제8장 단일 문서 / 다중 문서 프로그램

윈도우 프로그래밍 설계

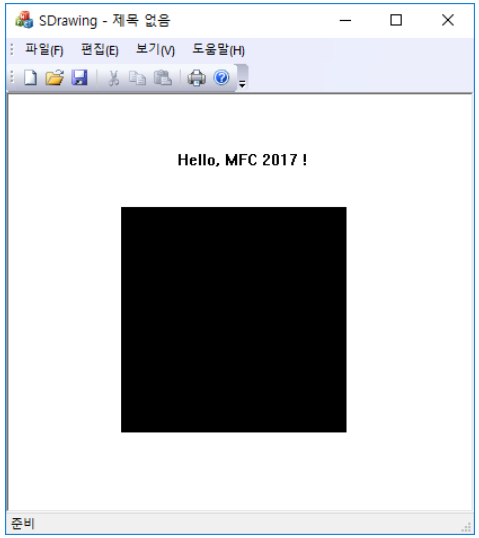
청주대학교 전자공학과
한철수

목차

- 프로그램 작성의 이해
- 단일 문서 프로그램 작성
- 다중 문서 프로그램 작성

단일 윈도우(Single Window)

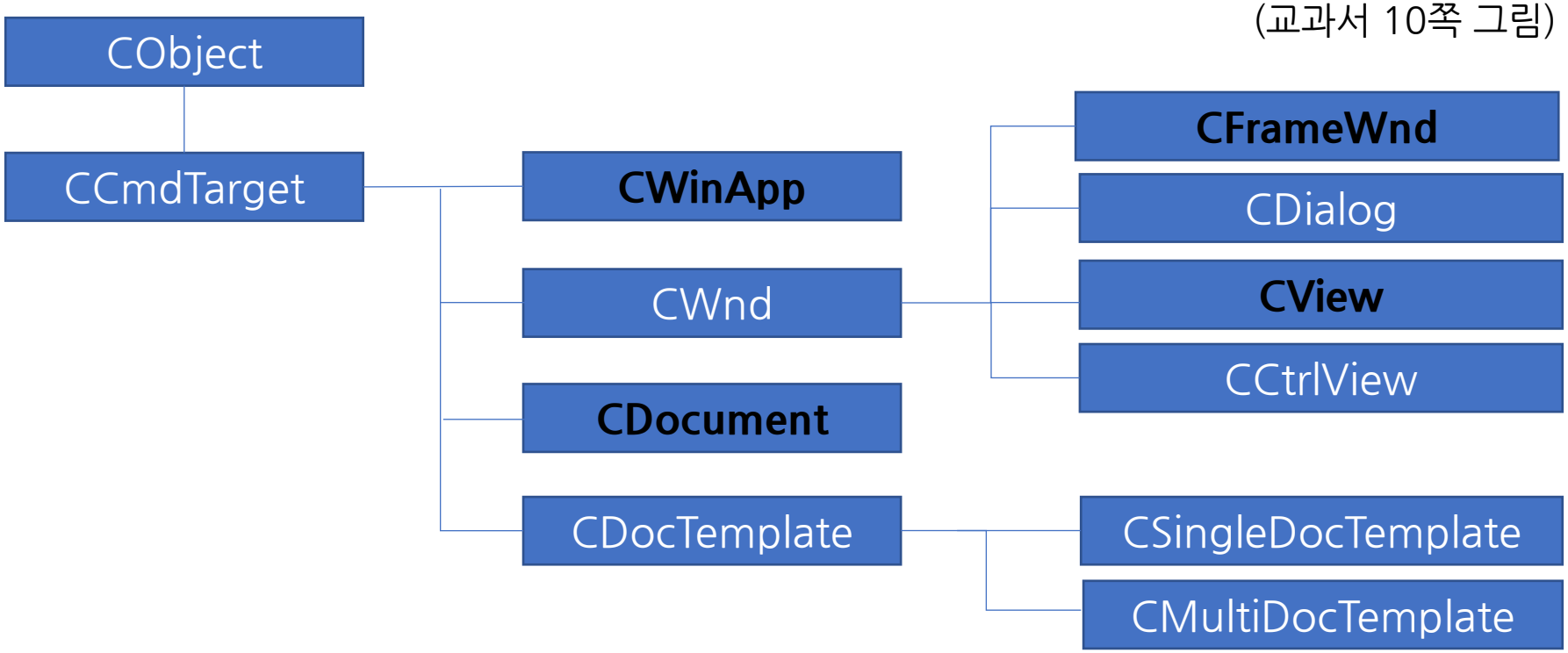
- MFC는 기본적으로 4개의 클래스가 모여서 하나의 윈도우(프로그램)를 생성함.
  - CFrameWnd
    - 윈도우 프레임을 관리함.
  - CView
    - 데이터를 화면에 보여주거나 사용자와 상호작용함.
  - CDocument
    - 데이터를 읽거나 저장함.
  - CWinApp
    - 프로그램을 구동 시킴.



