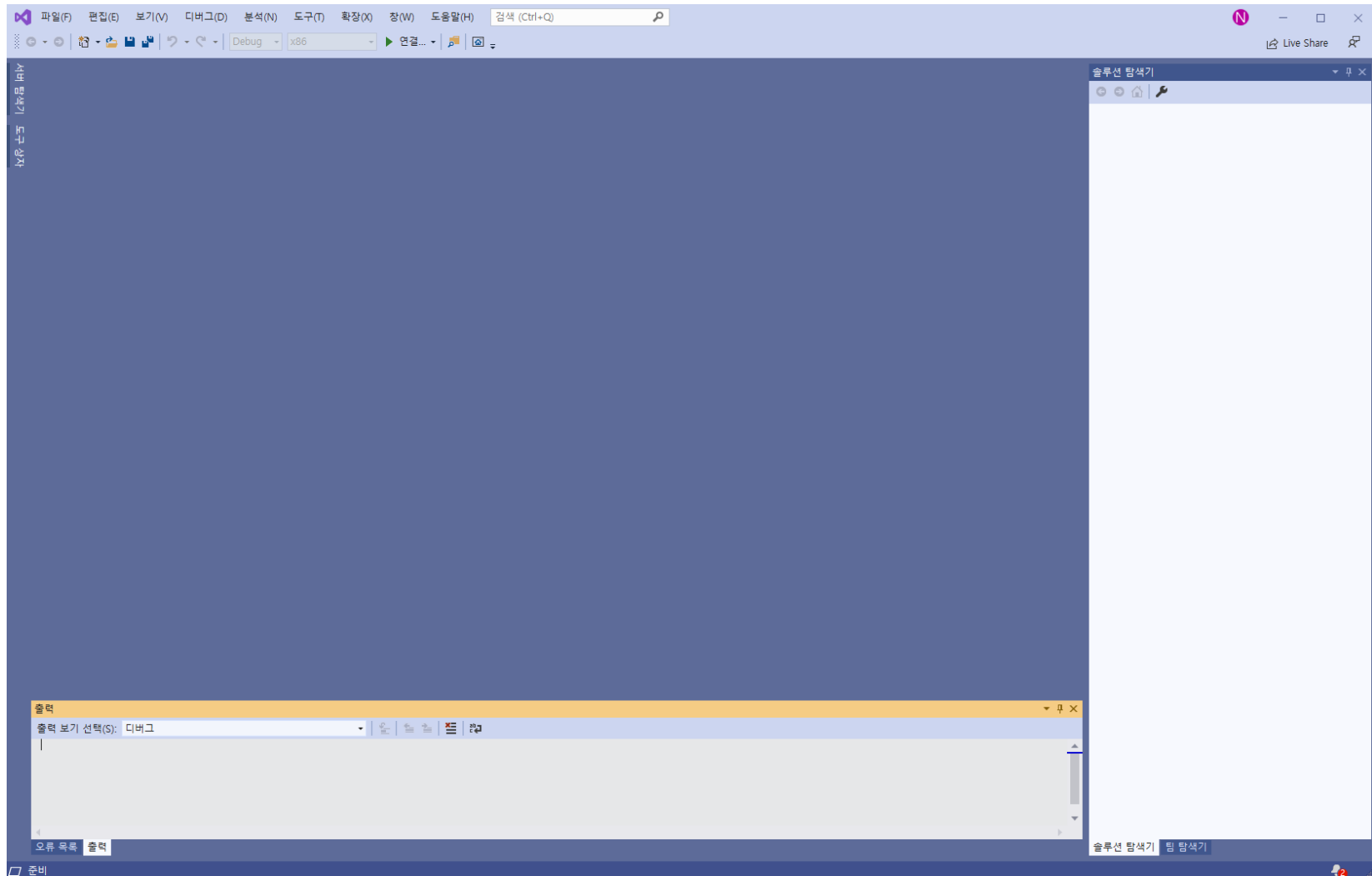


Windows Programming

3주차, 2023

Prof. Kim, Soo Kyun


시작하기 - Visual Studio 2019



새 프로젝트

새 프로젝트 만들기

최근 프로젝트 템플릿(R)

-  Windows 데스크톱 애플리케이션 C++
-  IronPython 애플리케이션 Python
-  Windows 데스크톱 마법사 C++
-  콘솔 앱 C++
-  빈 프로젝트 C++

템플릿 검색(Alt+S)(S)



모두 지우기(C)

C++

Windows

데스크톱



Windows 데스크톱 마법사

마법사를 사용하여 고유한 Windows 앱을 만드세요.

C++ Windows 데스크톱 콘솔 라이브러리



Windows 데스크톱 애플리케이션

Windows에서 실행되는 그래픽 사용자 인터페이스를 사용하는 애플리케이션용 프로젝트입니다.

C++ Windows 데스크톱



공유 항목 프로젝트

공유 항목 프로젝트는 여러 프로젝트 간에 파일을 공유하는 데 사용됩니다.

C++ Windows Android iOS Linux 데스크톱 콘솔
라이브러리 UWP 게임 모바일



비어 있는 앱(유니버설 Windows - C++/CX)

미리 정의된 컨트롤 또는 레이아웃이 없는 단일 페이지 UWP(유니버설 Windows 플랫폼) 앱용 프로젝트입니다.

C++ Windows Xbox UWP 데스크톱



MFC 앱

Windows에서 실행되는 복잡한 사용자 인터페이스를 포함하는 앱을 빌드하세요.

C++ Windows 데스크톱

다음(N)

새 프로젝트 구성

×

새 프로젝트 구성

Windows 데스크톱 애플리케이션 C++ Windows 데스크톱

프로젝트 이름(N)
Prac00

위치(L)
H:\₩내 드라이브₩[1학부3]컴퓨터그래픽스₩1주차₩ ...

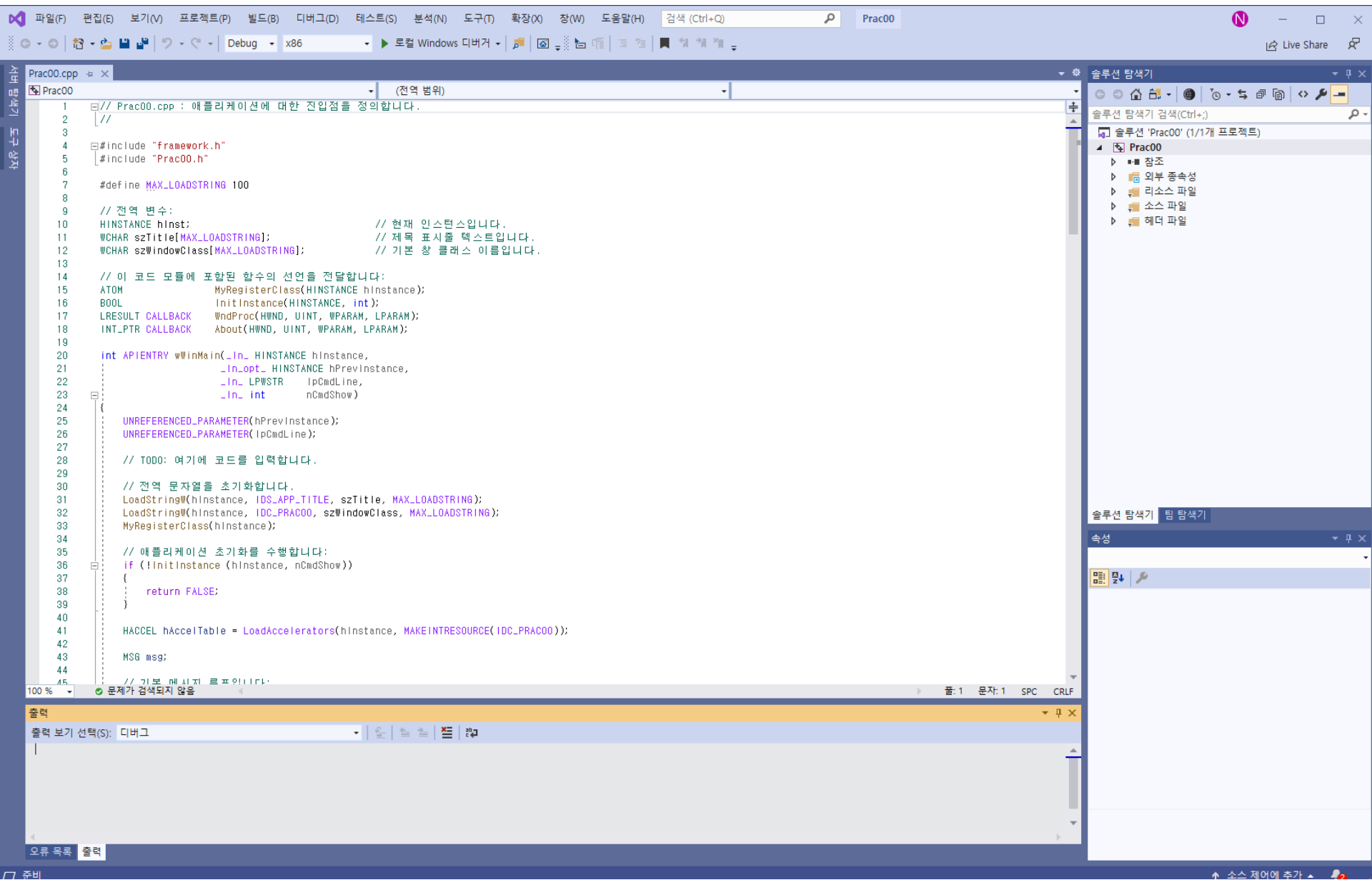
솔루션 이름(M) ⓘ
Prac00

☐ 솔루션 및 프로젝트를 같은 디렉터리에 배치(D)

뒤로(B)

만들기(C)

