산업기술의 최고 명문대학
한국산업기술대학교

9장. GUI 소켓 응용 프로그램

네트워크 게임 프로그래밍

Contents

- GUI 응용 프로그램의 구조와 동작 원리를 이해한다.
- GUI 소켓 응용 프로그램 작성 기법을 익힌다.
- 대화상자 기반 응용 프로그램의 구조와 동작 원리를 이해한다.
- 대화상자 기반 소켓 응용 프로그램 작성 기법을 익힌다.

GUI 응용 프로그램 (1)

❖ GUI 응용 프로그램 특징

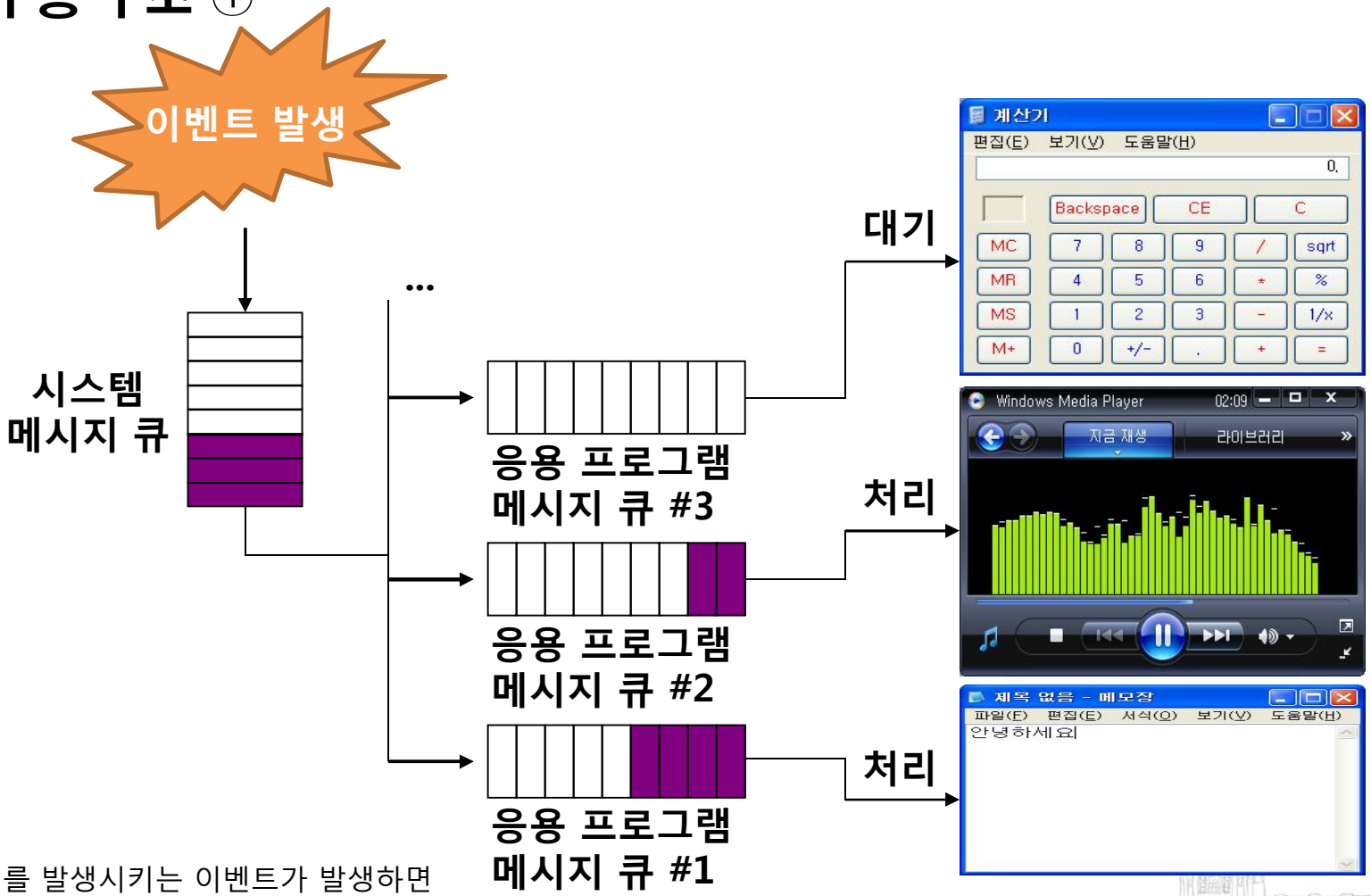
- API로 구현된 편리하고 화려한 사용자 인터페이스 제공
- 메시지 구동 구조로 동작

❖ 용어

- API(*Application Programming Interface*)
  - 윈도우 운영체제가 응용 프로그램에 제공하는 함수 집합
- 메시지(*message*)
  - 윈도우 운영체제가 응용 프로그램의 외부 또는 내부에 변화가 발생했음을 해당 응용 프로그램에 알리는 데 사용하는 개념

GUI 응용 프로그램 (2)

❖ 메시지 구동 구조 ①



1. 외부에서 메시지를 발생시키는 이벤트가 발생하면
2. 운영체제가 관리하는 시스템 메시지 큐에 정보가 저장
3. 각 GUI 응용 프로그램은 운영체제로부터 응용 프로그램 메시지 큐를 할당
4. 운영체제는 시스템 메시지 큐에 저장된 메시지를 적절한 응용 프록램의 메시지 큐에 전달
5. 각 프로그램은 자신의 메시지 큐를 감시하다가 메시지가 발생하여 큐에 들어오면 하나씩 꺼내 처리하고 메시지가 없으면 대기

