제4장 윈도우 메뉴

2018년 1학기 윈도우 프로그래밍

학습 목표

- **학습목표**
  - 리소스 파일을 만들고 윈도우에 메뉴를 등록하는 방법을 배운다.
  - 메뉴에서 발생하는 메시지를 처리하는 방법을 배운다.
  - 공용 대화상자의 종류를 알아보고 이를 적용해 프로그램의 완성도를 높인다.
  - 윈도우에 등록된 메뉴를 프로그램 실행 중에 수정하는 방법을 배운다.

- **내용**
  - 메뉴 만들기
  - 메뉴 사용하기
  - 공용 대화상자 이용하기
  - 메뉴 수정하기

1. 메뉴 만들기

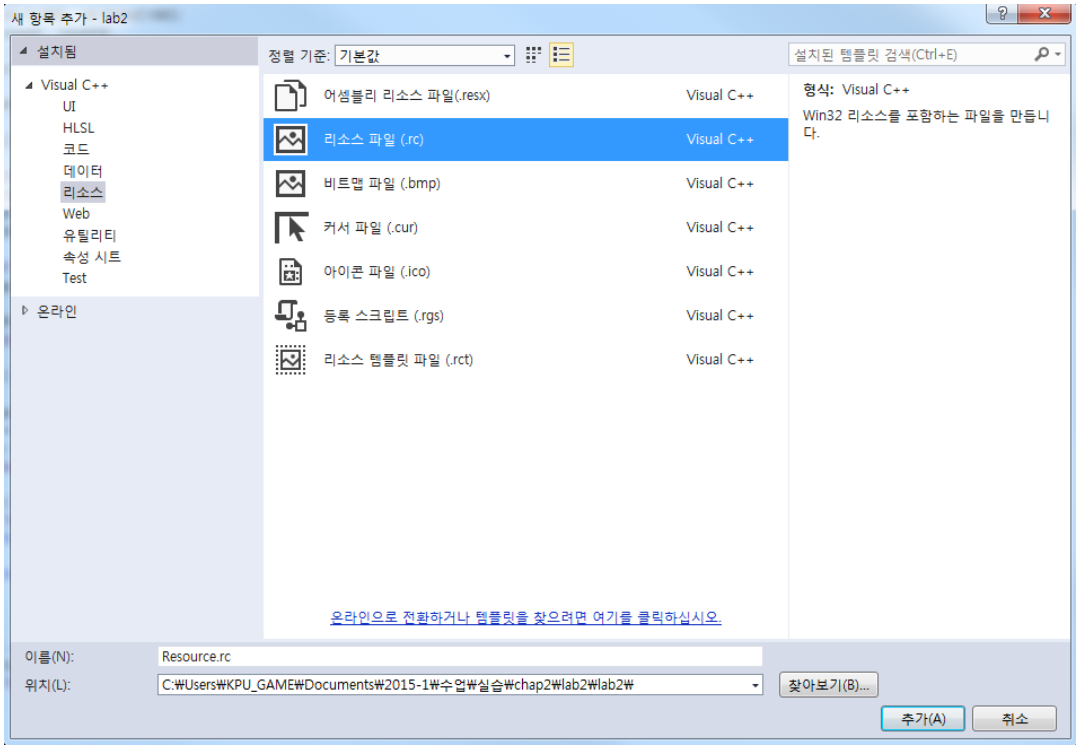
- **리소스 파일 작성**
  - 방법: 소스 파일 작성과 유사
  - C++ 소스 대신에 Resource Script 선택
  - 리소스 파일 이름 명시
  - 리소스-파일.rc(리소스 정의), resource.h(리소스 ID 정의) 생성

- **메뉴화면 편집**
  - 리소스 도구상자 이용
  - 속성 정의 : 프로그램과 연계
  - 메뉴 ID 정의 : 메뉴바에서

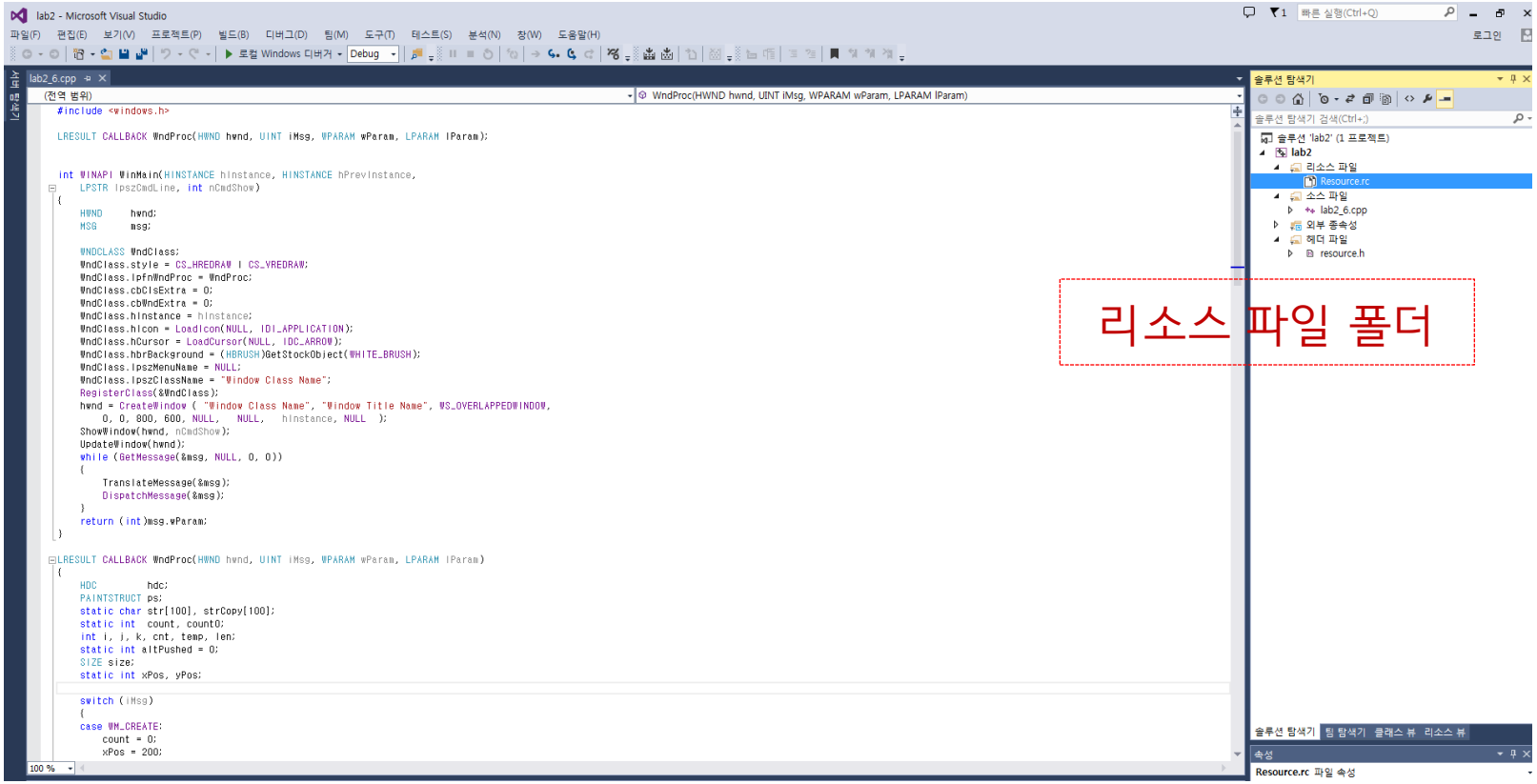
- **프로그램에서 메뉴 사용**
  - include "resource.h"
  - 메뉴ID 등록 : 윈도우 클래스
  - 메뉴 처리 : WM_COMMAND 메시지