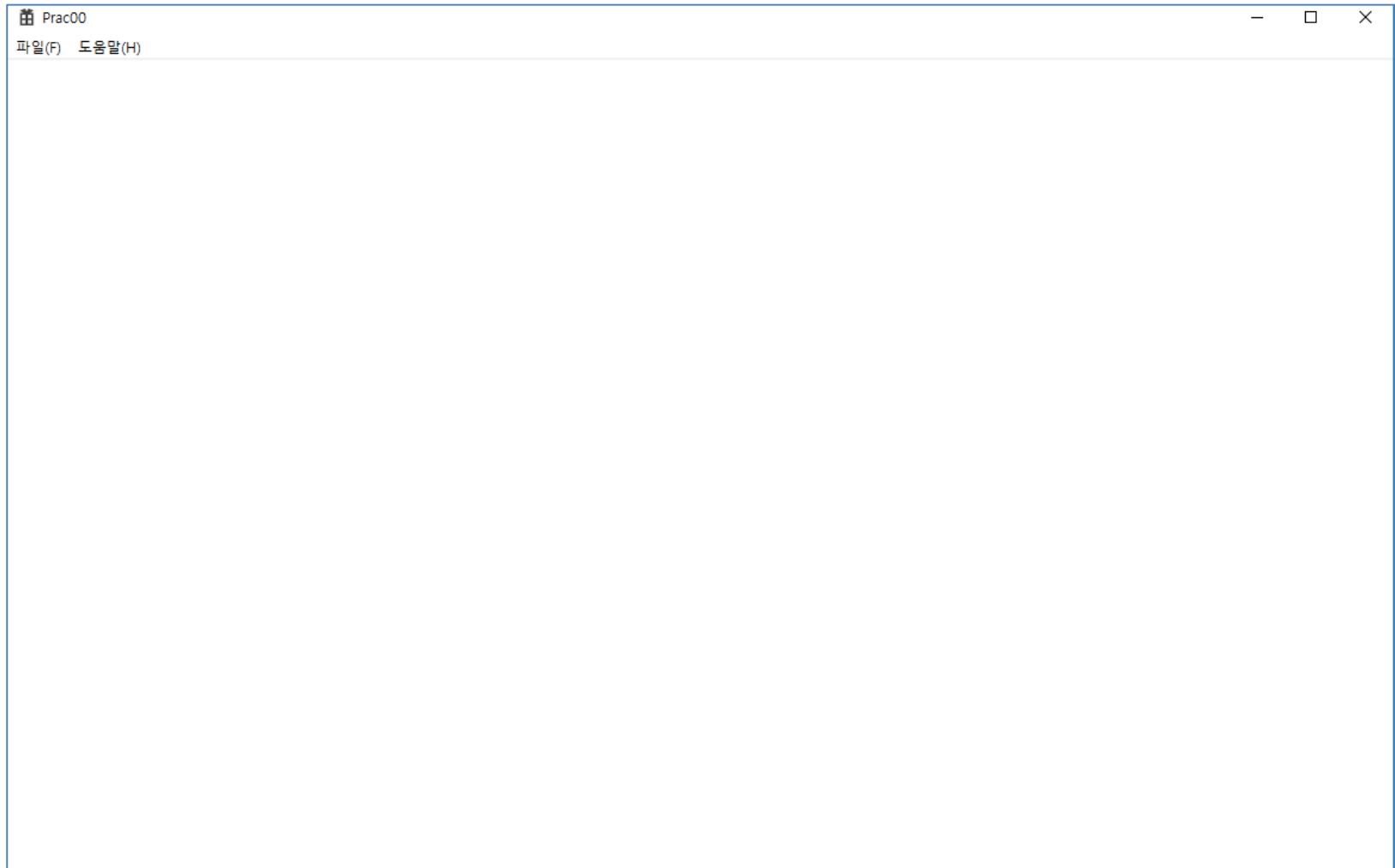
“Prac00” 솔루션



실행

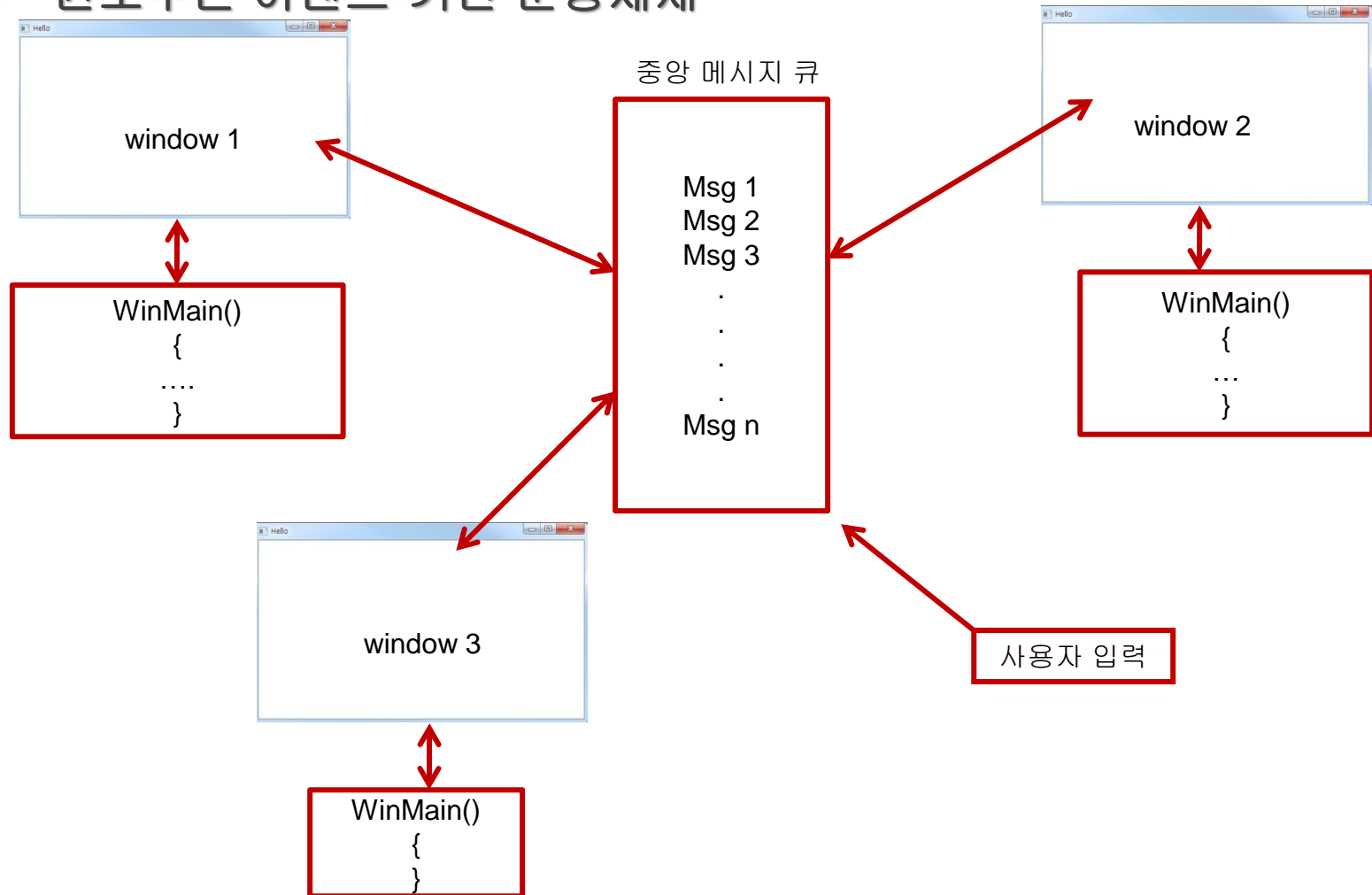
- 디버깅 시작 → F5
- 디버깅하지 않고 시작 → Ctrl + F5
- 디버깅 툴

실행 화면 (1)



Windows 프로그래밍의 기본

- 윈도우는 이벤트-기반 운영체제



“Prac00.cpp”

```
#include "framework.h"
```

```
#include "Prac00.h"
```

```
HWND MyWND = 0;
```

```
LRESULT CALLBACK WndProc( HWND, UINT, WPARAM, LPARAM );
```

```
int APIENTRY wWinMain(_In_ HINSTANCE hInstance, _In_opt_ HINSTANCE  
hPrevInstance, _In_ LPWSTR lpCmdLine, _In_ int nCmdShow)
```

```
{  
    // Create the main window
```

```
    // Main message loop
```

```
}
```

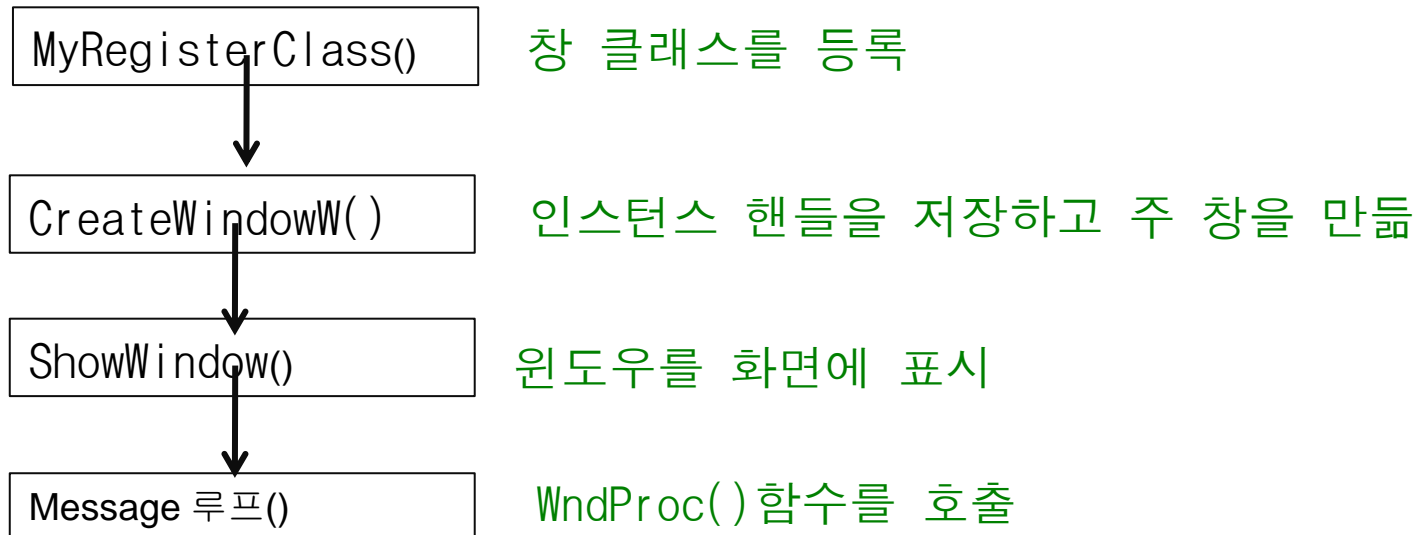
```
LRESULT CALLBACK WndProc(HWND hWnd, UINT msg, WPARAM wParam, LPARAM  
    lParam )
```

```
{
```

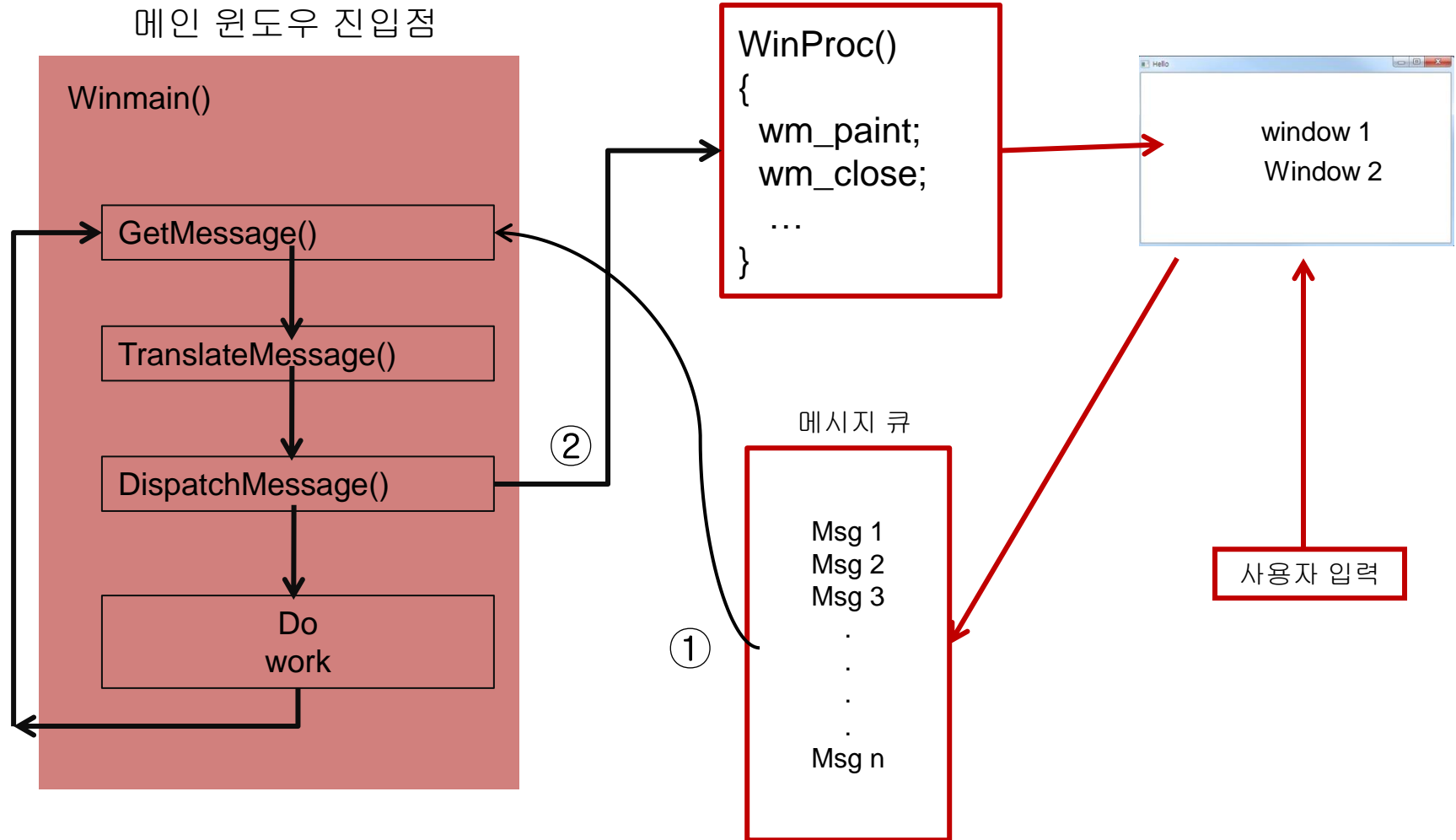
```
}
```

윈도우 생성 과정

②



메인 이벤트 루프



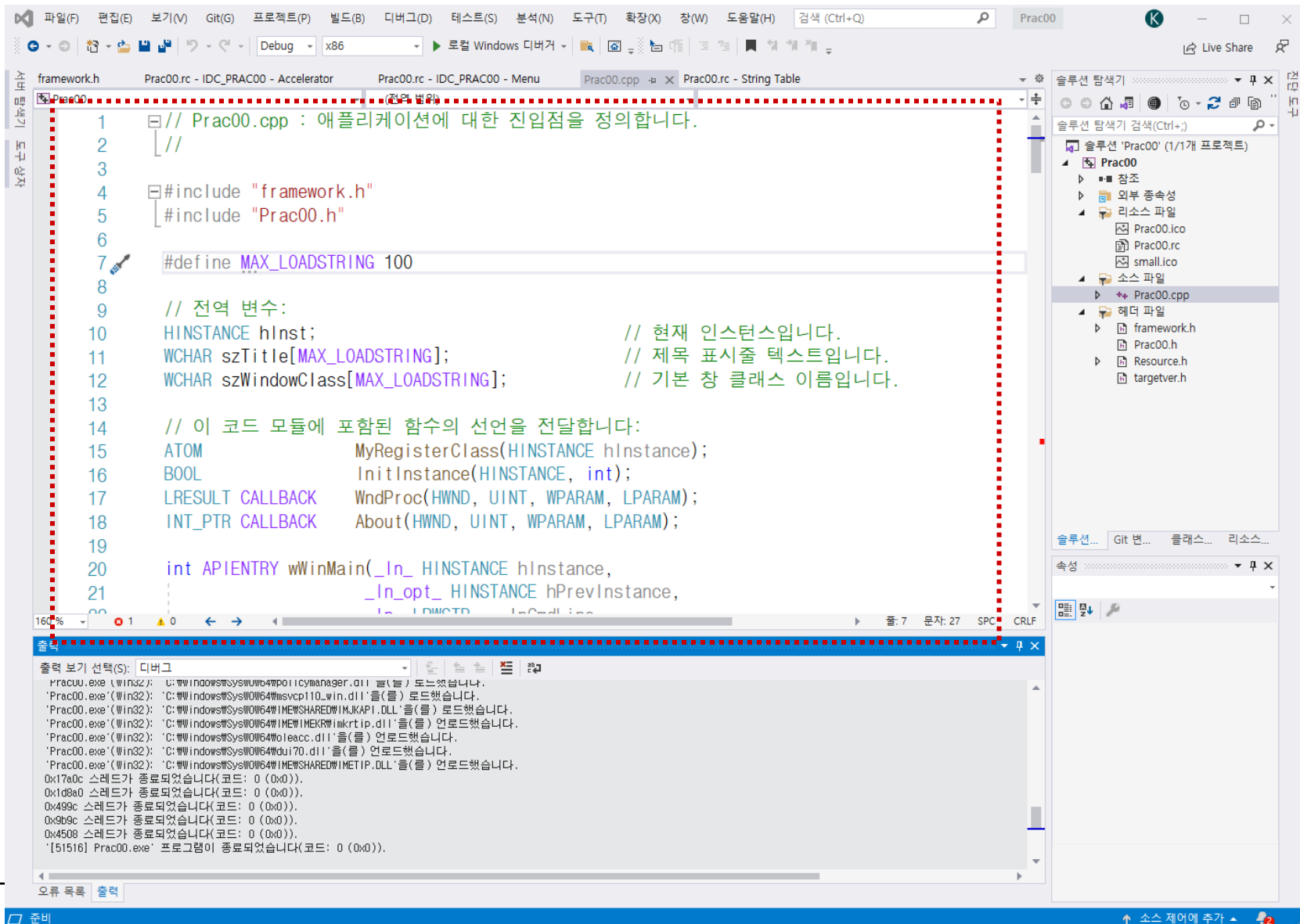
“Prac00.cpp”의 구조

```
#include "framework.h"
#include "Prac00.h"
#define MAX_LOADSTRING 100
// Global Variables:
HINSTANCE hInst;                // current instance
TCHAR szTitle[MAX_LOADSTRING]; // The title bar text
TCHAR szWindowClass[MAX_LOADSTRING]; // the main window class name

// Forward declarations of functions included in this code module:
ATOM MyRegisterClass(HINSTANCE hInstance);
BOOL InitInstance(HINSTANCE, int);
LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);
INT_PTR CALLBACK About(HWND, UINT, WPARAM, LPARAM);

int APIENTRY wWinMain(_In_ HINSTANCE hInstance,
                     _In_opt_ HINSTANCE hPrevInstance,
                     _In_ LPWSTR lpCmdLine,
                     _In_ int nCmdShow)
```