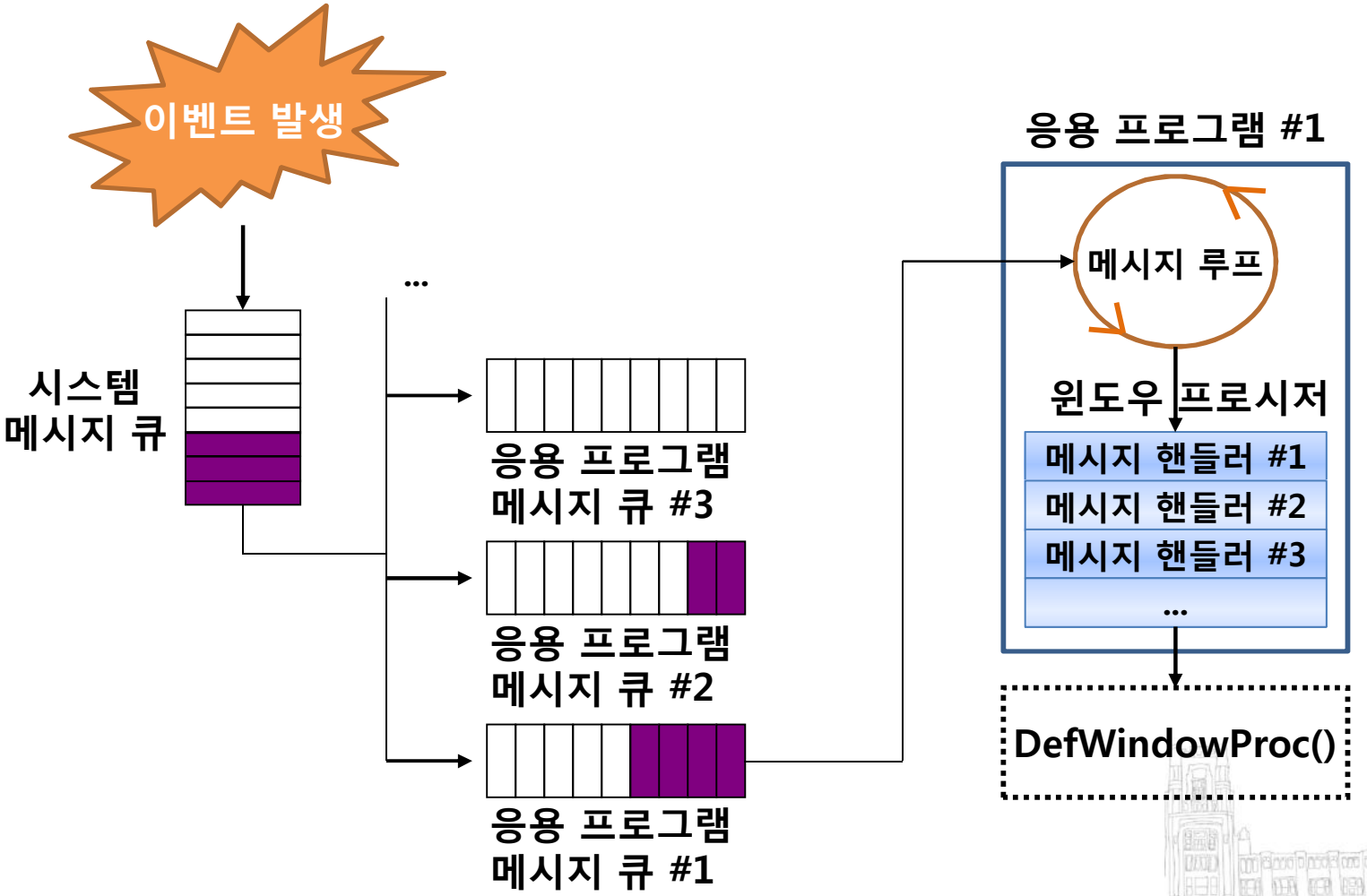
GUI 응용 프로그램 (3)

❖ 메시지 구동 구조 동작 원리 ①

- 외부에서 메시지를 발생시키는 이벤트가 발생하면
- 운영체제가 관리하는 시스템 메시지 큐에 정보가 저장
- 각 GUI 응용 프로그램은 운영체제로부터 응용 프로그램 메시지 큐를 할당
- 운영체제는 시스템 메시지 큐에 저장된 메시지를 적절한 응용 프록램의 메시지 큐에 전달
- 각 프로그램은 자신의 메시지 큐를 감시하다가 메시지가 발생하여 큐에 들어오면 하나씩 꺼내 처리하고 메시지가 없으면 대기함

GUI 응용 프로그램 (4)

❖ 메시지 구동 구조 ② - 앞 구조를 좀더 세부적으로 설명



1. 메시지를 받았을 때 동작을 결정하는 코드를 메시지 핸들러라 칭함
2. 프로그래머는 다양한 핸들러(키보드/마우스/메뉴 메시지등) 를 작성
3. 메시지 핸들러의 집합을 윈도우 프로시저라고 부르며, 메시지 처리 코드를 담고 있는 사용자 정의 함수임

GUI 응용 프로그램 (5)

❖ 메시지 구동 구조 동작 원리 ②

- GUI 응용 프로그램은 윈도우 프로시저에 전달된 메시지를 (메시지 핸들러에서) 자신만의 방식으로 처리
- 처리하지 않은 메시지는 윈도우 운영체제에 맡겨서 자동으로 처리

❖ 용어

- 메시지 핸들러(*message handler*)
  - 메시지를 받았을 때 동작을 결정하는 코드
- 윈도우 프로시저(*window procedure*)
  - 메시지 핸들러의 집합

GUI 응용 프로그램 (6)

