈ 추가로 쓰레드를 생성하는 않은 모델의 프로그램

• 멀티 쓰레드 모델의 프로그램:
  ✈ 프로그램 내에서 추가로 쓰레드를 생성하는 모델의 프로그램

윈도우에서의 쓰레드 생성방법

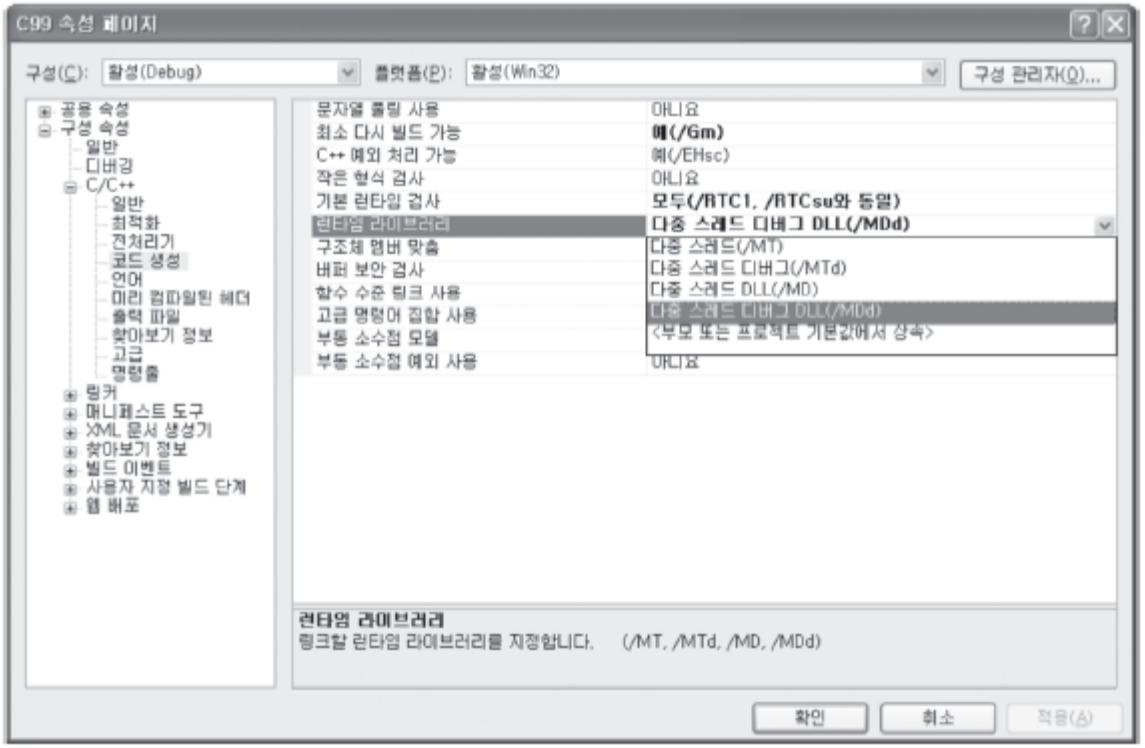
```
#include <windows.h>

HANDLE CreateThread(
    LPSECURITY_ATTRIBUTES lpThreadAttributes,
    SIZE_T dwStackSize,
    LPTHREAD_START_ROUTINE lpStartAddress,
    LPVOID lpParameter,
    DWORD dwCreationFlags,
    LPDWORD lpThreadId
);
```

➜ 성공 시 쓰레드 핸들, 실패 시 NULL 반환

- lpThreadAttributes: 쓰레드의 보안관련 정보전달, 디폴트 보안설정을 위해서 NULL 전달.
- dwStackSize: 쓰레드에게 할당할 스택의 크기를 전달, 0전달하면 디폴트 크기의 스택 생성.
- lpStartAddress: 쓰레드의 main 함수정보 전달.
- lpParameter: 쓰레드의 main 함수호출 시 전달할 인자정보 전달.
- dwCreationFlags: 쓰레드 생성 이후의 행동을 결정, 0을 전달하면 생성과 동시에 실행 가능한 상태가 된다.
- lpThreadId: 쓰레드 ID의 저장을 위한 변수의 주소 값 전달.

이 중에서 실제로 신경 쓸 것은 lpStartAddress와 lpParameter 두 가지 정도이다.

멀티 쓰레드 기반 프로그램 작성을 위한 환경설정



C/C++ 관련 라이브러리를 멀티 쓰레드 기반 라이브러리로 지정을 한다.