Includes, Global Variables, and Prototypes



The screenshot displays the Visual Studio IDE with a C++ project named 'Prac00'. The main editor window shows the file 'Prac00.cpp' with the following code:

```
1 // Prac00.cpp : 애플리케이션에 대한 진입점을 정의합니다.
2 //
3
4 #include "framework.h"
5 #include "Prac00.h"
6
7 #define MAX_LOADSTRING 100
8
9 // 전역 변수:
10 HINSTANCE hInst; // 현재 인스턴스입니다.
11 WCHAR szTitle[MAX_LOADSTRING]; // 제목 표시줄 텍스트입니다.
12 WCHAR szWindowClass[MAX_LOADSTRING]; // 기본 창 클래스 이름입니다.
13
14 // 이 코드 모듈에 포함된 함수의 선언을 전달합니다:
15 ATOM MyRegisterClass(HINSTANCE hInstance);
16 BOOL InitInstance(HINSTANCE, int);
17 LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);
18 INT_PTR CALLBACK About(HWND, UINT, WPARAM, LPARAM);
19
20 int APIENTRY wWinMain(_In_ HINSTANCE hInstance,
21 _In_opt_ HINSTANCE hPrevInstance,
```

The right sidebar shows the 'Solution Explorer' with the project structure:

- 솔루션 탐색기 (Solution Explorer)
- 솔루션 탐색기 검색 (Ctrl+;) (Search Solution Explorer)
- 솔루션 탐색기 (1/1개 프로젝트) (Solution Explorer (1/1 project))
- Prac00
 - 참조 (References)
 - 외부 종속성 (External Dependencies)
 - 리소스 파일 (Resource Files)
 - Prac00.ico
 - Prac00.rc
 - small.ico
 - 소스 파일 (Source Files)
 - Prac00.cpp
 - 헤더 파일 (Header Files)
 - framework.h
 - Prac00.h
 - Resource.h
 - targetver.h

The bottom status bar shows '준비' (Ready).

WinMain() - Creation



```
Proc00.cpp
Proc00 (전역 범위)
19
20 int APIENTRY wWinMain(_In_ HINSTANCE hInstance,
21                      _In_opt_ HINSTANCE hPrevInstance,
22                      _In_ LPWSTR lpCmdLine,
23                      _In_ int nCmdShow)
24 {
25     UNREFERENCED_PARAMETER(hPrevInstance);
26     UNREFERENCED_PARAMETER(lpCmdLine);
27
28     // TODO: 여기에 코드를 입력합니다.
29
30     // 전역 문자열을 초기화합니다.
31     LoadStringW(hInstance, IDS_APP_TITLE, szTitle, MAX_LOADSTRING);
32     LoadStringW(hInstance, IDC_PROCC00, szWindowClass, MAX_LOADSTRING);
33     MyRegisterClass(hInstance);
34
35     // 응용 프로그램 초기화를 수행합니다.
36     if (!InitInstance(hInstance, nCmdShow))
37     {
38         return FALSE;
39     }
40
41     HACCEL hAccelTable = LoadAccelerators(hInstance, MAKEINTRESOURCE(IDC_PROCC00));
42
43     MSG msg;
44
```

지난 시간 이어서...

MSDN - WinMain()

The screenshot shows a web browser window displaying the MSDN website. The address bar shows the URL `http://msdn.microsoft.com/en-us/library/windows/desktop/ff381406(v=vs.85).aspx`. The page title is "WinMain: The Application Entry Point". The left sidebar contains a navigation menu with the following items: "Learn to Program for Windows in C++", "Introduction to Windows Programming in C++", "Prepare Your Development Environment", "Windows Coding Conventions", "Working with Strings", "What Is a Window?", and "WinMain: The Application Entry Point". The main content area has the heading "WinMain: The Application Entry Point" and a paragraph explaining that every Windows program includes an entry-point function named either **WinMain** or **wWinMain**. Below this is a code block showing the signature of **wWinMain**:

```
int WINAPI wWinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PWSTR pCmdLine, int nCmdShow)
```

The four parameters are:

- hInstance* is something called a "handle to an instance" or "handle to a module." The operating system uses this value to identify the executable (EXE) when it is loaded in memory. The instance handle is needed for certain Windows functions — for example, to load icons or bitmaps.
- hPrevInstance* has no meaning. It was used in 16-bit Windows, but is now always zero.
- pCmdLine* contains the command-line arguments as a Unicode string.
- nCmdShow* is a flag that says whether the main application window will be minimized, maximized, or shown normally.

The function returns an **int** value. The return value is not used by the operating system, but you can use the return value to convey a status code to some other program that you write.

MSDN - WinMain()

▸ Learn to Program for Windows in C++

▀ Introduction to Windows Programming in C++

Prepare Your Development Environment

Windows Coding Conventions

Working with Strings

What Is a Window?

WinMain: The Application Entry Point

WinMain: The Application Entry Point

Every Windows program includes an entry-point function that is named either **WinMain** or **wWinMain**. Here is the signature for **wWinMain**.

```
int WINAPI wWinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PWSTR pCmdLine, int
```

Copy

<

>

hInstance : 프로그램의 인스턴스 핸들

(운영 체제는 이 값을 사용하여 실행 파일 또는 EXE가 메모리에 로드될 때 이를 식별)

hPrevInstance : 앞에 실행된 현재 프로그램의 인스턴스 핸들

(16비트 Windows에서 사용되었지만 지금은 항상 0)

lpCmdLine: pCmdLine은 명령줄 인수를 유니코드 문자열로 포함함

nCmdShow : 기본 애플리케이션 창을 최소화할지, 최대화할지 또는 정상적으로 표시할지를 나타내는 플래그임

WinMain() - Creation

The image shows a C++ IDE window titled 'Proc00.cpp' with a 'Proc00' tab. The code is in the '전역 범위' (Global Scope) and defines the `WinMain` function. A red arrow points to line 33, `MyRegisterClass(hInstance);`. To the right, a flowchart illustrates the sequence of operations: `MyRegisterClass()` (창 클래스를 등록), `CreateWindowW()` (인스턴스 핸들을 저장하고 주 창을 만들), `ShowWindow()` (윈도우를 화면에 표시), and `Message 루프()` (`WndProc()` 함수를 호출).

```
19
20 int APIENTRY wWinMain(_In_ HINSTANCE hInstance,
21                      _In_opt_ HINSTANCE hPrevInstance,
22                      _In_ LPWSTR lpCmdLine,
23                      _In_ int nCmdShow)
24 {
25     UNREFERENCED_PARAMETER(hPrevInstance);
26     UNREFERENCED_PARAMETER(lpCmdLine);
27
28     // TODO: 여기에 코드를 입력합니다.
29
30     // 전역 문자열을 초기화합니다.
31     LoadStringW(hInstance, IDS_APP_TITLE, szTitle, MAX_LOADSTRING);
32     LoadStringW(hInstance, IDC_PROCC00, szWindowClass, MAX_LOADSTRING);
33     MyRegisterClass(hInstance);
34
35     // 응용 프로그램 초기화를 수행합니다.
36     if (!InitInstance(hInstance, nCmdShow))
37     {
38         return FALSE;
39     }
40
41     HACCEL hAccelTable = LoadAccelerators(hInstance, MAKEINTRESOURCE(IDC_PROCC00));
42
43     MSG msg;
44
```

Flowchart steps:

- MyRegisterClass() → 창 클래스를 등록
- CreateWindowW() → 인스턴스 핸들을 저장하고 주 창을 만들
- ShowWindow() → 윈도우를 화면에 표시
- Message 루프() → WndProc() 함수를 호출

Registering the Window Class

```
Prac00.cpp  x
Prac00      (전역 범위)

61  // 함수: MyRegisterClass()
62  //
63  // 용도: 창 클래스를 등록합니다.
64  //
65  ATOM MyRegisterClass(HINSTANCE hInstance)
66  {
67      WNDCLASSEXW wcex;
68
69      wcex.cbSize = sizeof(WNDCLASSEX);
70
71      wcex.style      = CS_HREDRAW | CS_VREDRAW;
72      wcex.lpfnWndProc = WndProc;
73      wcex.cbClsExtra = 0;
74      wcex.cbWndExtra = 0;
75      wcex.hInstance  = hInstance;
76      wcex.hIcon       = LoadIcon(hInstance, MAKEINTRESOURCE(IDI_PRAC00));
77      wcex.hCursor     = LoadCursor(NULL, IDC_ARROW);
78      wcex.hbrBackground = (HBRUSH)(COLOR_WINDOW+1);
79      wcex.lpszMenuName = MAKEINTRESOURCEW(IDC_PRAC00);
80      wcex.lpszClassName = szWindowClass;
81      wcex.hIconSm      = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(IDI_SMALL));
82
83      return RegisterClassExW(&wcex);
84  }
```

MSDN - WNDCLASSEX

The screenshot shows a web browser window displaying the MSDN documentation for the `WNDCLASSEX` structure. The browser's address bar shows the URL `http://msdn.microsoft.com/en-us/library/windows/desktop/ms633577(v=vs.85).aspx`. The page header includes the Windows logo, "Dev Center - Desktop", and navigation links like "DASHBOARD", "GET STARTED", "DESIGN", "DEVELOP", "TEST AND DEPLOY", and "CERTIFY". A search bar is also present. The left sidebar contains a navigation tree with categories like "Desktop App UI", "Windows and Messages", "Window Classes", and "Window Class Reference". Under "Window Class Reference", "Window Class Structures" is expanded, showing "WNDCLASS" and "WNDCLASSEX". The main content area is titled "WNDCLASSEX structure" and contains the following text: "Contains window class information. It is used with the [RegisterClassEx](#) and [GetClassInfoEx](#) functions." and "The **WNDCLASSEX** structure is similar to the [WNDCLASS](#) structure. There are two differences. **WNDCLASSEX** includes the **cbSize** member, which specifies the size of the structure, and the **hIconSm** member, which contains a handle to a small icon associated with the window class." Below this text is a "Syntax" section with a C++ code block. The code block is titled "C++" and has a "Copy" button. The code defines the `tagWNDCLASSEX` structure with the following members: `UINT cbSize;`, `UINT style;`, `WNDPROC lpfnWndProc;`, `int cbClsExtra;`, `int cbWndExtra;`, `HINSTANCE hInstance;`, `HICON hIcon;`, `HCURSOR hCursor;`, and `HBRUSH hbrBackground;`.

Windows Dev Center - Desktop

DASHBOARD GET STARTED DESIGN DEVELOP TEST AND DEPLOY CERTIFY

Desktop technologies Server and system API index Samples Community

Sign in

Desktop App UI

Windows and Messages

Window Classes

Window Class Reference

Window Class Structures

WNDCLASS

WNDCLASSEX

WNDCLASSEX structure

Contains window class information. It is used with the [RegisterClassEx](#) and [GetClassInfoEx](#) functions.

The **WNDCLASSEX** structure is similar to the [WNDCLASS](#) structure. There are two differences. **WNDCLASSEX** includes the **cbSize** member, which specifies the size of the structure, and the **hIconSm** member, which contains a handle to a small icon associated with the window class.

Syntax

```
C++  
typedef struct tagWNDCLASSEX {  
    UINT      cbSize;  
    UINT      style;  
    WNDPROC   lpfnWndProc;  
    int       cbClsExtra;  
    int       cbWndExtra;  
    HINSTANCE hInstance;  
    HICON     hIcon;  
    HCURSOR   hCursor;  
    HBRUSH    hbrBackground;  
};
```

Copy

WinMain() - Creation

The image shows a C++ IDE window titled 'Proc00.cpp' with a tab icon. The code editor displays the `WinMain` function. To the right of the code, a flowchart illustrates the sequence of operations: `MyRegisterClass()` (창 클래스를 등록), `CreateWindowW()` (인스턴스 핸들을 저장하고 주 창을 만들), `ShowWindow()` (윈도우를 화면에 표시), and `Message 루프()` (`WndProc()` 함수를 호출). Red arrows point from the code lines to the flowchart boxes: one from line 33 to `MyRegisterClass()` and another from line 36 to `if (!InitInstance(hInstance, nCmdShow))`.