메뉴 만들기

- Visual Studio 2015 환경
  - [프로젝트] -> [리소스 파일] -> [새 항목 추가]

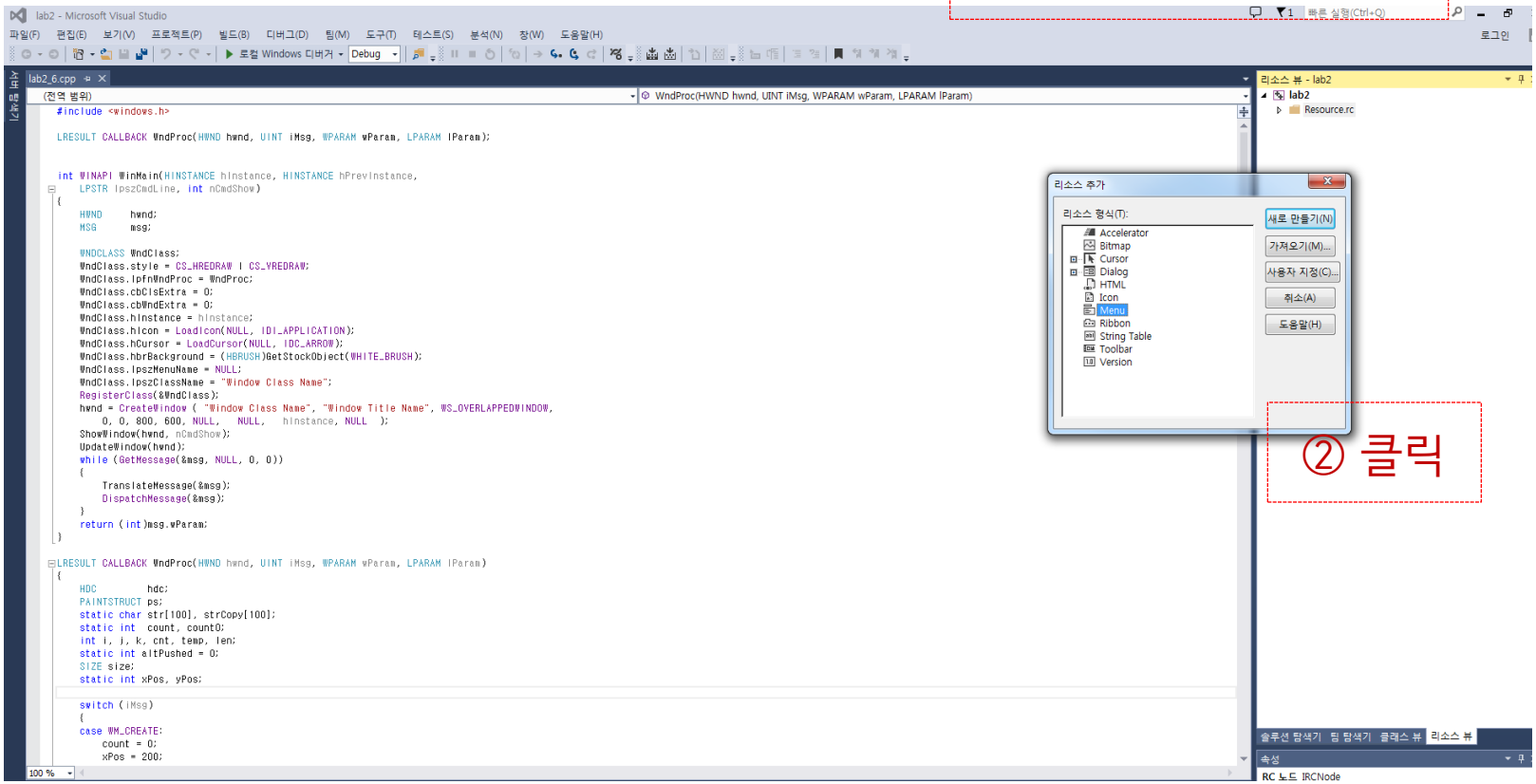


생성된 리소스 파일


lab2 - Microsoft Visual Studio
lab2_6.cpp
(전역 범위)
WndProc(HWND hwnd, UINT iMsg, WPARAM wParam, LPARAM lParam)
#include <windows.h>
LRESULT CALLBACK WndProc(HWND hwnd, UINT iMsg, WPARAM wParam, LPARAM lParam);
int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
LPSTR lpszCmdLine, int nCmdShow)
{
HWND hwnd;
MSG msg;
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 = "Window Class Name";
RegisterClass(&WndClass);
hwnd = CreateWindow ("Window Class Name", "Window Title Name", WS_OVERLAPPEDWINDOW,
0, 0, 800, 600, NULL, NULL, hInstance, NULL);
ShowWindow(hwnd, nCmdShow);
UpdateWindow(hwnd);
while (GetMessage(&msg, NULL, 0, 0))
{
TranslateMessage(&msg);
DispatchMessage(&msg);
}
return (int)msg.wParam;
}
LRESULT CALLBACK WndProc(HWND hwnd, UINT iMsg, WPARAM wParam, LPARAM lParam)
{
HDC hdc;
PAINTSTRUCT ps;
static char str[100], strCopy[100];
static int count, count0;
int i, j, k, cnt, temp, len;
static int altPushed = 0;
SIZE size;
static int xPos, yPos;
switch (iMsg)
{
case WM_CREATE:
count = 0;
xPos = 200;
솔루션 탐색기
솔루션 'lab2' (1 프로젝트)
lab2
리소스 파일
Resource.rc
소스 파일
lab2_6.cpp
외부 종속성
헤더 파일
resource.h
리소스 파일 폴더
솔루션 탐색기 팀 탐색기 클래스 뷰 리소스 뷰
속성
Resource.rc 파일 속성