❖ 예제 코드

```
#include <windows.h>

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);  // 윈도우 프로시저 원형 선언

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
                   LPSTR lpCmdLine, int nCmdShow)
{
  // 윈도우 클래스를 초기화 하고 등록
  WNDCLASS wndclass;
  wndclass.style = CS_HREDRAW | CS_VREDRAW;
  wndclass.lpfnWndProc = WndProc;
  wndclass.cbClsExtra = 0;
  wndclass.cbWndExtra = 0;
  wndclass.hInstance = hInstance;
  wndclass.hIcon = LoadIcon(NULL, IDI_APPLICATION);
  wndclass.hCursor = LoadCursor(NULL, IDC_ARROW);
  wndclass.hbrBackground = (HBRUSH)GetStockObject(WHITE_BRUSH);
  wndclass.lpszMenuName = NULL;
  wndclass.lpszClassName = "MyWndClass";
  if(!RegisterClass(&wndclass)) return 1;
```

윈도우 클래스는 윈도우의 다양한 특성을 정의하는 구조체로,
원하는 값을 초기화한 후 RegisterClass() 함수를 호출해 운영체제에 등록

GUI 응용 프로그램 (7)

❖ 예제 코드(계속)

```
// 등록한 윈도우 클래스를 기반으로 실제 윈도우 생성
HWND hWnd = CreateWindow("MyWndClass", "WinApp",
   WS_OVERLAPPEDWINDOW, 0, 0, 600, 300, NULL, NULL, hInstance, NULL);
if(hWnd == NULL) return 1;
ShowWindow(hWnd, nCmdShow);  // 처음 생성한 윈도우는 보이지 않으므로 ShowWindows()와 UdateWindows() 를 차례로 호출
UpdateWindow(hWnd);

// 메시지 루프
MSG msg;
while(GetMessage(&msg, NULL, 0, 0) > 0){        // 메시지 큐에서 메시지를 꺼냄
   TranslateMessage(&msg);                      // 키보드 메시지 등이면 처리 한 후,
   DispatchMessage(&msg);                       // 윈도우 프로시저에 전달
}
return msg.wParam;
}
```

GUI 응용 프로그램 (8)

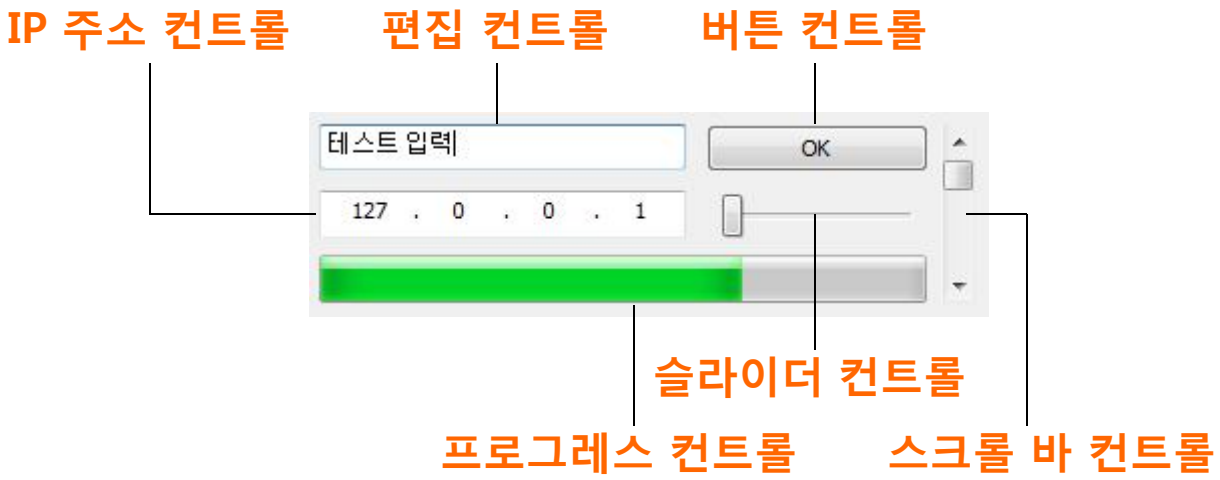
❖ 예제 코드(계속)

```
// 윈도우 메시지를 처리하는 핵심 함수인 윈도우 프로시저
LRESULT CALLBACK WndProc(HWND hWnd, UINT uMsg,
                         WPARAM wParam, LPARAM lParam)
{
  switch(uMsg){
  case WM_CREATE:          // 윈도우가 생성되었을 때 발생하는 메시지 처리
    return 0;
  case WM_SIZE:            // 크기가 변경되었을 때 발생하는 메시지 처리
    return 0;
  case WM_DESTROY:         // 종료 버튼을 눌렀을 때 발생하는 메시지 처리
    PostQuitMessage(0);
    return 0;
  }
  return DefWindowProc(hWnd, uMsg, wParam, lParam); // 처리하지 않은 메시지는 윈도우 운영체제가 알아서 처리
}
```

GUI 응용 프로그램 (9)