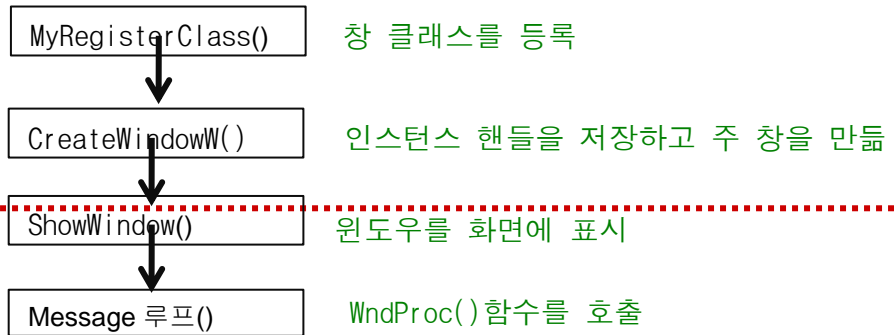
```
19
20 int APIENTRY wWinMain(_In_ HINSTANCE hInstance,
21                      _In_opt_ HINSTANCE hPrevInstance,
22                      _In_ LPWSTR lpCmdLine,
23                      _In_ int nCmdShow)
24 {
25     UNREFERENCED_PARAMETER(hPrevInstance);
26     UNREFERENCED_PARAMETER(lpCmdLine);
27
28     // TODO: 여기에 코드를 입력합니다.
29
30     // 전역 문자열을 초기화합니다.
31     LoadStringW(hInstance, IDS_APP_TITLE, szTitle, MAX_LOADSTRING);
32     LoadStringW(hInstance, IDC_PROCC00, szWindowClass, MAX_LOADSTRING);
33     MyRegisterClass(hInstance);
34
35     // 응용 프로그램 초기화를 수행합니다.
36     if (!InitInstance(hInstance, nCmdShow))
37     {
38         return FALSE;
39     }
40
41     HACCEL hAccelTable = LoadAccelerators(hInstance, MAKEINTRESOURCE(IDC_PROCC00));
42
43     MSG msg;
44
```

Flowchart steps:

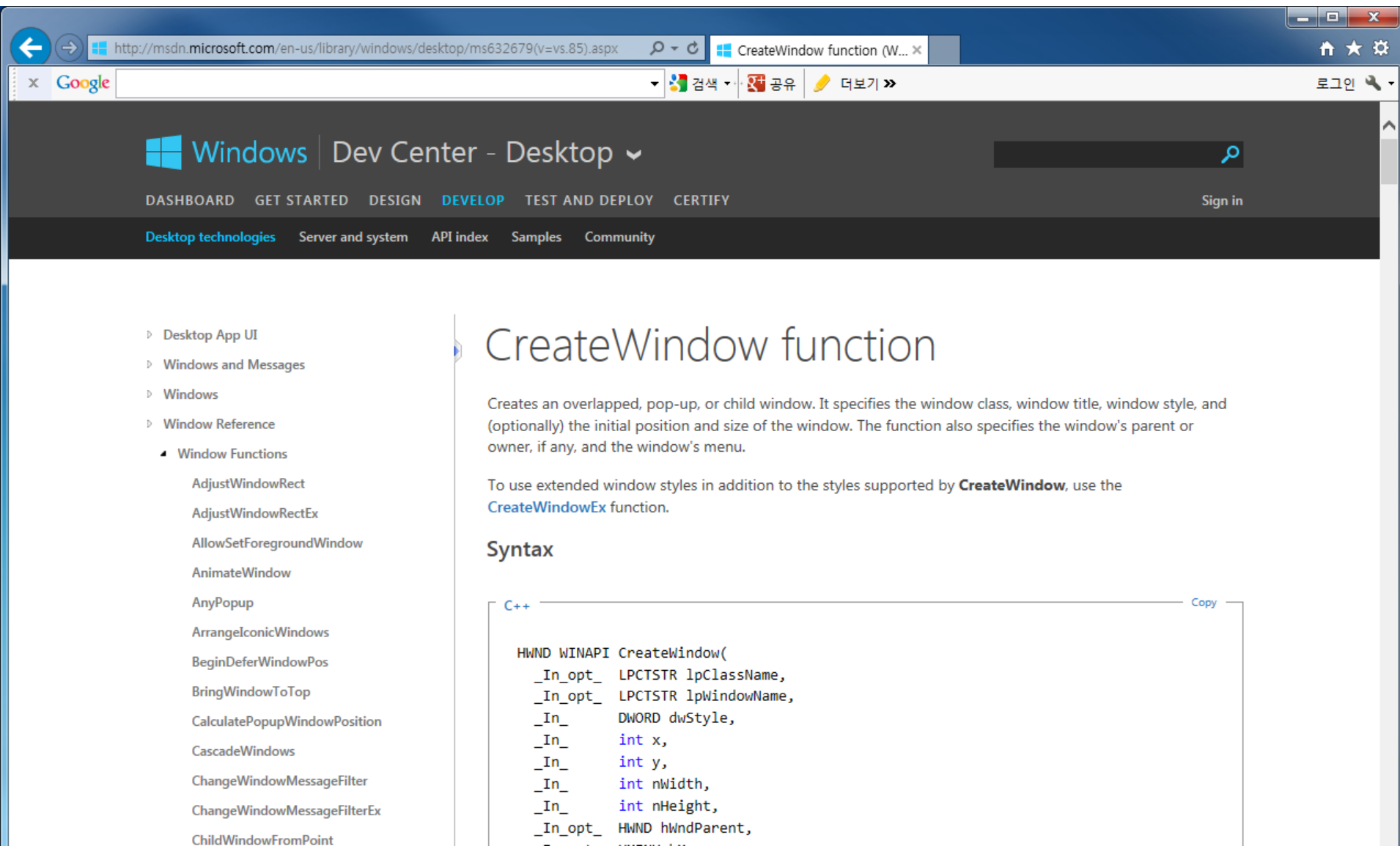
- MyRegisterClass() → 창 클래스를 등록
- CreateWindowW() → 인스턴스 핸들을 저장하고 주 창을 만들
- ShowWindow() → 윈도우를 화면에 표시
- Message 루프() → WndProc() 함수를 호출

Creating a Window

```
Proc00.cpp  Proc00  (전역 범위)
91  // 설명:
92  //
93  // 이 함수를 통해 인스턴스 핸들을 전역 변수에 저장하고
94  // 주 프로그램 창을 만든 다음 표시합니다.
95  //
96  BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)
97  {
98      hInst = hInstance; // 인스턴스 핸들을 전역 변수에 저장합니다.
99
100     HWND hWnd = CreateWindowW(szWindowClass, szTitle, WS_OVERLAPPEDWINDOW,
101                             CW_USEDEFAULT, 0, CW_USEDEFAULT, 0, nullptr, nullptr, hInstance, nullptr);
102
103     if (!hWnd)
104     {
105         return FALSE;
106     }
107
108     ShowWindow(hWnd, nCmdShow);
109     UpdateWindow(hWnd);
110
111     return TRUE;
112 }
113
```



MSDN - CreateWindow()



The screenshot shows the MSDN website interface. The browser's address bar displays the URL: `http://msdn.microsoft.com/en-us/library/windows/desktop/ms632679(v=vs.85).aspx`. The page title is "CreateWindow function (W...". The website header includes the Windows logo, "Dev Center - Desktop", and navigation links: DASHBOARD, GET STARTED, DESIGN, DEVELOP, TEST AND DEPLOY, CERTIFY. A search bar and a "Sign in" link are also present. The left sidebar contains a tree view of the "Window Functions" category, listing various functions like AdjustWindowRect, AdjustWindowRectEx, AllowSetForegroundWindow, AnimateWindow, AnyPopup, ArrangeIconicWindows, BeginDeferWindowPos, BringWindowToTop, CalculatePopupWindowPosition, CascadeWindows, ChangeWindowMessageFilter, ChangeWindowMessageFilterEx, and ChildWindowFromPoint. The main content area is titled "CreateWindow function" and provides a description: "Creates an overlapped, pop-up, or child window. It specifies the window class, window title, window style, and (optionally) the initial position and size of the window. The function also specifies the window's parent or owner, if any, and the window's menu." Below the description, it states: "To use extended window styles in addition to the styles supported by **CreateWindow**, use the [CreateWindowEx](#) function." The "Syntax" section shows the C++ signature for the function:

```
HWND WINAPI CreateWindow(  
    _In_opt_ LPCTSTR lpClassName,  
    _In_opt_ LPCTSTR lpWindowName,  
    _In_     DWORD dwStyle,  
    _In_     int x,  
    _In_     int y,  
    _In_     int nWidth,  
    _In_     int nHeight,  
    _In_opt_ HWND hWndParent,
```

MSDN - CreateWindow()

- Desktop App UI
- Windows and Messages
- Windows
- Window Reference
 - Window Functions
 - AdjustWindowRect
 - AdjustWindowRectEx
 - AllowSetForegroundWindow
 - AnimateWindow

CreateWindow function

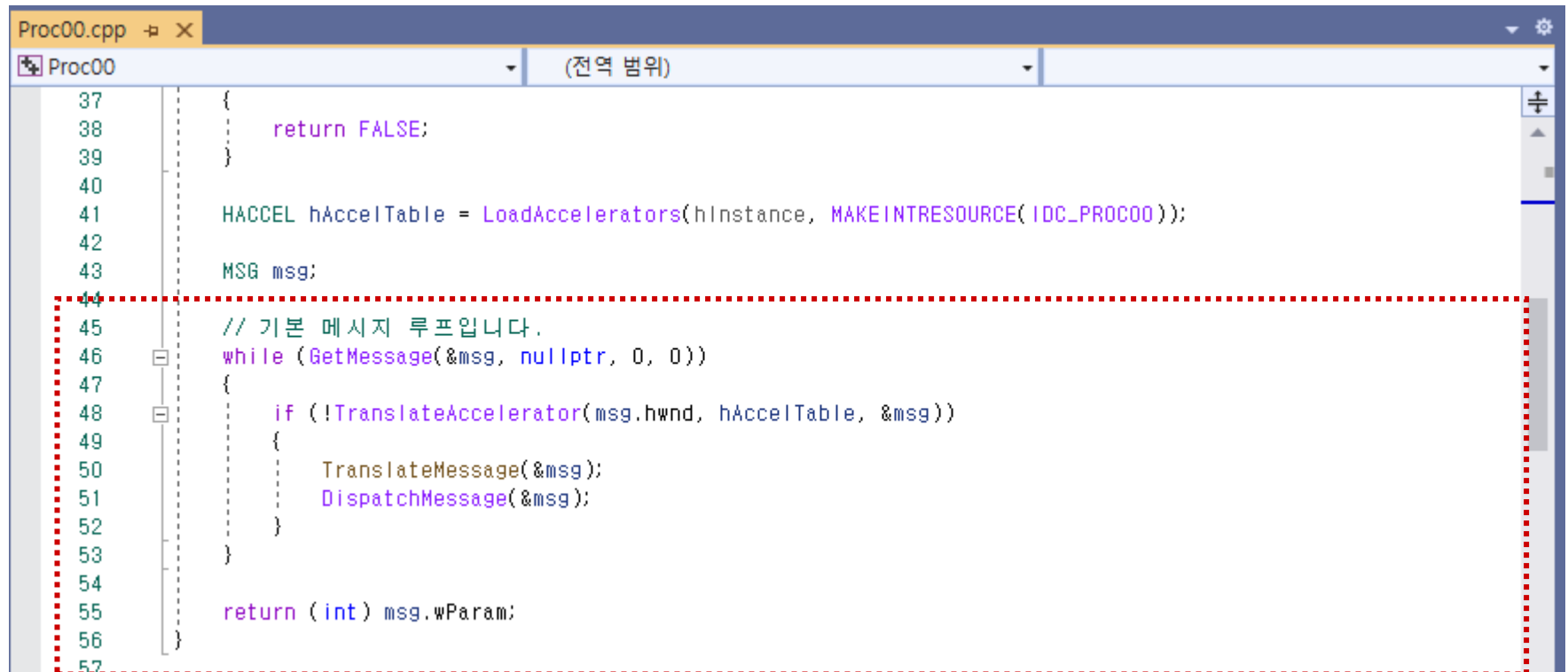
Creates an overlapped, pop-up, or child window. It specifies the window class, window title, window style, and (optionally) the initial position and size of the window. The function also specifies the window's parent or owner, if any, and the window's menu.

To use extended window styles in addition to the styles supported by **CreateWindow**, use the [CreateWindowEx](#) function.

Syntax

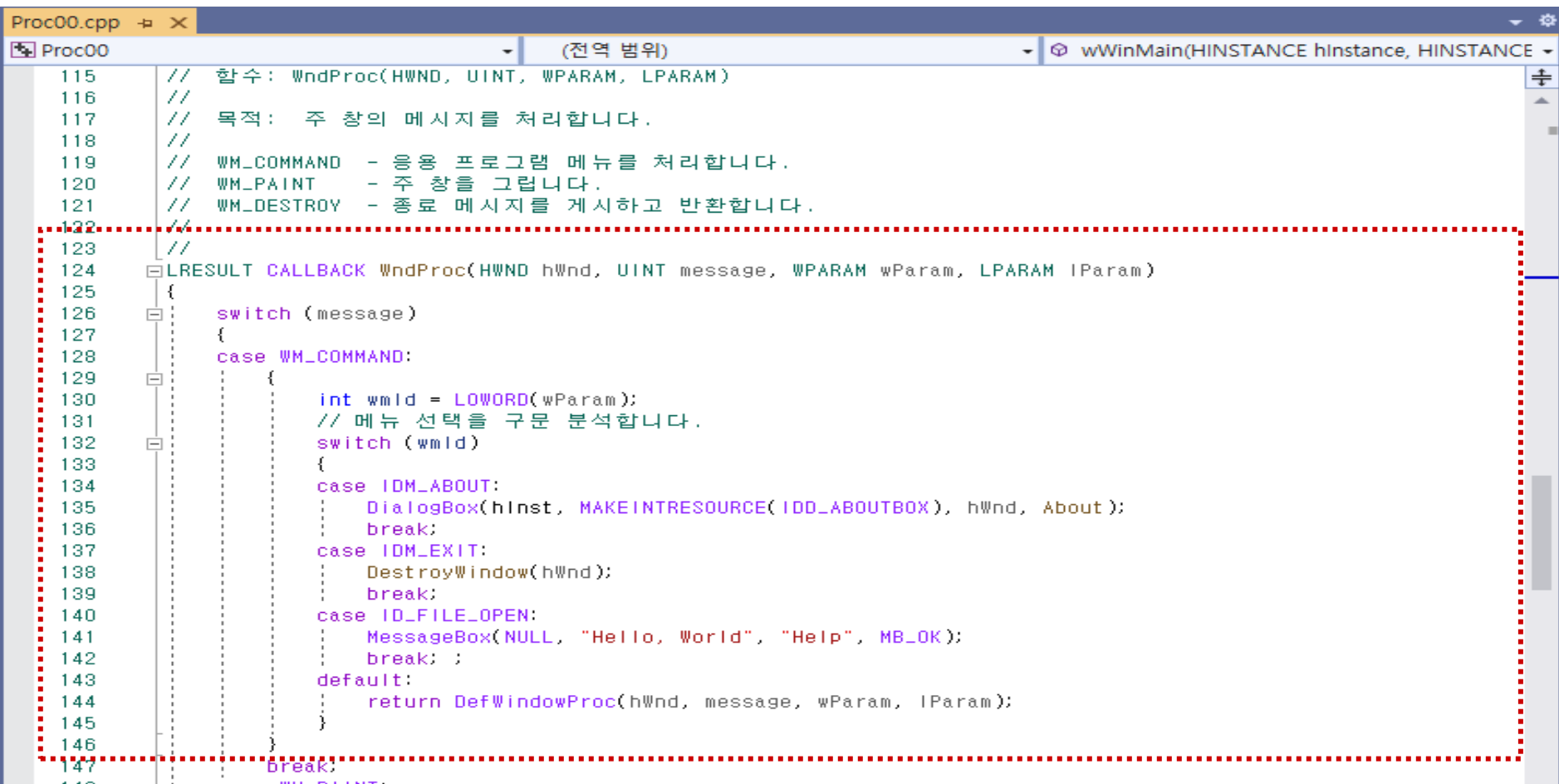
```
HWND WINAPI CreateWindow(LPCWSTR lpClassName, //윈도우 클래스 이름
    LPCWSTR lpWindowName, //윈도우 인스턴스 캡션 명(타이틀)
    DWORD dwStyle, //윈도우 스타일
    int X, //좌상단 x좌표
    int Y, //좌상단 y좌표
    int nWidth, //너비
    int nHeight, //높이
    HWND hWndParent, //부모 윈도우 핸들
    HMENU hMenu, //메뉴 핸들
    HINSTANCE hInstance, //모듈의 인스턴스 핸들
    LPVOID lpParam); //생성할 때 전달할 인자
```