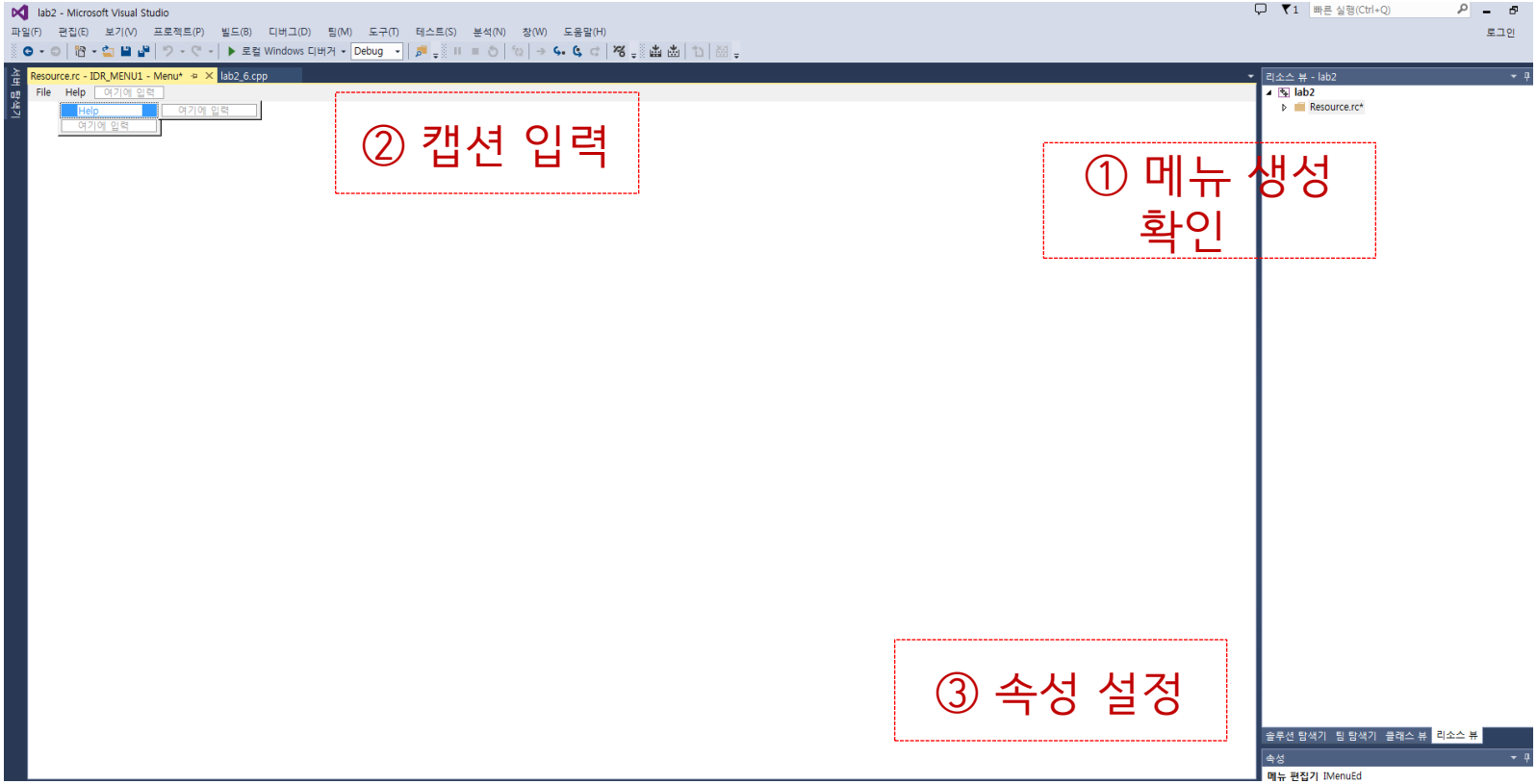
메뉴 만들기

① 리소스 파일 폴더를 선택한 후 클릭

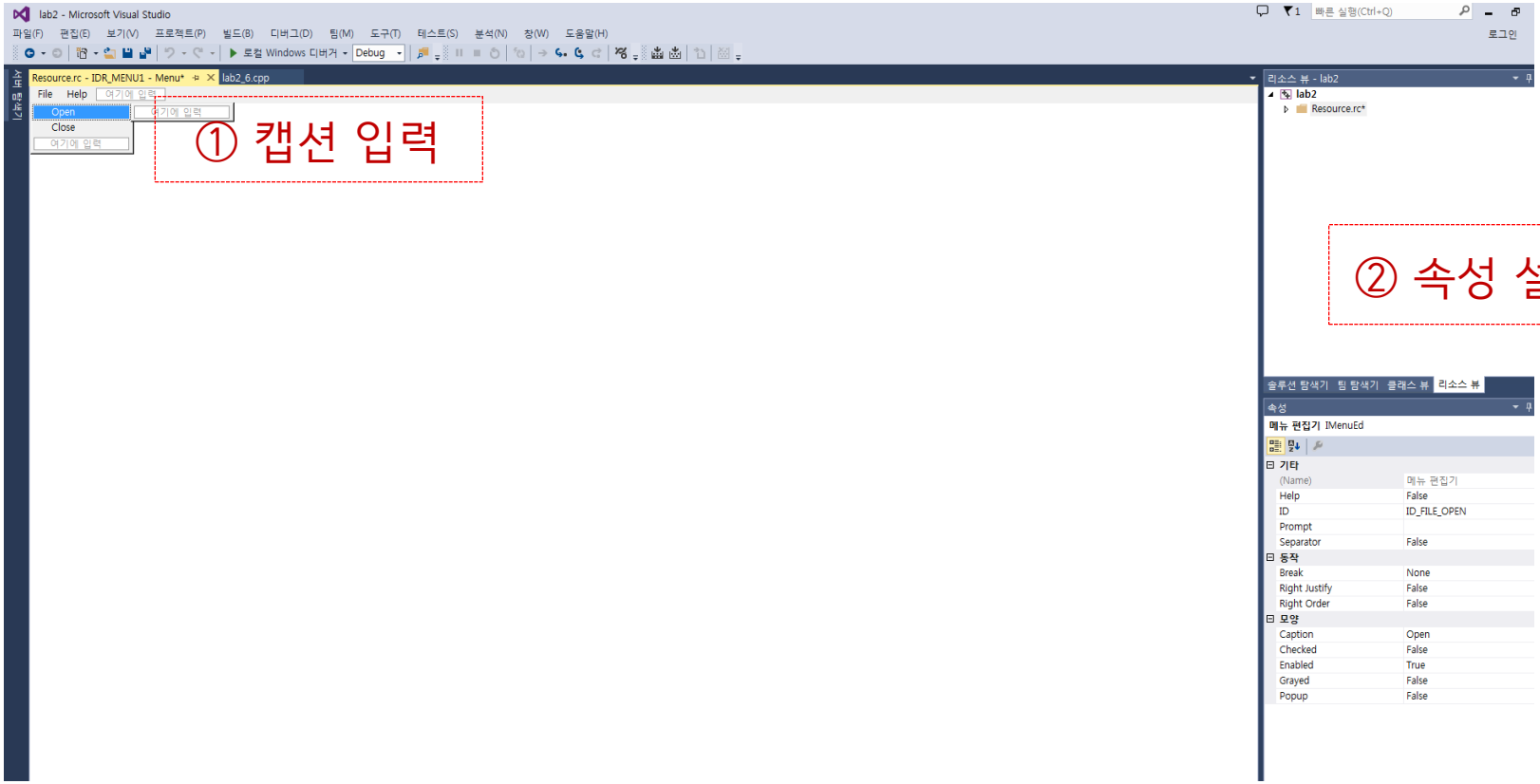


② 클릭

메뉴 항목 편집


lab2 - Microsoft Visual Studio
Resource.rc - IDR_MENU1 - Menu*
lab2_6.cpp
File
Help
여기에 입력
② 캡션 입력
① 메뉴 생성 확인
③ 속성 설정
리소스 뷰 - lab2
lab2
Resource.rc*
속성
메뉴 편집기 IMenuEd

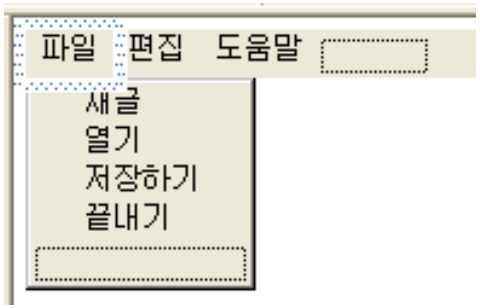
메뉴 항목 추가


lab2 - Microsoft Visual Studio
Resource.rc - IDR_MENU1 - Menu*
lab2_6.cpp
File
Help
Open
Close