❖ 컨트롤

- 표준화된 형태와 특성을 제공하는 일종의 윈도우
  - 사용자의 입력을 받거나 출력을 할 수 있음

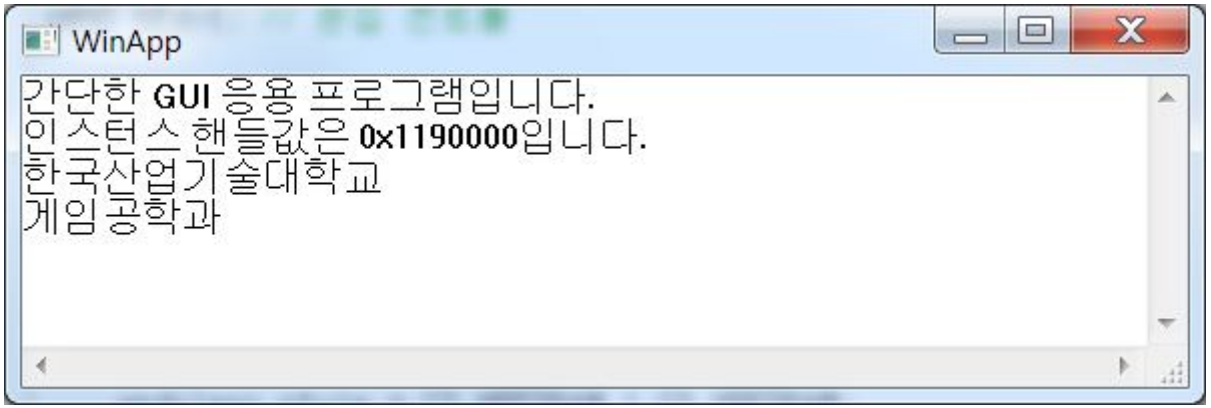


- 독립적인 윈도우가 아닌 자식 윈도우로 존재
  - 부모 윈도우는 SendMessage( ) 함수를 사용해 컨트롤에 메시지를 보냄으로써 컨트롤을 기정의된 방식으로 제어함

GUI 응용 프로그램 작성과 테스트

실습 9-1

P309~

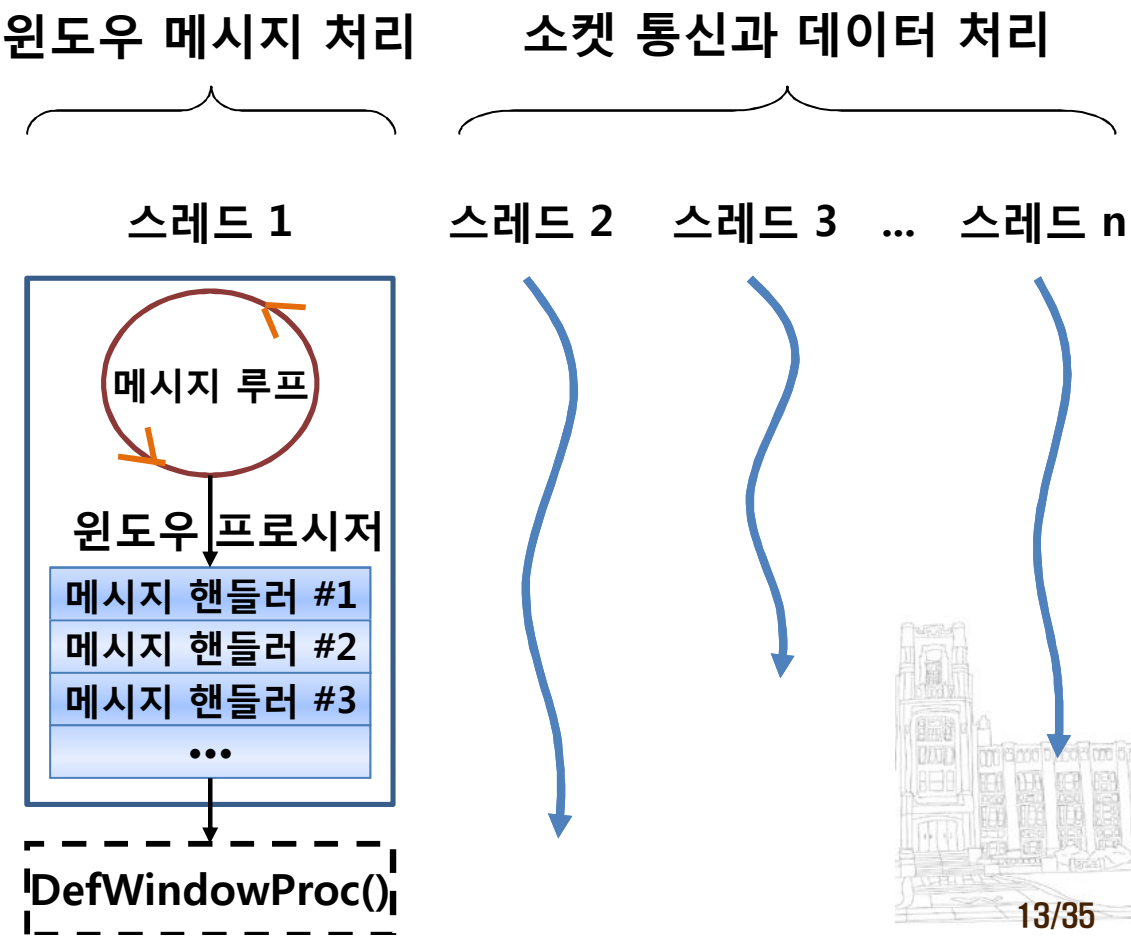


GUI 소켓 응용 프로그램

❖ GUI 소켓 응용 프로그램 구조

- 메시지 구동 구조에서 윈도우 메시지를 처리할 때 시간이 오래 걸리는 작업은 피해야 함
- 시간이 오래 걸리는 작업을 수행할 경우 메시지 루프가 정지하고 이로 인해 다른 메시지를 처리할 수 없음