그래야 멀티 쓰레드 기반에서 안정적으로 호출가능 한 C/C++ 관련 함수들이 호출된다.

쓰레드 안전한 C 표준함수의 호출 위한 쓰레드 생성

```
#include <process.h>

uintptr_t _beginthreadex(
     void *security,
     unsigned stack_size,
     unsigned ( *start_address )( void * ),
     void *arglist,
     unsigned initflag,
     unsigned *thrdaddr
);
```

➜ 성공 시 쓰레드 핸들, 실패 시 0 반환

앞서 소개한 CreateThread 함수와 비교하여 매개변수가 지니는 의미와 순서까지 동일하다. 그런데 이 함수를 통해서 생성된 쓰레드는 표준 C/C++ 표준함수에 대해서 안정적으로 동작한다. 따라서 표준 C/C++ 함수의 호출이 필요한 프로그램 작성시 반드시 위의 함수를 통해서 쓰레드를 생성해야 한다.

쓰레드 생성의 예


```
#include <process.h>    /* _beginthreadex, _endthreadex */
unsigned WINAPI ThreadFunc(void *arg);

int main(int argc, char *argv[])
{
    HANDLE hThread;
    unsigned threadID;
    int param=5;

    hThread=(HANDLE)_beginthreadex(NULL, 0, ThreadFunc, (void*)&param, 0, &threadID);
    if(hThread==0)
    {
        puts("_beginthreadex() error");
        return -1;
    }
    Sleep(3000);
    puts("end of main");
    return 0;
}
```

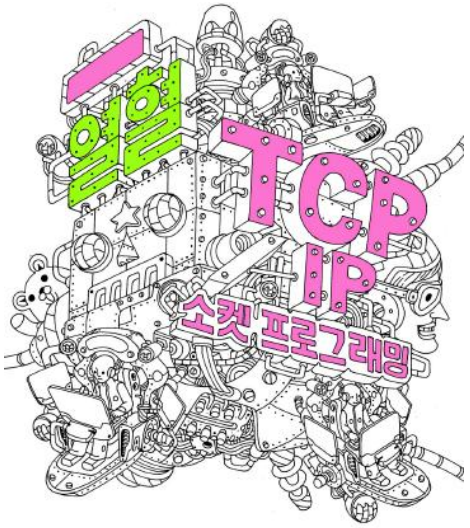
예제 thread1_win.c

```
unsigned WINAPI ThreadFunc(void *arg)
{
    int i;
    int cnt=*((int*)arg);
    for(i=0; i<cnt; i++)
    {
        Sleep(1000); puts("running thread");
    }
    return 0;
}
```

```
running thread      running thread
running thread      running thread
running thread      end of main
end of main         running thread
```

실행결과

리눅스와 달리 윈도우의 쓰레드는 쓰레드 함수를 반환하면 자동으로 소멸이 된다.



Chapter 19-3. 커널 오브젝트의 두 가지 상태

윤성우 저 열혈강의 TCP/IP 소켓 프로그래밍 개정판

커널 오브젝트의 상태, 그리고 상태의 확인

커널 오브젝트의 두 가지 상태

- non-signled 상태

  ✈ 이벤트가 발생하지 않은 상태, 해당 리소스가 특정 상황에 이르지 않은 상태

- signaled 상태

  ✈ 이벤트가 발생한 상태, 해당 리소스가 특정상황에 도달한 상태

커널 오브젝트의 상태 정보를 통해서 리소스의 상황을 인식하게 된다.

예를 들어서 프로세스나 쓰레드의 커널 오브젝트는 프로세스, 쓰레드 생성시 non-signaled 상태에 있다가, 종료 시에 signaled 상태가 된다.

커널 오브젝트의 상태 확인

```
#include <windows.h>

DWORD WaitForSingleObject(HANDLE hHandle, DWORD dwMilliseconds);
```

➜ 성공 시 이벤트 정보, 실패 시 WAIT_FAILED 반환

- hHandle — 상태확인의 대상이 되는 커널 오브젝트의 핸들을 전달.
- dwMilliseconds — 1/1000초 단위로 타임아웃을 지정, 인자로 INFINITE 전달 시, 커널 오브젝트가 signaled 상태가 되기 전에는 반환하지 않는다.
- 반환 값 — signaled 상태로 인한 반환 시, WAIT_OBJECT_0 반환, 타임아웃으로 인한 반환 시 WAIT_TIMEOUT 반환.