① 캡션 입력
리소스 뷰 - lab2
lab2
Resource.rc*
② 속성 설정
속성
메뉴 편집기 IMenuEd
기타
(Name)
Help
ID
Prompt
Separator
동작
Break
Right Justify
Right Order
모양
Caption
Checked
Enabled
Grayed
Popup

메뉴 만들기

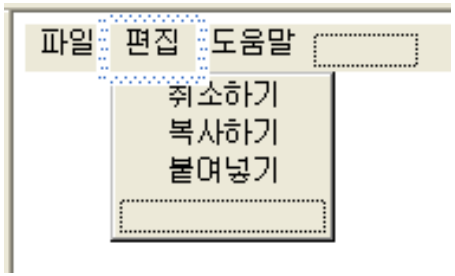
- **예제 내용**
  - 다음의 메뉴항목 설정표를 참고하여 기본 메뉴를 작성하기.
    -> 다음 페이지 설명 계속

Caption	ID	속성
파일		Pop-up
새글	ID_FILENEW	디폴트
열기	ID_FILEOPEN	디폴트
저장하기	ID_FILESAVE	디폴트
끝내기	ID_EXIT	디폴트

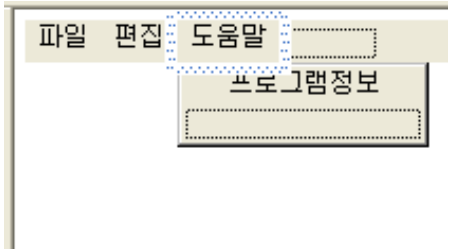


실습 4-1(계속)