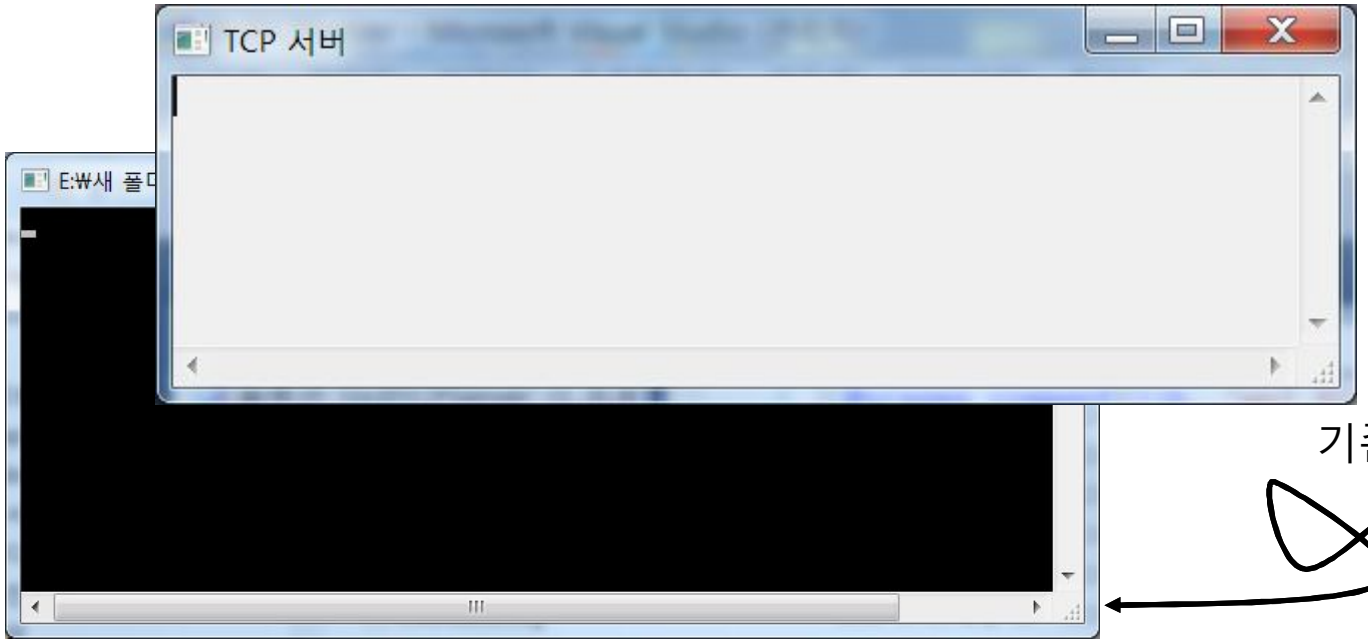
- 소켓 함수들은 이러한 이유로 별도의 스레드로 분리해야 함.
- 윈도우 메시지를 처리하는 스레드 외에 소켓 통신을 담당하거나 데이터를 처리하는 스레드가 별도로 존재



GUI 소켓 응용 프로그램 작성과 테스트

실습 9-2 - GUITCPServer

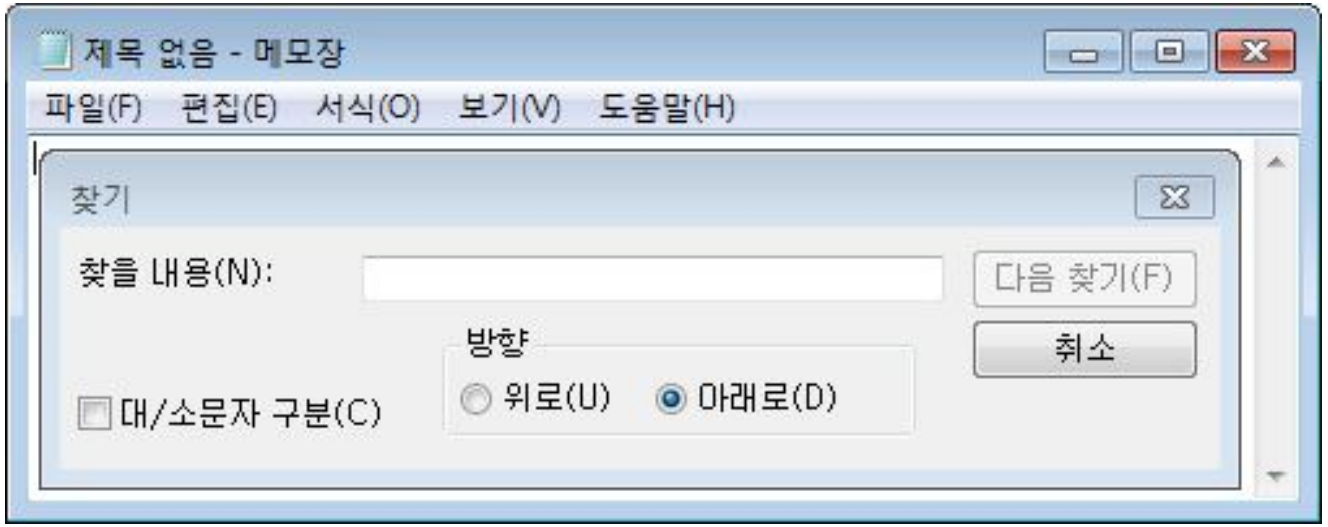
P318~ , 기존 MultithreadTCPServer 예제를 그대로 사용하였음



대화상자 기반 응용 프로그램 (1)

❖ 대화상자

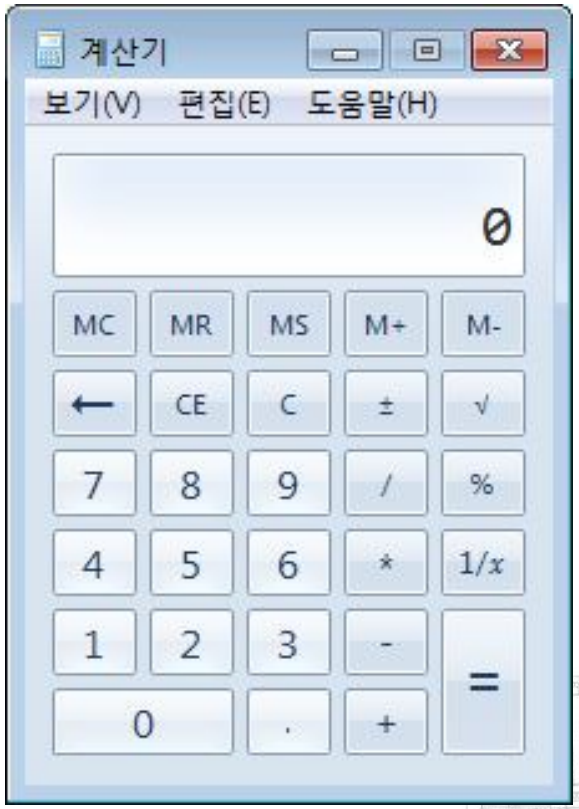
- 다양한 컨트롤을 포함하는 일종의 윈도우
  - 사용자의 입력을 받거나 정보를 출력