WinMain() - Message Loop



```
Proc00.cpp -> X
Proc00 (전역 범위)
37 {
38     return FALSE;
39 }
40
41 HACCEL hAccelTable = LoadAccelerators(hInstance, MAKEINTRESOURCE(IDC_PROC00));
42
43 MSG msg;
44
45 // 기본 메시지 루프입니다.
46 while (GetMessage(&msg, nullptr, 0, 0))
47 {
48     if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))
49     {
50         TranslateMessage(&msg);
51         DispatchMessage(&msg);
52     }
53 }
54
55 return (int) msg.wParam;
56 }
57
```

The Window Procedure (1)



```
115 // 함수: WndProc(HWND, UINT, WPARAM, LPARAM)
116 //
117 // 목적: 주 창의 메시지를 처리합니다.
118 //
119 // WM_COMMAND - 응용 프로그램 메뉴를 처리합니다.
120 // WM_PAINT - 주 창을 그립니다.
121 // WM_DESTROY - 종료 메시지를 게시하고 반환합니다.
122 //
123 //
124 LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)
125 {
126     switch (message)
127     {
128     case WM_COMMAND:
129     {
130         int wmid = LOWORD(wParam);
131         // 메뉴 선택을 구문 분석합니다.
132         switch (wmid)
133         {
134         case IDM_ABOUT:
135             DialogBox(hInst, MAKEINTRESOURCE(IDD_ABOUTBOX), hWnd, About);
136             break;
137         case IDM_EXIT:
138             DestroyWindow(hWnd);
139             break;
140         case ID_FILE_OPEN:
141             MessageBox(NULL, "Hello, World", "Help", MB_OK);
142             break;
143         default:
144             return DefWindowProc(hWnd, message, wParam, lParam);
145         }
146     }
147     break;
148     case WM_PAINT:
149     {
150         PAINTSTRUCT ps;
151         Paint(hWnd, &ps);
152     }
153     break;
154     default:
155         return DefWindowProc(hWnd, message, wParam, lParam);
156     }
```

LRESULT: 기본적으로 32비트 정수 값을 나타내며, 미리 지정된 형식

CALLBACK: 이벤트가 발생했을 때 윈도우(운영체제)에 의해서 호출되는 것

The Window Procedure (2)

Prac00.ico - Icon [48x48, 8비트, BMP] Prac00.rc - String Table [영어(미국)] Prac00.cpp Prac00.h Prac00.rc - IDC_PRAC00 - Accelerator

Prac00 (전역 범위)

```
141         break;
142     case IDM_FILE_OPEN:
143         MessageBox(NULL, L"Open a file...", L"File Open Menu", MB_OK);
144         break;
145     default:
146         return DefWindowProc(hWnd, message, wParam, lParam);
147     }
148 }
149 break;
150 case WM_PAINT:
151 {
152     PAINTSTRUCT ps;
153     HDC hdc = BeginPaint(hWnd, &ps);
154     // TODO: 여기에 hdc를 사용하는 그리기 코드를 추가합니다...
155     EndPaint(hWnd, &ps);
156 }
157 break;
158 case WM_LBUTTONDOWN:
159     MessageBox(NULL, L"Hello, World", L"Help", MB_OK);
160     break;
161 case WM_KEYDOWN:
162     DestroyWindow(hWnd);
163     break;
164 case WM_DESTROY:
165     PostQuitMessage(0);
166     break;
167 default:
168     return DefWindowProc(hWnd, message, wParam, lParam);
169 }
170 return 0;
171 }
172
173 // 정보 대화 상자의 메시지 처리기입니다.
```

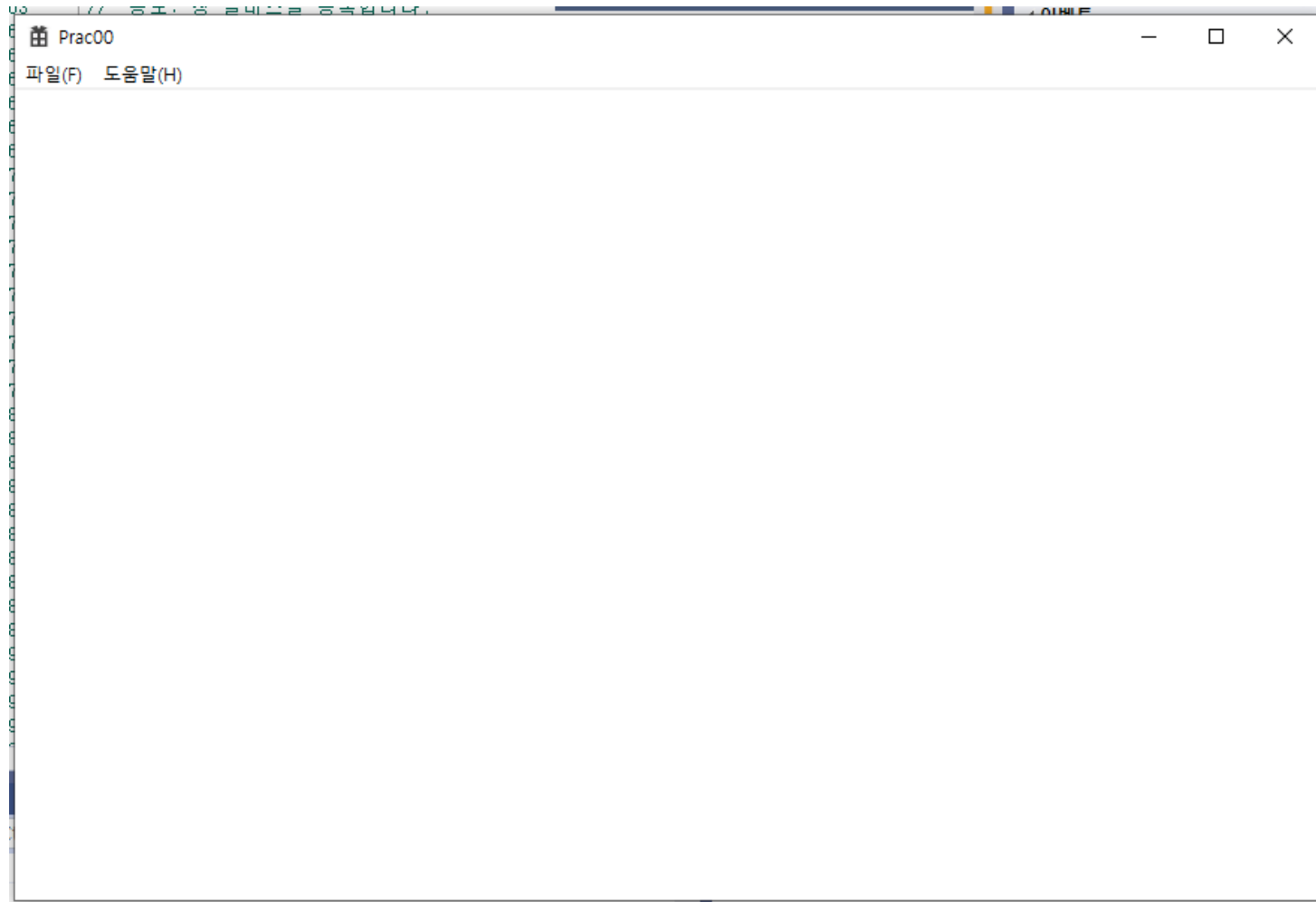
szTitle x → ×

Aa AB * 현재 문서

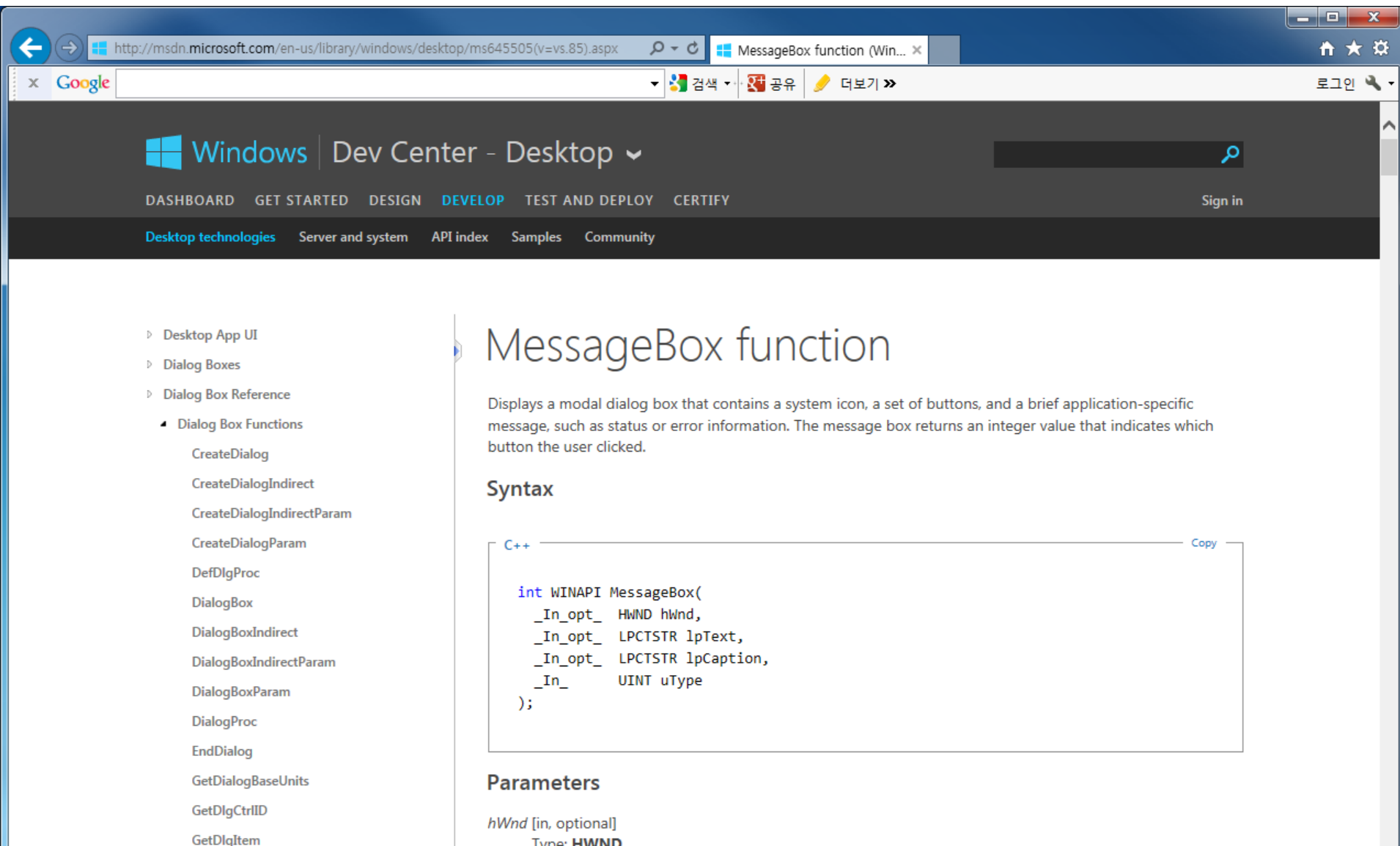
Handling Window Messages

```
149     break;
150     case WM_PAINT:
151     {
152         PAINTSTRUCT ps;
153         HDC hdc = BeginPaint(hWnd, &ps);
154         // TODO: 여기에 hdc를 사용하는 그리기 코드를 추가합니다...
155         EndPaint(hWnd, &ps);
156     }
157     break;
158     case WM_LBUTTONDOWN:
159         MessageBox(NULL, L"Hello, World", L"Help", MB_OK);
160         break;
161     case WM_KEYDOWN:
162         DestroyWindow(hWnd);
163         break;
164     case WM_DESTROY:
165         PostQuitMessage(0);
166         break;
167     default:
168         return DefWindowProc(hWnd, message, wParam, lParam);
169     }
170     return 0;
171 }
```

실행 화면 (2)



MSDN - MessageBox()



The screenshot shows the MSDN website for the MessageBox function. The browser address bar displays the URL: [http://msdn.microsoft.com/en-us/library/windows/desktop/ms645505\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/ms645505(v=vs.85).aspx). The page title is "MessageBox function (Win...". The navigation bar includes "Windows Dev Center - Desktop" and a search bar. The main content area is titled "MessageBox function" and includes a description, syntax, and parameters.

Desktop App UI
Dialog Boxes
Dialog Box Reference
Dialog Box Functions
CreateDialog
CreateDialogIndirect
CreateDialogIndirectParam
CreateDialogParam
DefDlgProc
DialogBox
DialogBoxIndirect
DialogBoxIndirectParam
DialogBoxParam
DialogProc
EndDialog
GetDialogBaseUnits
GetDlgCtrlID
GetDlgItem

MessageBox function

Displays a modal dialog box that contains a system icon, a set of buttons, and a brief application-specific message, such as status or error information. The message box returns an integer value that indicates which button the user clicked.