Caption	ID	속성
편집		Pop-up
취소하기	ID_EDITUNDO	디폴트
복사하기	ID_EDITCOPY	디폴트
붙여넣기	ID_EDITPASTE	디폴트



Caption	ID	속성
도움말		Pop-up
프로그램정보	ID_INFORM	디폴트



2. 메뉴 사용하기

- 메뉴를 선택했을 때 WM_COMMAND 메시지가 전달
  - WM_COMMAND
    - 메뉴의 메뉴항목을 선택하면 발생하는 메시지
    - Command 메시지라 부름

  - LOWORD(wParam)
    - 선택된 메뉴항목의 ID가 정수로 들어 있음
  - HIWORD(wParam)
    - 0 (이벤트 소스)
  - lParam
    - 0

윈도우에 메뉴 붙이기

- 응용 프로그램에 메뉴 리소스를 불러오는 방법 3가지

1) 윈도우 클래스를 만들 때 메뉴를 정의

```
winclass.lpszMenuName = MAKEINTRESOURCE ( MYMENU );
```

2) 메인 윈도우를 생성할 때 메뉴를 첨부

```
winclass.lpszMenuName = NULL;
CreateWindow (szWindowClass,  szTitle, WS_OVERLAPPEDWINDOW,
              0, 0, 800, 600, NULL,
              LoadMenu ( hinstance, MAKEINTRESOURCE (MYMENU ) ),
              hInstance, NULL);
```

3) 초기 윈도우 생성이 끝난 후에 붙인다.

```
HMENU hmymenu = LoadMenu (hinstance, MAKEINTRESOURCE ( MYMENU ) );
SetMenu (hwnd, hmymenu);
```

윈도우에 메뉴 붙이기

```
#include "resource.h"                    // 리소스 파일 첨부


int WINAPI WinMain ( … )
{
    // wndclass 속성 설정에서
        wndclass.lpszMenuName  = MAKEINTRESOURCE (IDR_MENU);
                                                    // 메뉴 ID 등록       : 클래스 파일
    …
}
```

- MAKEINTRESOURCE: 리소스에 대한 정수형 상수를 문자열로 변환하는 매크로 함수
  - LPTSTR MAKEINTRESOURCE(
    WORD wInt;   // 리소스에 대한 정수형 상수
    );

커맨드 메시지 처리하기

```
int answer;

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

{
   switch (iMsg)
   {
        case WM_COMMAND:
             switch (LOWORD(wParam)) {
                  case ID_FILENEW:
                       MessageBox (hwnd, "새파일을 열겠습니까 ?", "새파일 선택" , MB_OKCANCEL );
                       break;

                  case ID_EXIT:
                       answer = MessageBox (hwnd, "파일을 저장하고 끝내겠습니까 ? ",
                                                  "끝내기 선택", MB_YESNOCANCEL );
                       if (answer == IDYES || answer == IDNO)
                                  PostQuitMessage (0);;
                       break;
             }
        break;
}
```

메시지박스

uType	반환 값
MB_OK	IDOK
MB_OKCANCEL	IDOK, IDCANCEL
MB_YESNO	IDYES, IDNO
MB_YESNOCANCEL	IDYES, IDNO, IDCANCEL

- **메시지박스: 사용자에게 경고나 알림 메시지를 주는 대화상자**

int **MessageBox** (HWND hwnd, LPCTSTR lpText, LPCTSTR lpCaption, UINT uType);

- lpText: 메시지 박스에 표시될 글
- lpCaption: 메시지 박스의 타이틀바에 표시될 글
- uType과 함수 반환값

리소스: 아이콘

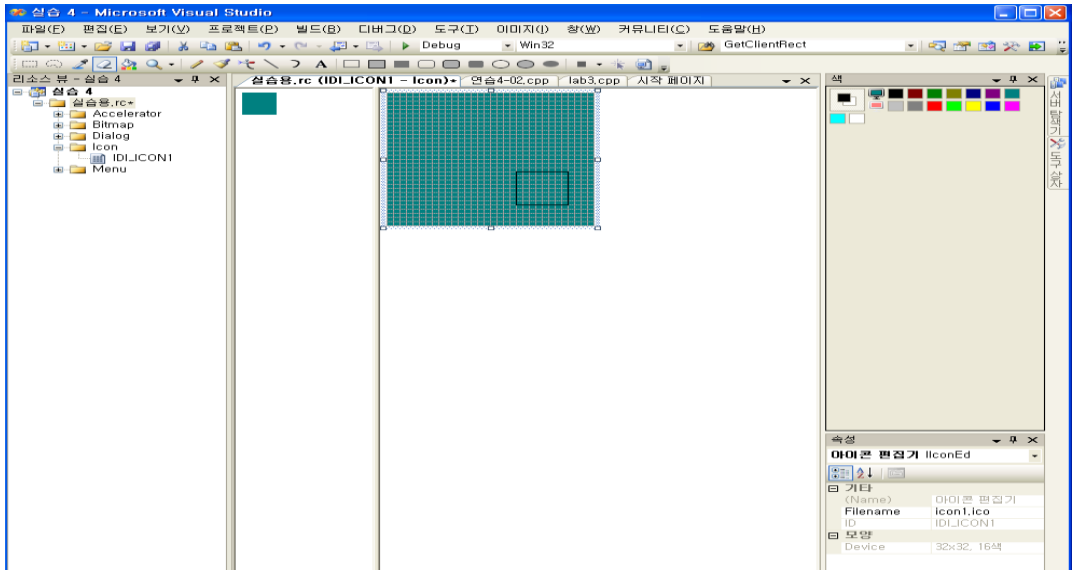
- 아이콘은 프로그램의 메인 윈도우가 최소화(아이콘화)되었을 때나 배경화면에 등록 될 때 응용 프로그램을 나타내는 작은 그래픽 이미지이다.
- 아이콘과 커서는 크게 두 가지 종류의 리소스가 있다.
  - 내장(built-in) 리소스와 사용자 정의 리소스이다.
  - 내장 리소스: 윈도우즈에서 기본적으로 제공하는 것이다.
  - LoadIcon() 함수의 첫 번째 인자에 "NULL"을 주고 두 번째 인자로 아이콘 이름 문자열을 지정하며 리턴된 핸들을 wc.hIcon에 대입한다.
- 손(IDI_HAND), 느낌표(IDI_WARNING) 등 9가지가 있으며 지금까지 사용한 표준 내장 아이콘(IDI_APPLICATION)은 다음과 같은 윈도우 모양을 갖는다.

```
wc.hIcon      = LoadIcon ( NULL, IDI_APPLICATION );
wc.hIconSm = LoadIcon ( NULL, IDI_APPLICATION );
```