대화상자 기반 응용 프로그램 (2)

❖ 대화상자 기반 응용 프로그램 (dialog-based application)

- 대화상자가 독립적인 응용 프로그램으로 존재
- 코드와 더불어 대화상자 템플릿을 만들어야 함
- 대화상자 템플릿 (dialog box template): 대화상자 자체와 대화상자에 포함된 컨트롤에 관한 정보를 담고 있는 이진(binary) 데이터



대화상자 기반 응용 프로그램 (3)

❖ 대화상자 기반 응용 프로그램 작성 순서

❶ 프로젝트에 리소스 파일(*.rc) 추가. 리소스 파일(resource file or resourece script file)이란 대화상자 자체와 대화상자에 포함된 컨트롤에 관한 정보를 담고 있는 텍스트 데이터

❷ 리소스 파일에 대화상자 리소스를 추가하고 비주얼 C++의 리소스 편집기를 이용해 시각적으로 디자인

❸ 프로젝트를 빌드하면 컴파일 단계에서 리소스 컴파일러가 실행되어 *.rc 파일을 이진 포맷인 *.res로 변환

- *.res 파일은 링크 단계에서 실행 파일 내부에 리소스로 포함됨

❹ 대화상자 생성을 요청하는 API 함수 호출

- 윈도우 운영체제는 실행 파일 내부의 대화상자 리소스(=대화상자 템플릿)를 토대로 대화상자를 생성

대화상자 기반 응용 프로그램 (4)

❖ 예제 코드

```
#include <windows.h>
#include "resource.h"

BOOL CALLBACK DlgProc(HWND, UINT, WPARAM, LPARAM);     //대화상자 프로시져

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
                   LPSTR lpCmdLine, int nCmdShow)
{
   DialogBox(hInstance, MAKEINTRESOURCE(IDD_DIALOG1), NULL, DlgProc);  //대화상자 생성
   return 0;
}

BOOL CALLBACK DlgProc(HWND hDlg, UINT uMsg,
                      WPARAM wParam, LPARAM lParam)
{
```

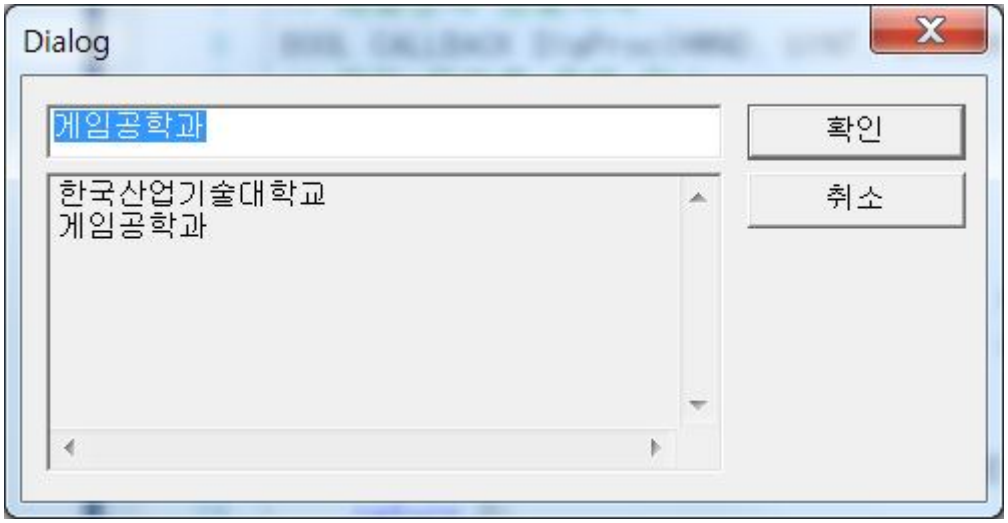
대화상자 기반 응용 프로그램 (5)

❖ 예제 코드(계속)

```
switch(uMsg){
   case WM_INITDIALOG:                    // 대화상자 생성 초기에 발생하는 메시지
      return TRUE;
   case WM_COMMAND:                       // 대화상자에 포함된 컨트롤에서 발생하는 메시지
      switch(LOWORD(wParam)){
      case IDOK:                            //확인버튼을 이용한 대화상자 종료
         EndDialog(hDlg, IDOK);
         return TRUE;
      case IDCANCEL:                        //취소버튼을 이용한 대화상자 종료
         EndDialog(hDlg, IDCANCEL);
         return TRUE;
      }
      return FALSE;
   }
   return FALSE;
}
```