Syntax

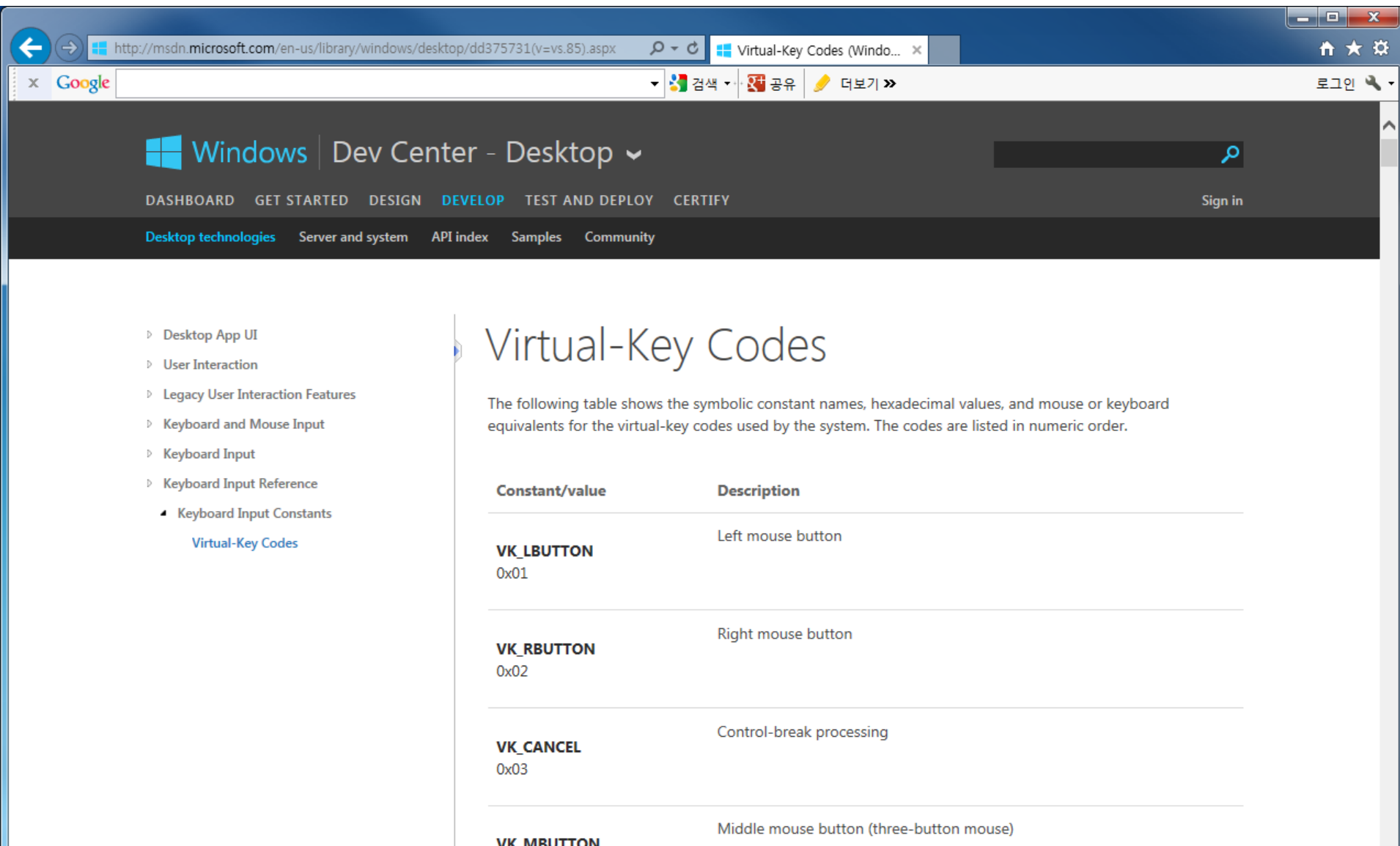
```
C++  
int WINAPI MessageBox(  
    _In_opt_  HWND hWnd,  
    _In_opt_  LPCTSTR lpText,  
    _In_opt_  LPCTSTR lpCaption,  
    _In_      UINT uType  
);
```

Copy

Parameters

hWnd [in, optional]
Type: **HWND**

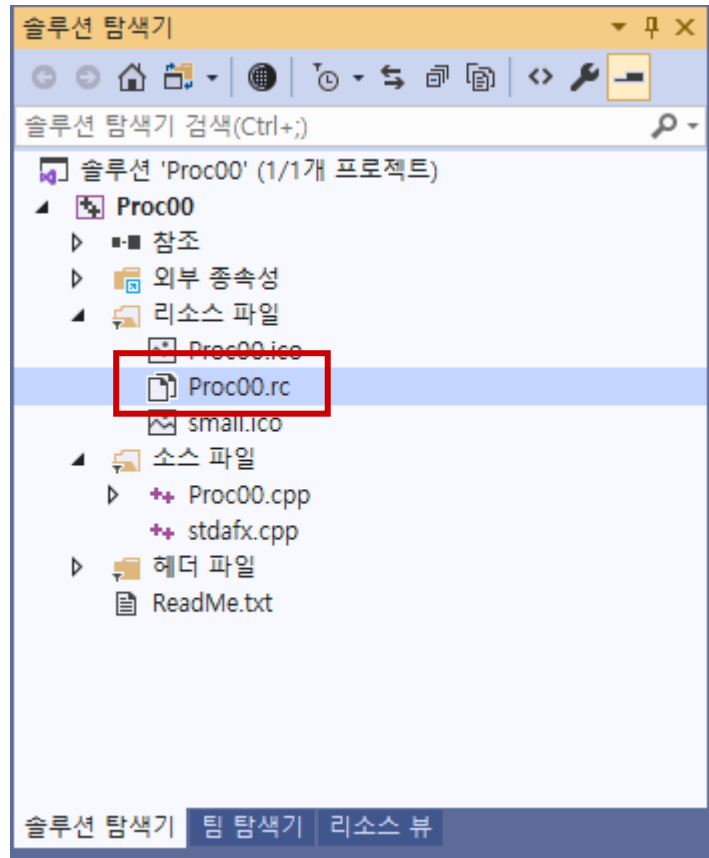
MSDN - Virtual-Key Codes



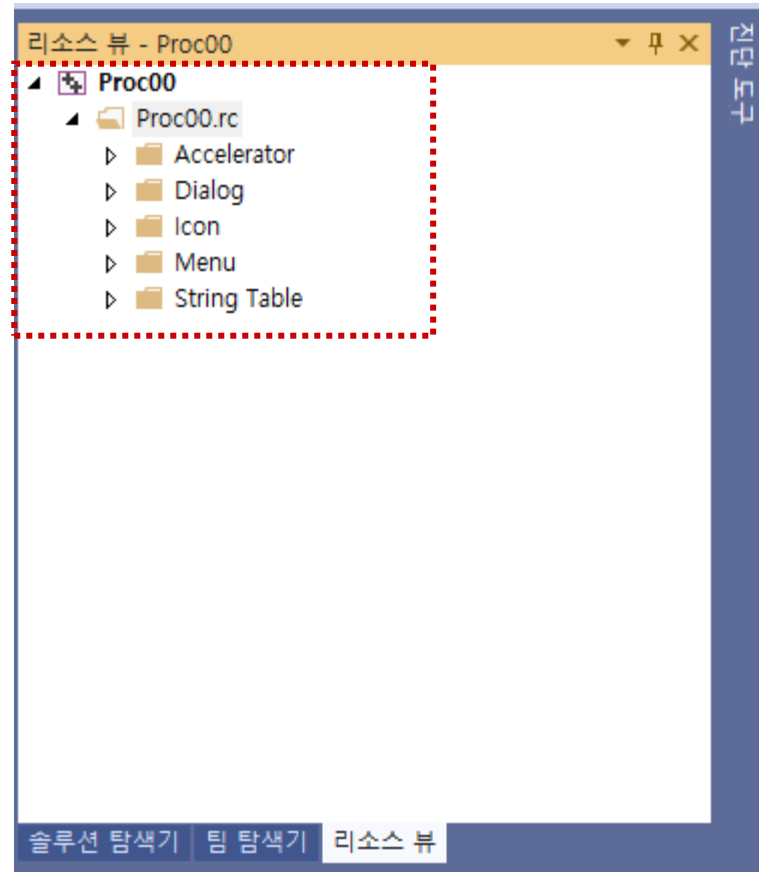
The screenshot shows a web browser window with the URL [http://msdn.microsoft.com/en-us/library/windows/desktop/dd375731\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/dd375731(v=vs.85).aspx). The page is titled "Virtual-Key Codes" and is part of the "Windows Dev Center - Desktop" section. The left sidebar contains a navigation menu with the following items: Desktop App UI, User Interaction, Legacy User Interaction Features, Keyboard and Mouse Input, Keyboard Input, Keyboard Input Reference, and Keyboard Input Constants. The "Keyboard Input Constants" item is expanded, showing "Virtual-Key Codes" as a sub-item. The main content area features the title "Virtual-Key Codes" and a paragraph explaining that the following table shows symbolic constant names, hexadecimal values, and mouse or keyboard equivalents for the virtual-key codes used by the system. The table is organized into two columns: "Constant/value" and "Description".

Constant/value	Description
VK_LBUTTON 0x01	Left mouse button
VK_RBUTTON 0x02	Right mouse button
VK_CANCEL 0x03	Control-break processing
VK_MBUTTON	Middle mouse button (three-button mouse)

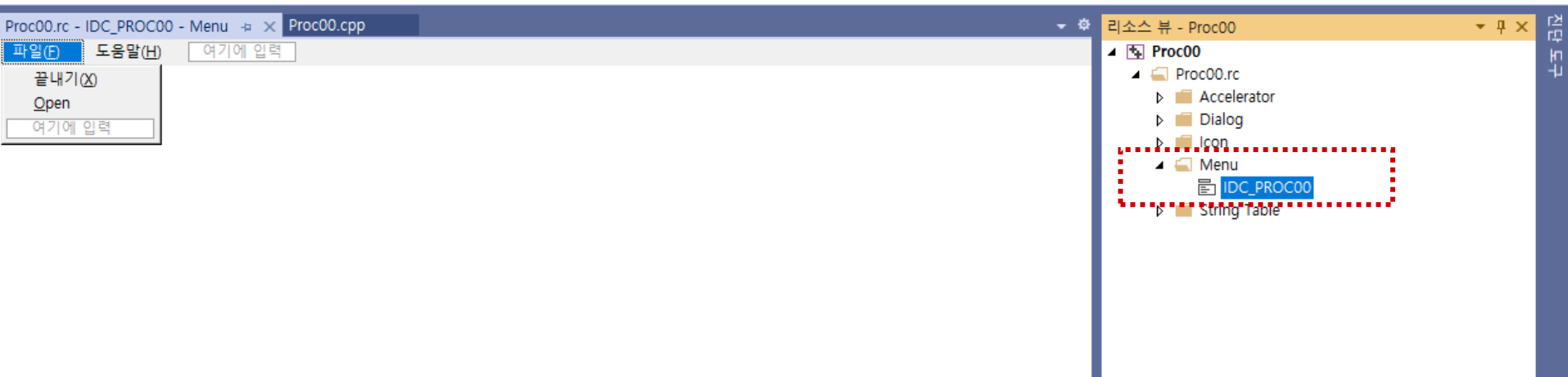
Resource (1)



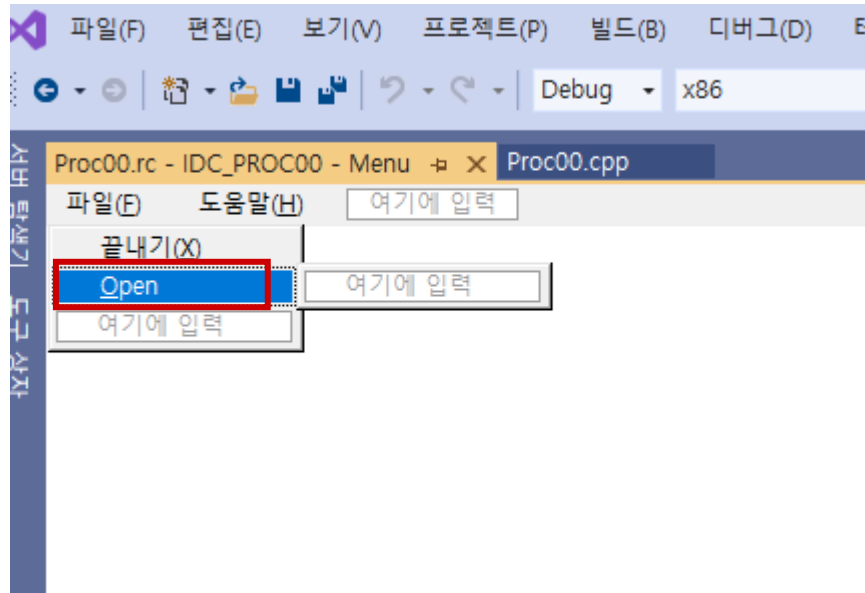
Resource (2)



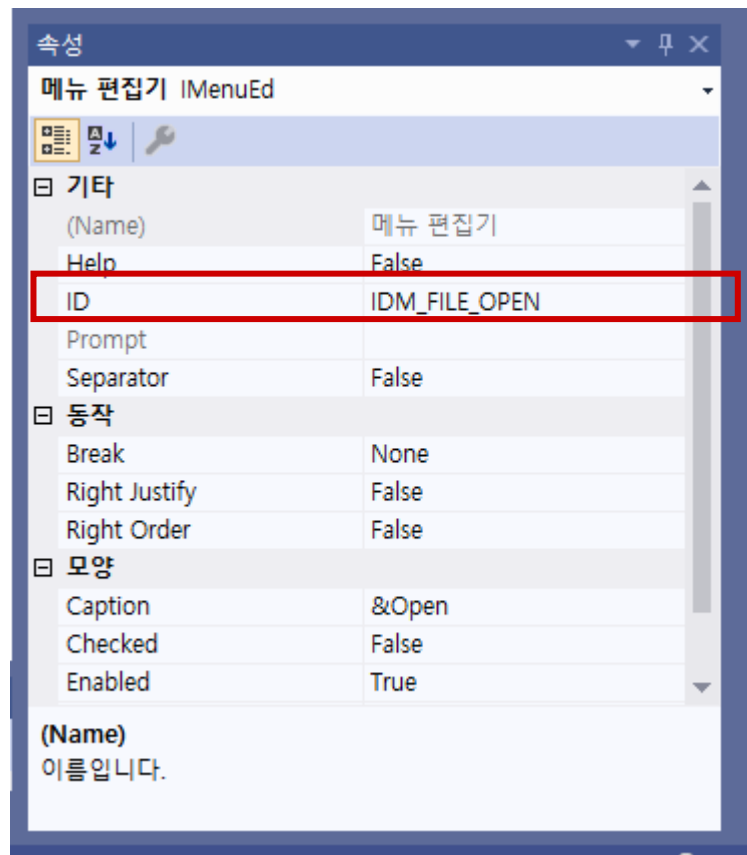
Menu (1)