Value	Description
IDI_APPLICATION	Default application icon.
IDI_ASTERISK	Same as IDI_INFORMATION.
IDI_ERROR	Hand-shaped icon.
IDI_EXCLAMATION	Same as IDI_WARNING.
IDI_HAND	Same as IDI_ERROR.
IDI_INFORMATION	Asterisk icon.
IDI_QUESTION	Question mark icon.
IDI_WARNING	Exclamation point icon.
IDI_WINLOGO	Windows logo icon.

리소스: 아이콘 편집

- 32×32픽셀에 칼라로 원하는 아이콘을 그릴 수 있다.
- 화면 오른쪽에 그림판 프로그램에서 볼 수 있는 그래픽 편집 도구들이 보인다.
- 혹시 이 도구들이 보이지 않는 경우에는 캡션바를 제외한 비 클라이언트 영역에서 마우스 오른쪽 버튼을 클릭하여 나오는 팝업 메뉴에서 “색상창표시"를 체크하면 된다.



리소스: 커서

- 커서는 마우스의 위치를 나타내는 작은 그래픽 이미지이다.
  - 내장 커서는 아래의 표와 같은 모양이 제공된다.
  - LoadCursor()함수의 첫 번째 인자에 "NULL"을 주고 두 번째 인자로 커서 이름 문자열을 지정하며 리턴된 핸들을 wc.hCursor에 대입한다.
- 화살표, 모래시계 등 11가지가 있으며 지금까지 사용한 표준 내장 커서는 화살표 모양을 갖는다.
- 아이콘과 마찬가지로 새로운 리소스에서 커서를 선택하여 비트맵을 그린다.

```
wc.hCursor = LoadCursor ( NULL ,IDC_ARROW);
```

값	커서	모양
IDC_APPSTARTING	프로그램이 시작될 때 사용된다.	
IDC_ARROW	표준 화살표 커서	
IDC_CROSS	십자 모양의 커서. 정확한 선택을 해야 할 때 사용된다.	
IDC_IBEAM	I자 모양의 커서. 주로 문자열 입력 영역에 사용된다.	
IDC_ICON	Win32에서는 사용되지 않음	
IDC_NO	원 안의 빗금이 쳐진 커서이며 드래그 금지 구역을 나타낸다.	
IDC_SIZE	Win32에서는 사용되지 않음	
IDC_SIZEALL	4방향 화살표	
IDC_SIZENESW	좌하우상 크기조절 커서	
IDC_SIZENS	수직 크기조절 커서	
IDC_SIZENWSE	좌상우하 크기조절 커서	
IDC_SIZEWE	수평 크기조절 커서	
IDC_UPARROW	수직 화살표	
IDC_WAIT	모래 시계 커서. 시간이 오래 걸리는 작업을 할 때 사용된다.	

리소스: 커서 편집

- 사용자 정의 아이콘과 리소스를 만들었으면 소스에 새로 만든 리소스를 연결해 주어야 한다.
  - wc.hIcon = LoadIcon (hInstance, MAKEINTRESOURCE (IDI_ICON1));
  - wc.hCursor = LoadCursor (hInstance, MAKEINTRESOURCE(IDC_CURSOR1));
  - wc.hIconSm = LoadIcon (hInstance, MAKEINTRESOURCE(IDI_ICON1));

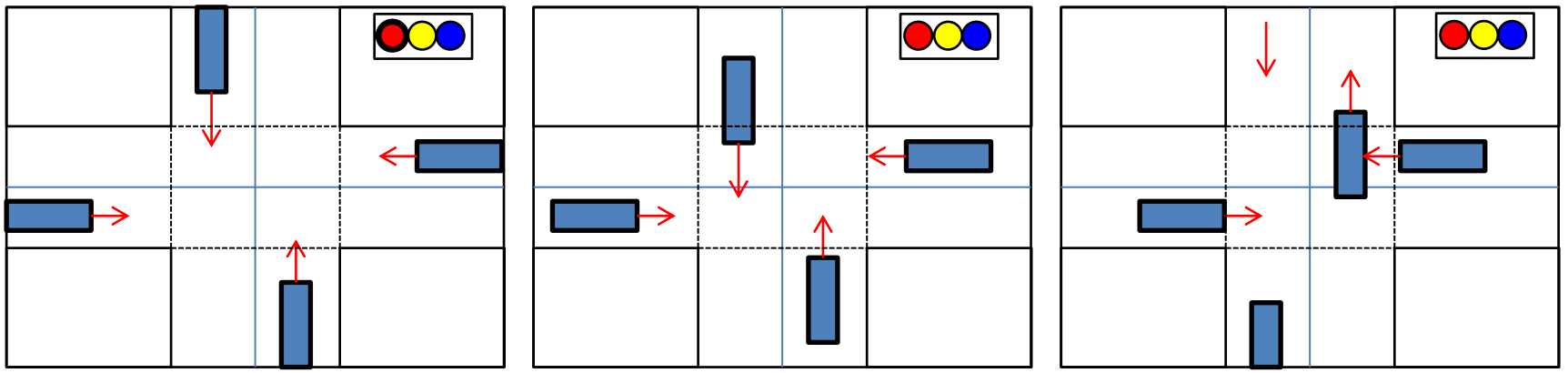
- 두 함수의 첫 번째 인자에 "NULL"대신 WinMain()함수의 첫 번째 인자인 "hInstance"를 쓰며 두 번째 인자에 만든 리소스 ID를 쓴다.
- 두 번째 인자로 리소스명을 바로 쓰면 안되고, MAKEINTRESOURCE 매크로로 변환해야 한다.
  - LPCTSTR MAKEINTRESOURCE (WORD wInt);
    - 정수 타입의 리소스 id를 문자열로 변환한다.
    - 리턴값: 문자열
    - wInt: 리소스의 정수형 ID

- 해당 리소스 헤더파일을 추가한다.