실습 9-3

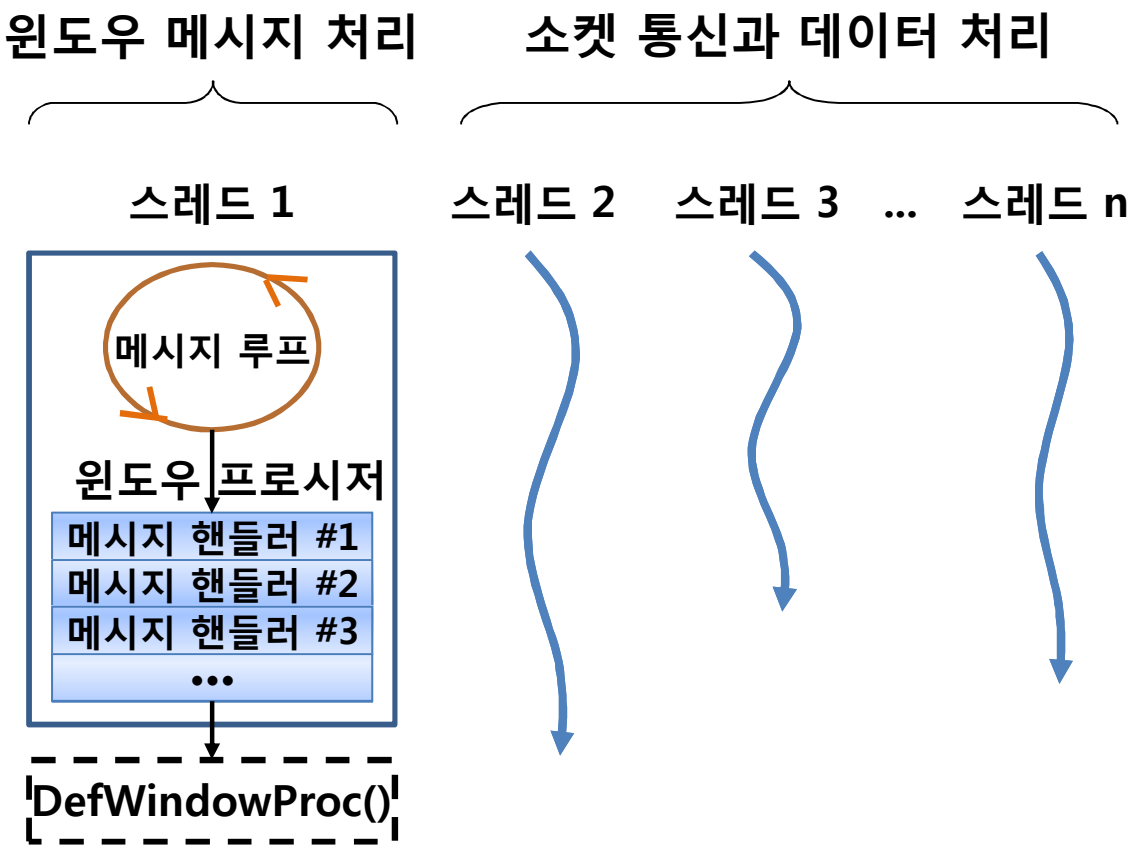
P332~



대화상자 기반 소켓 응용 프로그램 (1)

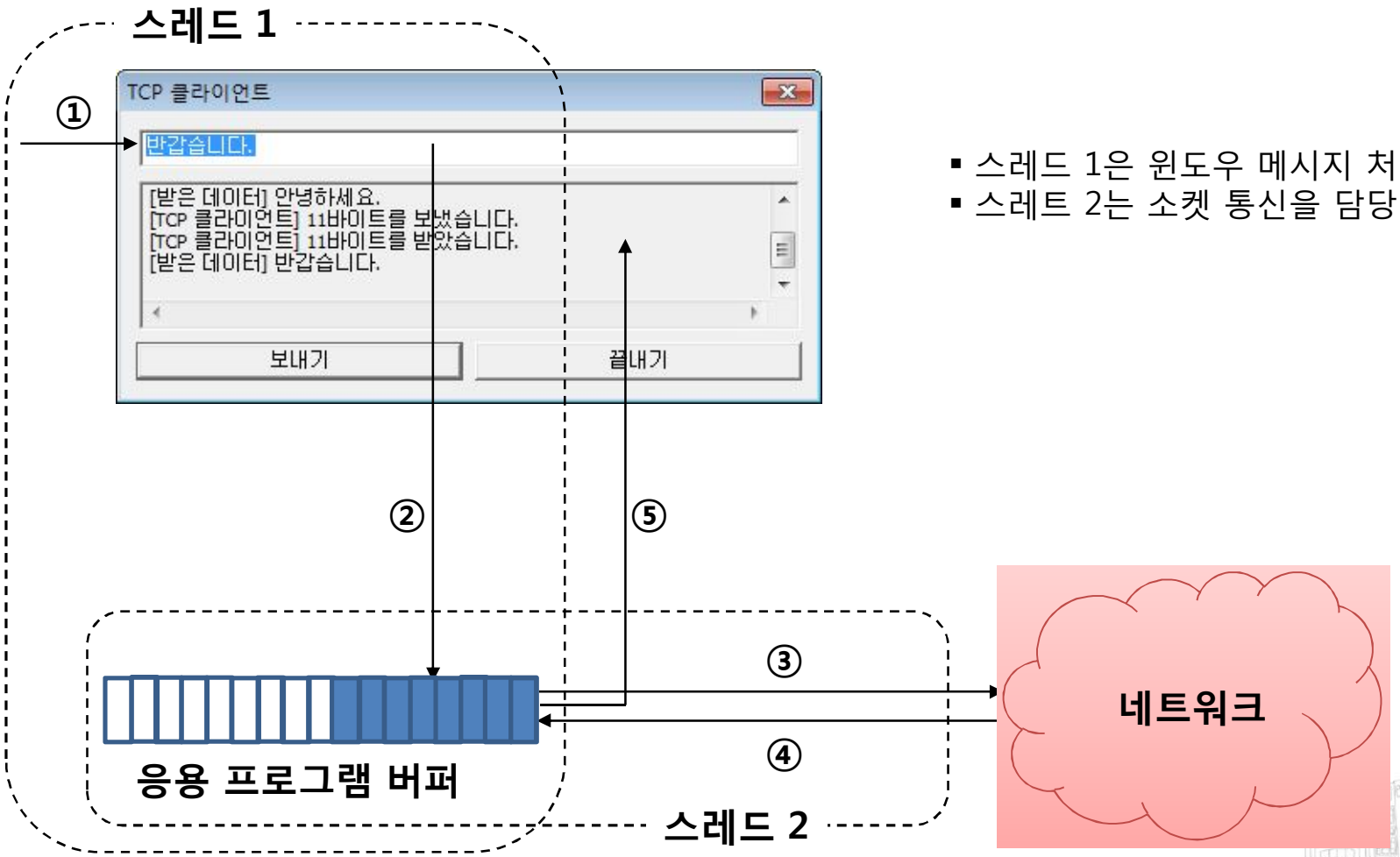
❖ 대화상자 기반 소켓 응용 프로그램 구조

- 일반 GUI 소켓 응용 프로그램과 동일



대화상자 기반 소켓 응용 프로그램 (2)