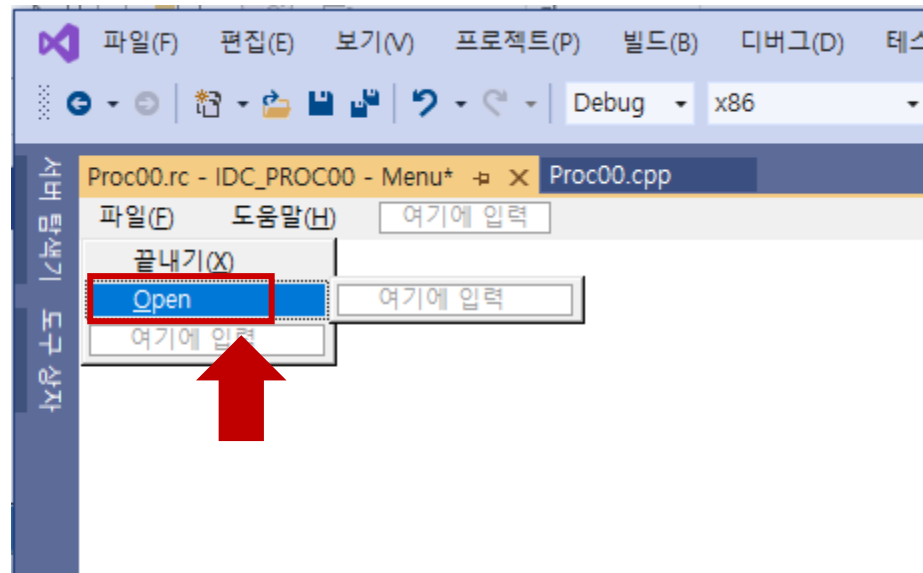
Menu (2)



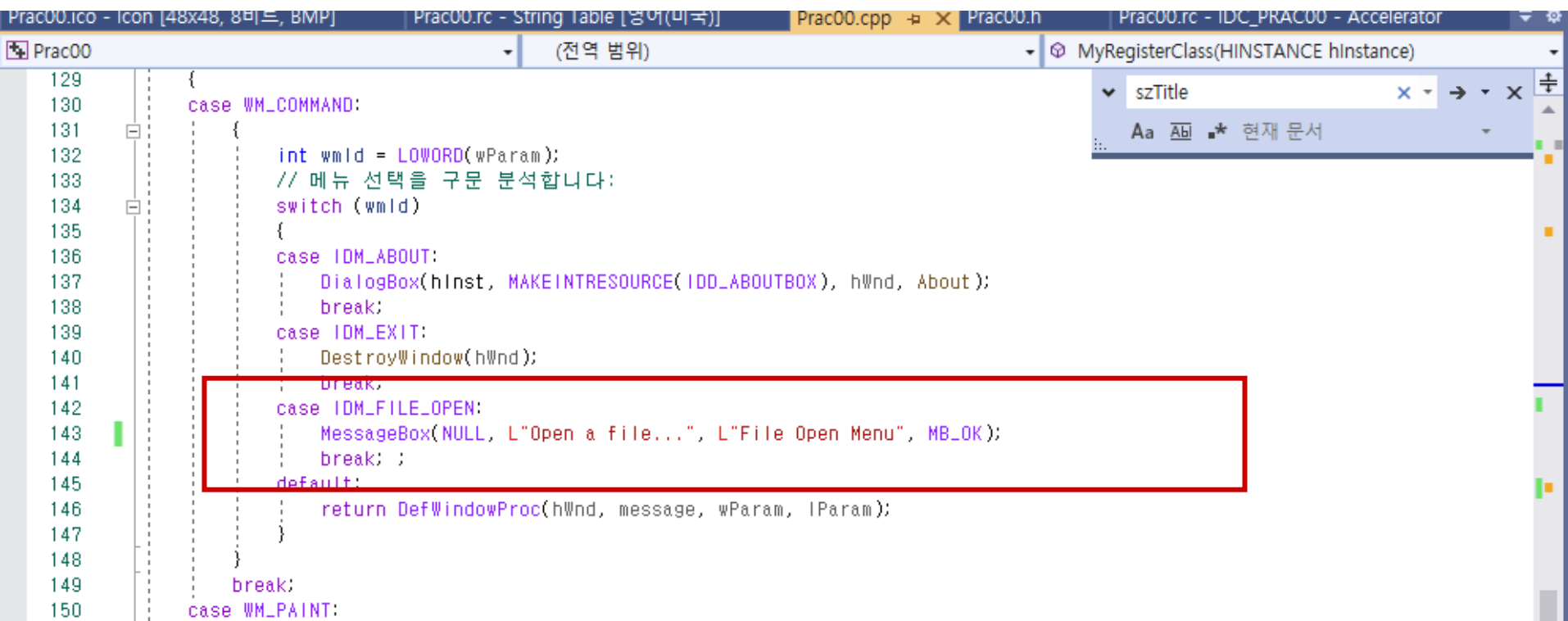
Menu (3)



Menu (4)



Menu (5)

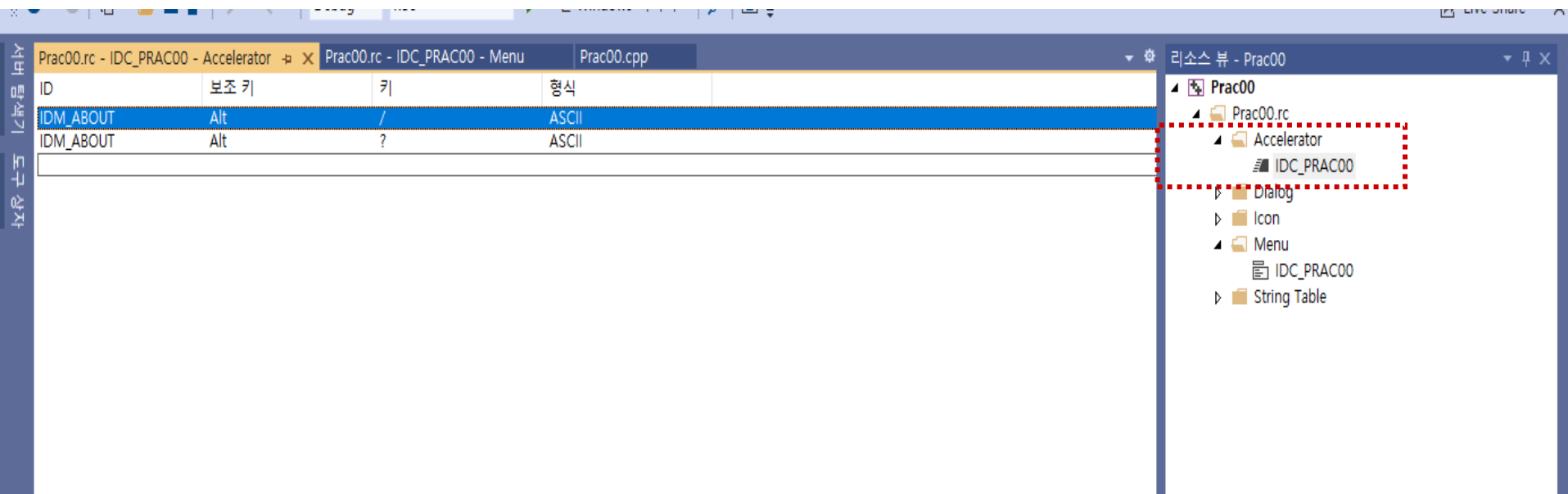


The screenshot shows a code editor with the following tabs: Prac00.ico - Icon [48x48, 8비트, BMP], Prac00.rc - String Table [영어(미국)], Prac00.cpp, Prac00.h, and Prac00.rc - IDC_PRAC00 - Accelerator. The active file is Prac00.cpp, showing the function MyRegisterClass(HINSTANCE hInstance). The code is as follows:




```
129 {
130     case WM_COMMAND:
131     {
132         int wmid = LOWORD(wParam);
133         // 메뉴 선택을 구문 분석합니다:
134         switch (wmid)
135         {
136             case IDM_ABOUT:
137                 DialogBox(hInst, MAKEINTRESOURCE(IDD_ABOUTBOX), hWnd, About);
138                 break;
139             case IDM_EXIT:
140                 DestroyWindow(hWnd);
141                 break;
142             case IDM_FILE_OPEN:
143                 MessageBox(NULL, L"Open a file...", L"File Open Menu", MB_OK);
144                 break; ;
145             default:
146                 return DefWindowProc(hWnd, message, wParam, lParam);
147         }
148     }
149     break;
150     case WM_PAINT:
```

The code block for `IDM_FILE_OPEN` (lines 142-144) is highlighted with a red rectangle. The IDE interface includes a Solution Explorer on the left, a Properties window on the right showing `szTitle`, and a status bar at the bottom indicating the current document.

Accelerator (1)

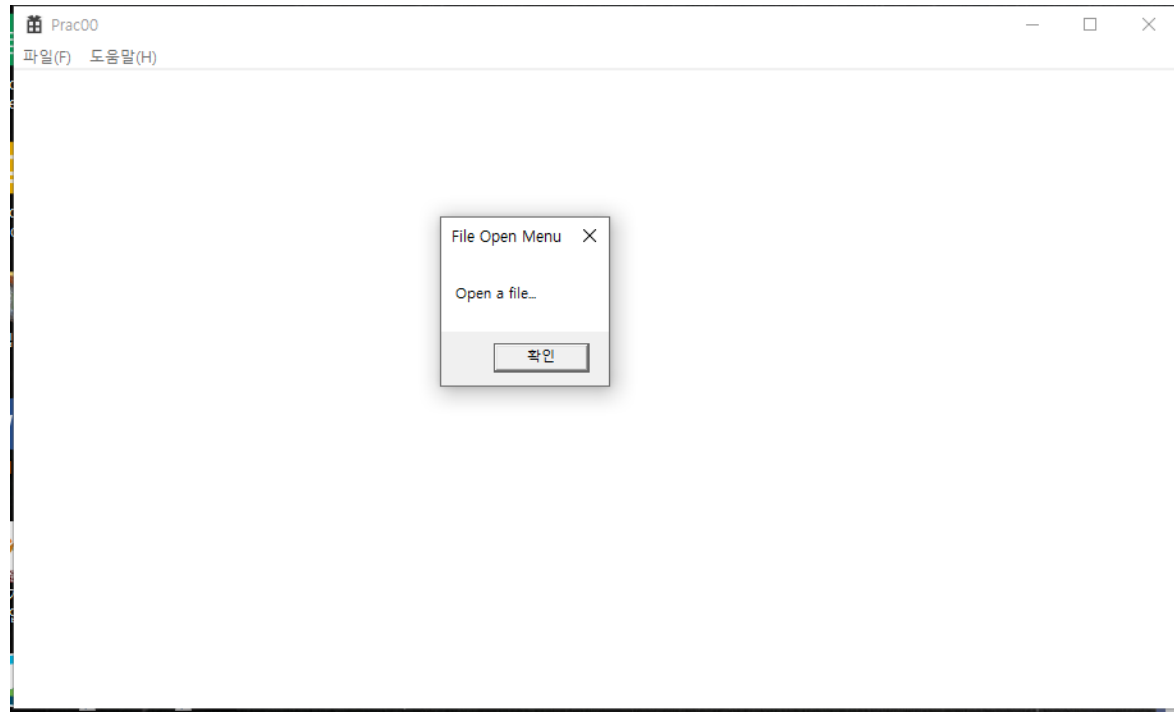


Accelerator (2)

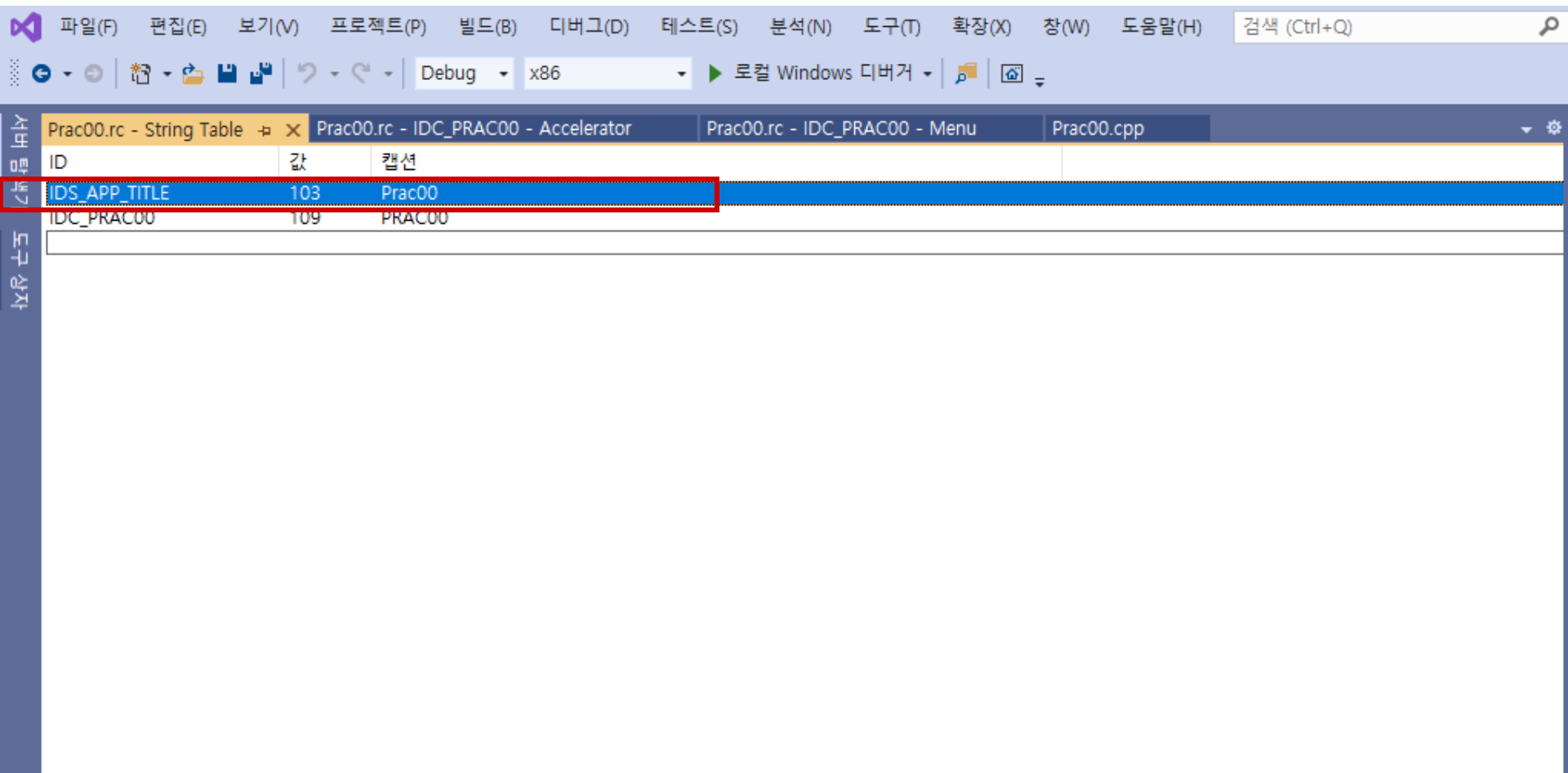
Prac00.rc - IDC_PRAC00 - Accelerator+  				Prac00.rc - IDC_PRAC00 - Menu+	Prac00.cpp		
ID	보조 키	키	형식				
IDM_ABOUT	Alt	/	ASCII				
IDM_ABOUT	Alt	/	ASCII				
IDM_FILE_OPEN	Ctrl	O	VIRTKEY				

실행 화면 (3)

- 실행 해 볼 것
- Ctrl + O를 누르면...