전달된 핸들의 커널 오브젝트가 signaled 상태가 되어야 함수가 반환을 한다. 그리고 signaled 상태로 인한 반환 시 WAIT_OBJECT_0가 반환됨에 주목!

다수의 커널 오브젝트를 대상으로 signaled 상태의 관찰에 활용되는 함수

```
#include <windows.h>

DWORD WaitForMultipleObjects(
    DWORD nCount, const HANDLE* lpHandles, BOOL bWaitAll, DWORD dwMilliseconds);
```

➜ 성공 시 이벤트 정보, 실패 시 WAIT_FAILED 반환

- nCount — 검사할 커널 오브젝트의 수 전달.
- lpHandles — 핸들정보를 담고 있는 배열의 주소 값 전달.
- bWaitAll — TRUE 전달 시, 모든 검사대상이 signaled 상태가 되어야 반환, FALSE 전달 시, 검사대상 중 하나라도 signaled 상태가 되면 반환.
- dwMilliseconds — 1/1000초 단위로 타임아웃 지정, 인자로 INFINITE 전달 시, 커널 오브젝트가 signaled 상태가 되기 전에는 반환하지 않는다.

호출된 함수가 반환되면서 자동으로 non-siganed 상태로 변경되는 커널 오브젝트를 가리켜 auto-reset 모드 커널 오브젝트라 한다.
반대 : manual-reset 모드 커널 오브젝트

커널 오브젝트의 상태 확인의 예

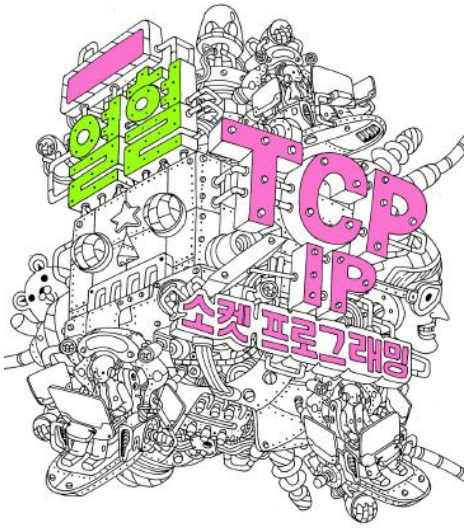
예제 thread2_win.c

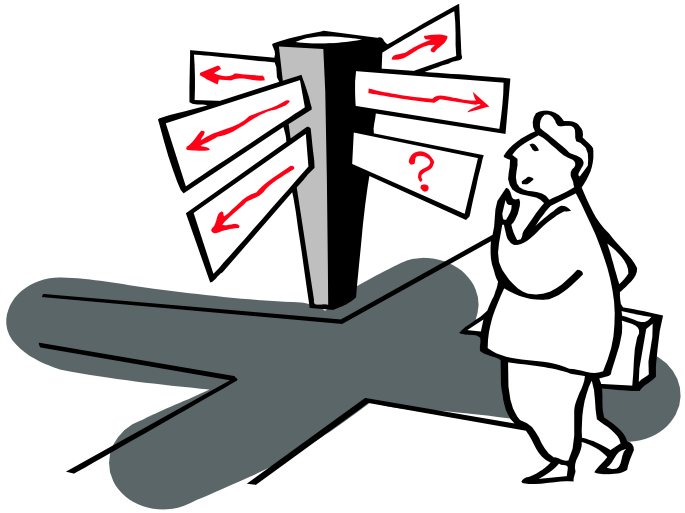
```
int main(int argc, char *argv[])
{
    HANDLE hThread;
    DWORD wr;
    unsigned threadID;
    int param=5;

    hThread=(HANDLE)_beginthreadex(NULL, 0, ThreadFunc, (void*)&param, 0, &threadID);
    if(hThread==0)
    {
        puts("_beginthreadex() error");
        return -1;
    }

    if((wr=WaitForSingleObject(hThread, INFINITE))==WAIT_FAILED)
    {
        puts("thread wait error");
        return -1;
    }

    printf("wait result: %s \n", (wr==WAIT_OBJECT_0) ? "signaled":"time-out");
    puts("end of main");
    return 0;
}
```

실행결과

```
running thread
running thread
running thread
running thread
running thread
wait result: signaled
end of main
```

```
unsigned WINAPI ThreadFunc(void *arg)
{
    int i;
    int cnt=*((int*)arg);
    for(i=0; i<cnt; i++)
    {
        Sleep(1000); puts("running thread");
    }
    return 0;
}
```





Chapter 19가 끝났습니다. 질문 있으신지요?