실습 4-1

- **제목**
  - 실습 3-3에 메뉴 붙이기

- **내용**
  - 3-3 사거리 자동차 실습에 도로 외의 부분에 신호등을 그린다.
    - 빨강: 사거리 앞에 무조건 선다.
    - 파랑: 움직이기 시작하고 사거리에서 한대씩 지나간다.
    - 노랑: 1초 후에 모든 자동차가 선다.
  - **메뉴를 만든다:**
    - 게임: Start/end (자동차들이 움직이기 시작/멈춤)
    - 속도: 가속/감속 (자동차들의 속도를 가속/감속)
    - 신호등: 빨강/파랑/노랑 (위의 신호등 내용을 메뉴로 실행) - 선택된 신호등 둘레를 두껍게 그린다.



실습 4-2

- **제목**
  - 실습 3-5에 메뉴 붙이기
- **내용**
  - 벽돌 깨기 게임에 메뉴를 붙인다.
  - **메뉴를 만든다.**
    - 게임: Start/end (공 튀기기 시작/프로그램 종료)
    - 속도: Fast/Medium/Slow (공의 이동 속도가 빠르게/중간/느리게)
    - 벽돌 단: 2/3/4 (벽돌을 2단/3단/4단)

실습 4-3

더블 버퍼링 (Double Buffering)

- **더블 버퍼링**
  - 화면 출력용 DC는 GDI 함수를 사용할 때마다 바로 화면에 출력되기 때문에 복잡한 그림을 그리기 위해 많은 GDI 함수를 사용하면 화면이 깜빡이는 현상이 발생한다.
  - DC를 한 개 더 사용하여 그림을 비트맵 객체에 그려 깜빡이는 현상을 발생하지 않게 할 수 있다.

- **추가된 메모리 디바이스 컨텍스트가 추가된 버퍼 역할을 하기 때문에 이 방법을 더블버퍼링이라 부름**
  - 메모리 디바이스 컨텍스트를 생성하여 하나 더 사용
  - 추가된 메모리 디바이스 컨텍스트에 그리기를 원하는 그림들을 모두 출력
  - 화면 디바이스 컨텍스트로 한꺼번에 옮기는 방법을 이용

- **메모리 DC 생성하기**
  - HDC **CreateCompatibleDC** (HDC hdc);
    - hdc에 호환되는 메모리 디바이스 컨텍스트 생성
    - hdc: dc 핸들
  - BOOL **DeleteDC** (HDC hdc);
    - 생성한 메모리 DC를 제거한다.

더블 버퍼링 (Double Buffering)

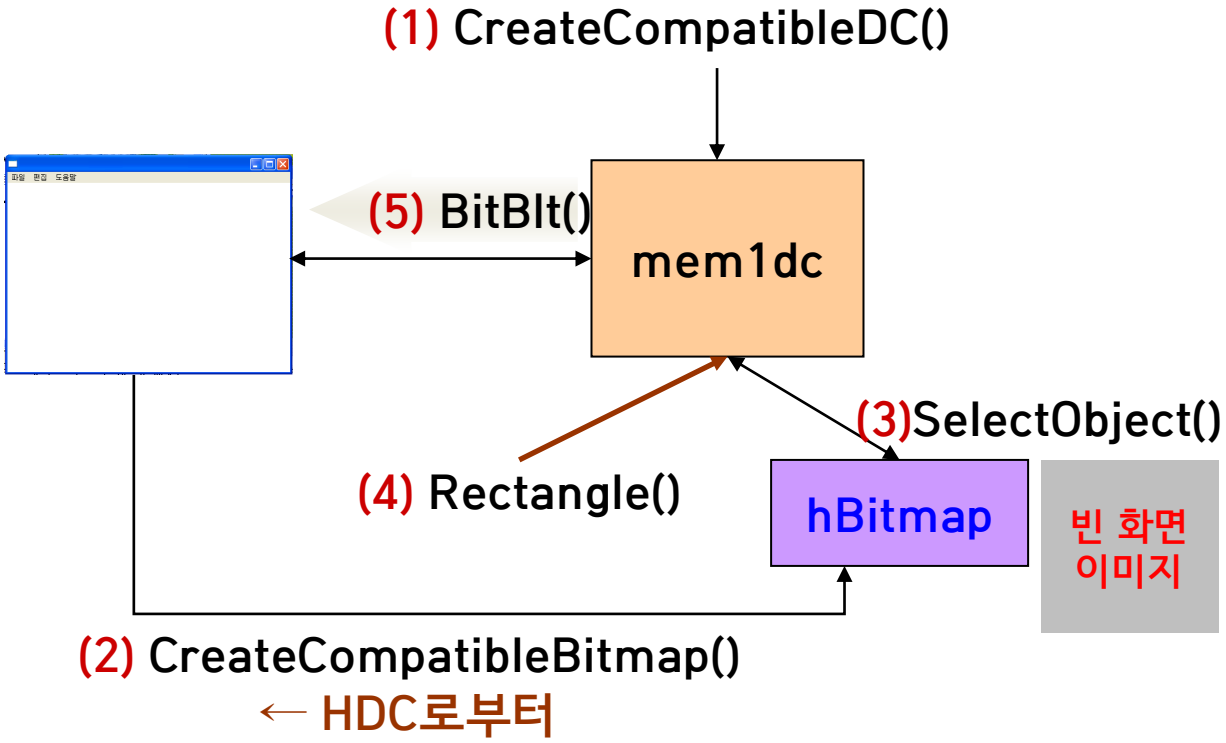
- **생성한 DC는 비트맵과 연결해서 그려야 한다.**
  - CreateCompatibleDC로 생성한 DC는 출력 대상이 없이 그리기 특성만 가지고 있는 DC
  - 따라서, 반드시 메모리 DC를 비트맵 객체와 연결하여 그리기를 진행해야 한다.