String Table (1)



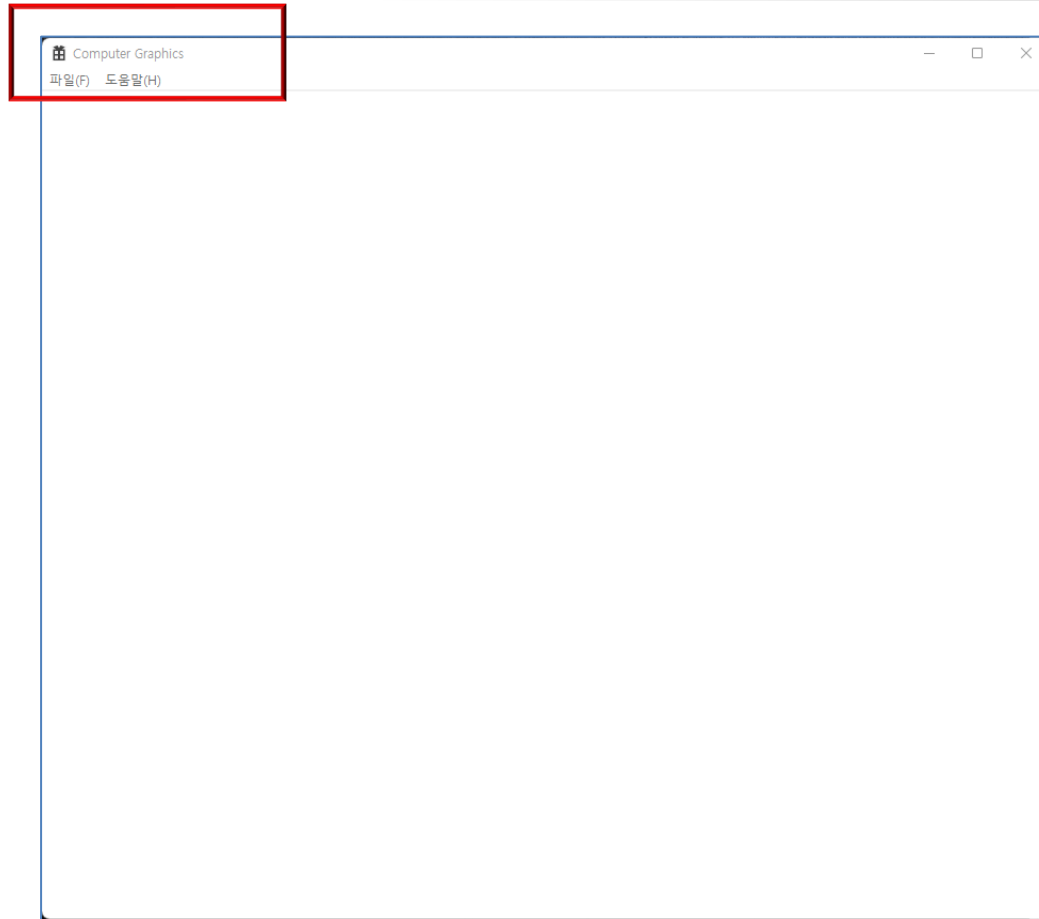
String Table (2)

Visual Studio interface showing the String Table editor for a project named Prac00. The table lists string IDs and their corresponding values.

ID	값	캡션
IDS_APP_TITLE	103	Computer Graphics
IDC_PRAC00	109	PRAC00

The table is displayed in the 'Prac00.rc - String Table [영어(미국)]' tab. The 'IDS_APP_TITLE' row is highlighted with a red border.

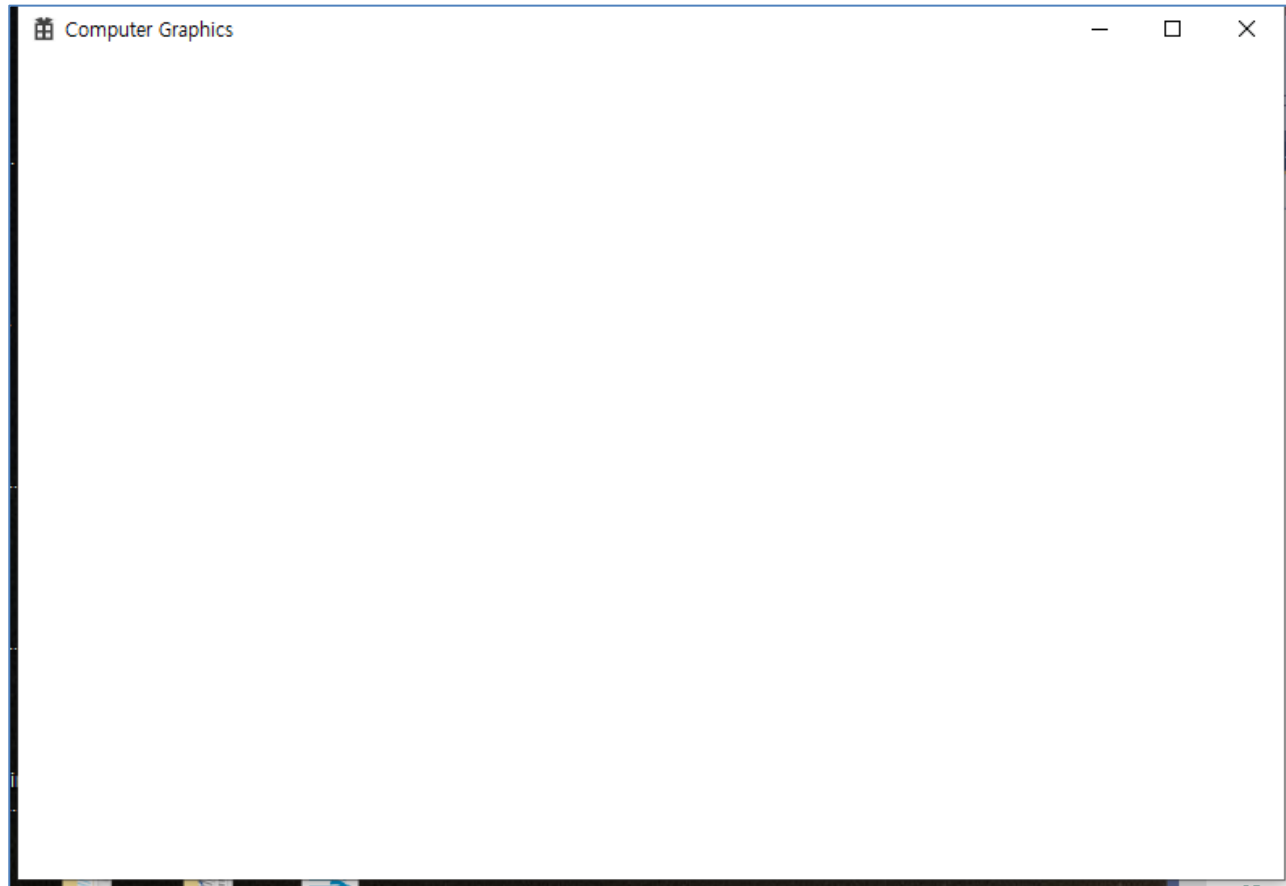
실행 화면 (4)



메뉴 제거

```
64  L//
65  ATOM MyRegisterClass(HINSTANCE hInstance)
66  {
67      WNDCLASSEXW wcex;
68
69      wcex.cbSize = sizeof(WNDCLASSEX);
70
71      wcex.style      = CS_HREDRAW | CS_VREDRAW;
72      wcex.lpfnWndProc = WndProc;
73      wcex.cbClsExtra = 0;
74      wcex.cbWndExtra = 0;
75      wcex.hInstance  = hInstance;
76      wcex.hIcon       = LoadIcon(hInstance, MAKEINTRESOURCE(IDI_PROCOO));
77      wcex.hCursor     = LoadCursor(nullptr, IDC_ARROW);
78      wcex.hbrBackground = (HBRUSH)(BLACK_BRUSH);
79      wcex.lpszMenuName = 0; // MAKEINTRESOURCEW(IDC_PROCOO);
80      wcex.lpszClassName = szWindowClass;
81      wcex.hIconSm      = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(IDI_SMALL));
82
83      return RegisterClassExW(&wcex);
84  }
85
```

실행 화면 (4)



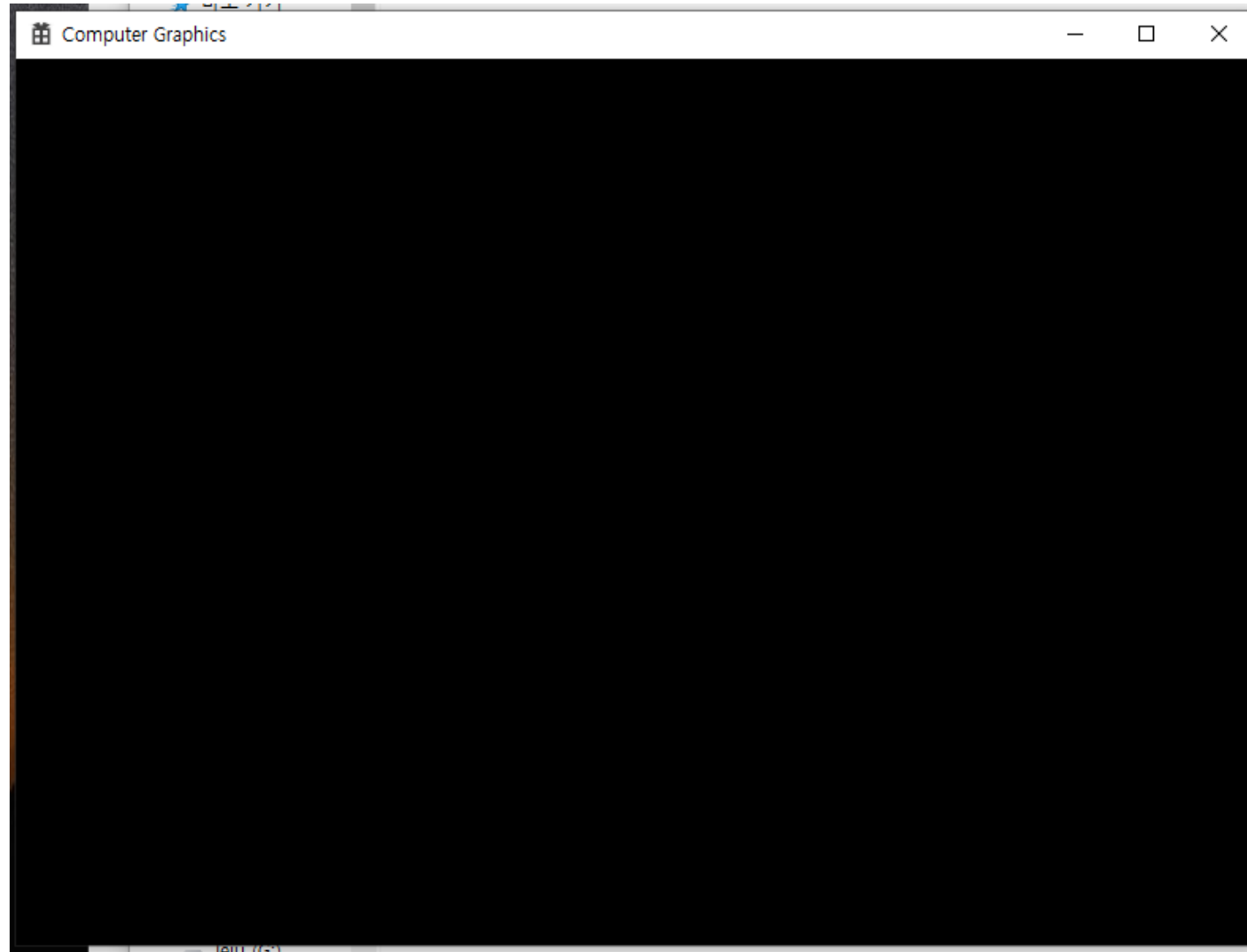
연습 문제 (실습 30분)

- 배경색 - 다른 색으로 변경 (예: 검정색)
- 윈도우 크기 - 800 x 600으로 변경
- 타이틀 - '학번 이름'으로 변경
- Esc 버튼 눌렀을 때, 대화상자 생성
 - “종료하시겠습니까?”
 - OK or Cancel
- [Option] OK 했을 경우에만 종료하도록 프로그램 변경

과제

- 배경색 - 다른 색으로 변경 (예: 초록색, 파랑색 등)
- 윈도우 크기 - 800 x 600으로 변경
- 타이틀 - '학번과 이름'으로 변경
- Esc 버튼 눌렀을 때, 대화상자 생성
 - “종료하시겠습니까?”
 - OK or Cancel
- [Option] OK 했을 경우에만 종료하도록 프로그램 변경
- 제출기한
 - 2023년 4월 3일 23시 59분까지 제출 할 것
- 제출방법
 - ‘하영드리미 → 리포트’에 제출
 - “3주차_컴퓨터그래픽스_학번_이름.doc” 파일 이용할 것
 - 마감 후 과제 받지 않음

실행 화면 (5)



The End