❖ 스레드 동기화가 필요한 상황



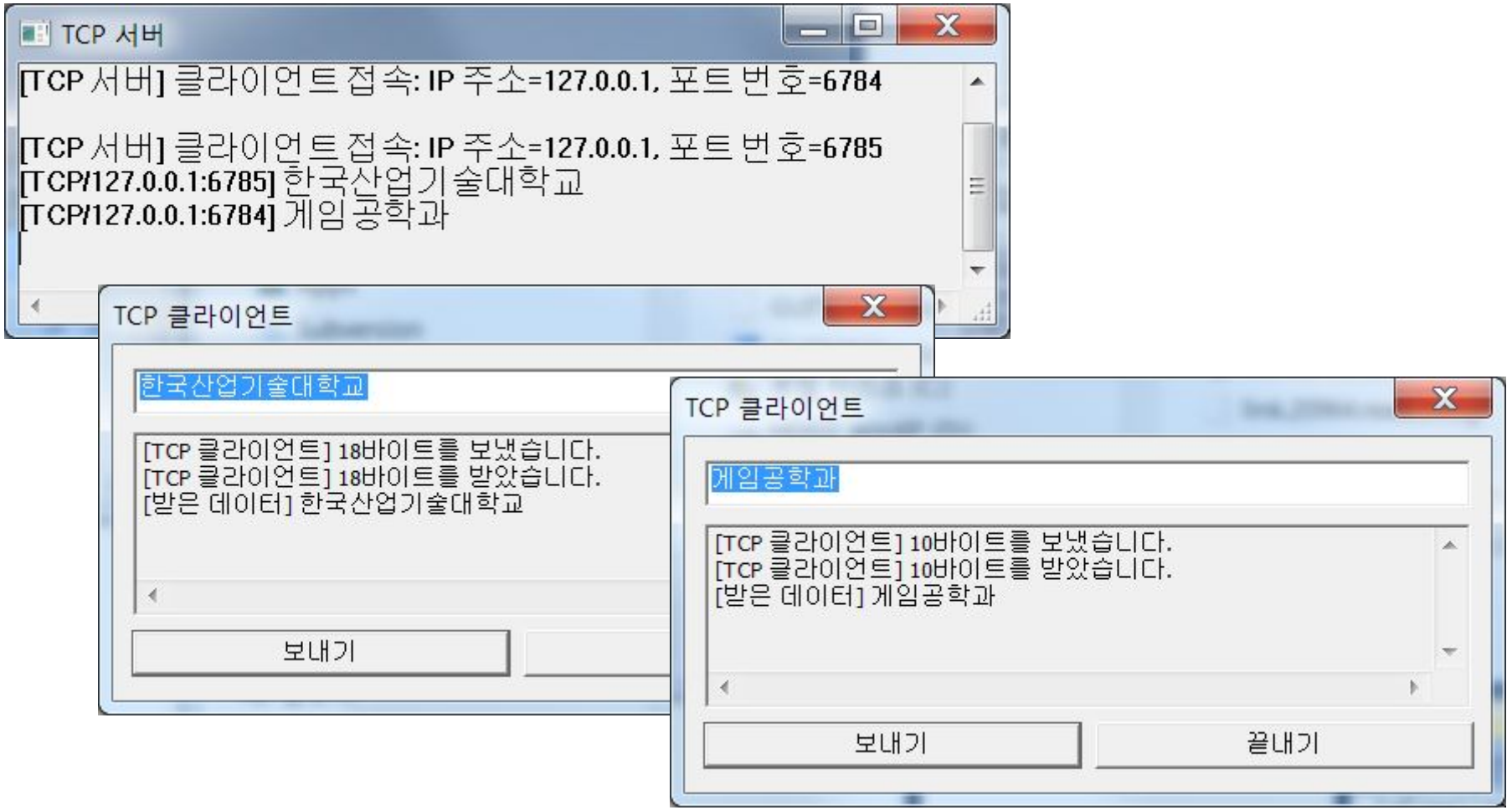
- 스레드 1은 윈도우 메시지 처리
- 스레트 2는 소켓 통신을 담당

대화상자 기반 소켓 응용 프로그램 (3)

- 응용 프로그램의 동작 순서를 두 스레드와 연관지어 요약하면
  - (스레드 1) 사용자가 편집 컨트롤에 글자를 입력한 후 <보내기> 버튼을 누른다.
  - (스레드 1) 편집 컨트롤에 입력된 문자열을 응용 프로그램 버퍼에 저장한다.
  - (스레드 2) 응용 프로그램 버퍼에 저장된 데이터를 에코(echo) 서버에 보낸다.
  - (스레드 2) 에코 서버가 보낸 데이터를 응용 프로그램 버퍼에 읽어들인다.
  - (스레드 2) 응용 프로그램 버퍼에 저장된 데이터를 편집 컨트롤에 출력한다.

실습 9-4

P341~ ; DlgApp 예제에 소켓 소스를 추가한 경우



산업기술의 최고 명문대학
한국산업기술대학교

Thank You !

oasis01@gmail.com / rhqudtn75@nate.com