  - 즉, 다음의 단계로 더블 버퍼링을 진행한다.
    - **즉, DC와 호환되는 메모리 DC를 만든다.**
    - **그리기 특성을 가지고 있는 비트맵 객체를 만든다.**
    - **메모리 DC와 비트맵 객체를 연결한다.**
    - **메모리 DC에 모든 그리기를 진행한다.**
    - **메모리 DC의 내용을 화면 DC에 복사한다.**

- **비트맵 오브젝트 생성하기**
  - HBITMAP **CreateCompatibleBitmap** (HDC hdc, int nWidth, int nHeight);
    - hdc와 호환되는 비트맵을 생성하여 반환
    - 생성된 비트맵은 hdc와 호환되는 어떤 메모리 DC에서도 선택되어질 수 d있다.
    - hdc: DC 핸들
    - nWidth/nHeight: 작성하는 비트맵의 가로/세로 사이즈
  - BOOL **DeleteObject** (GDIOBJ hObject);
    - 생성한 비트맵 객체를 제거한다.

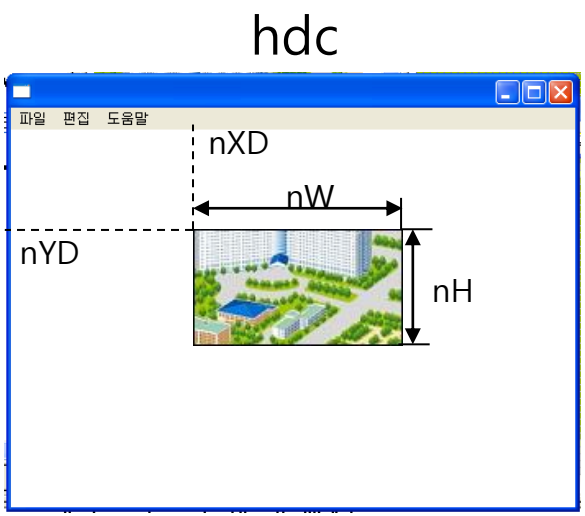
더블 버퍼링 (Double Buffering)

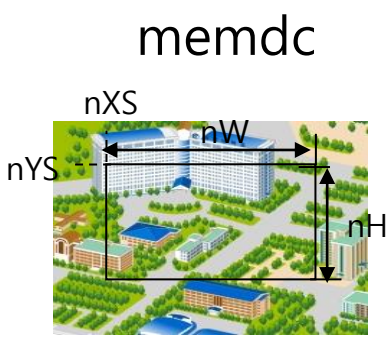
- 더블 버퍼링 기법



더블 버퍼링 (Double Buffering)

- **DC간의 영역 복사 함수**
  - BitBlt 함수
    - DC 간의 영역 고속 복사, 메모리 DC의 표면에 그려져 있는 비트맵을 화면 DC로 복사하여 비트맵을 화면에 출력
    - BOOL **BitBlt** (HDC hdc, int nXD, int nYD, int nW, int nH,
      HDC memdc, int nXS, int nYS, DWORD dwRop);
      - hdc: 복사 대상 DC
      - nXD, nYDest: 복사 대상의 x, y 좌표 값
      - nW, nHeight: 복사 대상의 폭과 높이
      - memdc: 복사 소스 DC
      - nXS, nYS: 복사 소스의 좌표
      - dwRop: 래스터 연산 방법
        - » SRCCOPY: 소스값을 그대로 칠한다.





더블 버퍼링 (Double Buffering)

- StretchBlt 함수
  - DC간의 이미지 확대 또는 축소하여 복사
  - BOOL **StretchBlt** (HDC hdc, int nXD, int nYD, int nW, int nH,
    HDC memdc, int nXS, int nYS, int nWS, int nHS, DWORD dwRop);
    - hdc: 복사대상 DC
    - nXD, nYD: 복사대상 DC x, y 좌표값
    - nW, nH: 복사대상 DC의 폭과 높이
    - memdc: 복사소스 DC
    - nXS, nYS: 복사소스 DC의 x, y 좌표값
    - nWS, nHS복사소스 DC의 폭과 높이
    - dwRop: 래스터 연산 방법

더블 버퍼링 (Double Buffering)

- 사용 예)

```
HDC memDC;
HBITMAP hBitmap;

case WM_PAINT:
	hdc = BeginPaint(hwnd, &ps);
	memDC = CreateCompatibleDC (hdc);
	hBitmap = CreateCompatibleBitmap (hdc, rt.right, rt.bottom);

	SelectObject (memDC, (HBITMAP) hBitmap);

	Rectangle (memDC, 0, 0, rt.right, rt.bottom);
	DrawBlock (memDC, blockYnum+1, blockXnum);
	Rectangle (memDC, rect.left, rect.top, rect.right, rect.bottom);

	Ellipse (memDC, s_ellipse.left, s_ellipse.top, s_ellipse.right, s_ellipse.bottom);

	BitBlt (hdc, 0, 0, rt.right, rt.bottom, memDC, 0, 0, SRCCOPY);

	DeleteDC (memDC);
	DeleteObject (hBitmap);

	EndPaint(hwnd, &ps);
break;
```

더블 버퍼링 (Double Buffering)

```
case WM_MOUSEMOVE:
	hdc = GetDC(hwnd);
	if (Selection) {
		mx = LOWORD(lParam);
		my = HIWORD(lParam);

		rect.left = mx;
		rect.right = mx + size;

		InvalidateRect (hwnd, NULL, false);

	}
	ReleaseDC(hwnd, hdc);
break;
```