- 대화상자 기반 프로그램에는 CDocument 클래스가 없고 CFrameWnd 클래스와 Cview 클래스를 대신하여 CDialog 클래스가 있음.
  - CDialog
  - CWinApp

MFC 클래스 구성도

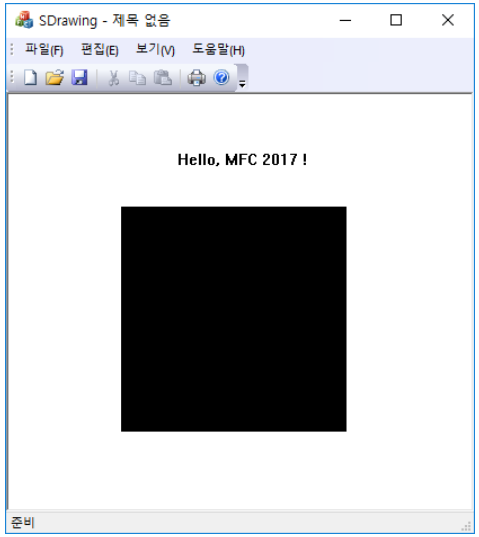
(교과서 10쪽 그림)



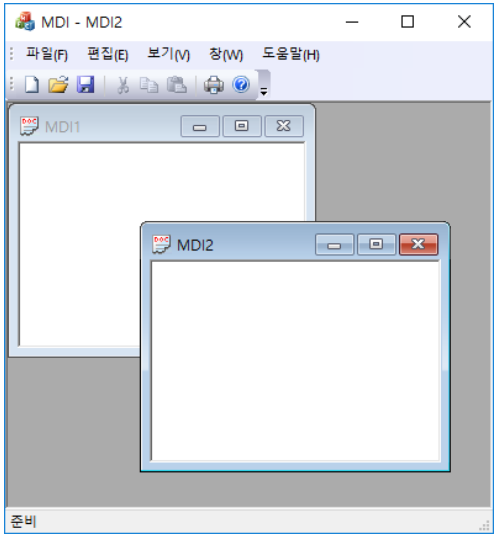
- 각 클래스에 대한 설명은 교과서 10~11쪽을 참고할 것.
- 더 자세한 내용은 아래 웹페이지를 참고하거나 인터넷 검색을 이용할 것.
  - https://docs.microsoft.com/en-us/cpp/mfc/reference/mfc-classes

MFC 프로그램 종류

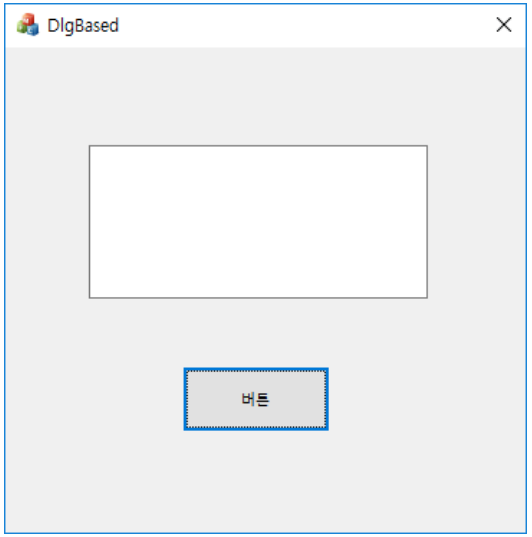
- 단일 문서 인터페이스 프로그램
  - SDI(Single Document Interface) 프로그램
- 다중 문서 인터페이스 프로그램
  - MDI(Multi Document Interface) 프로그램
- 대화상자 기반 프로그램
  - Dialog-based 프로그램



SDI



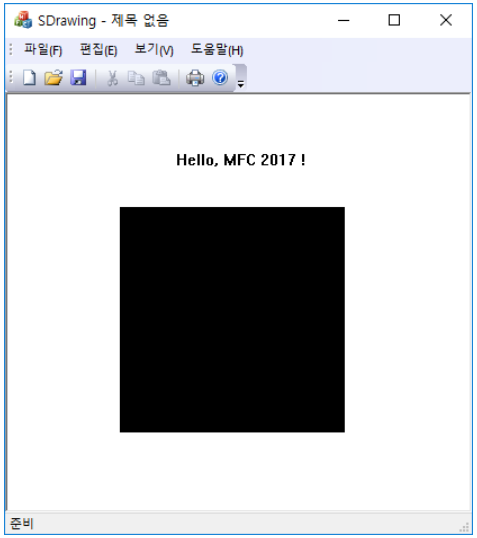
MDI



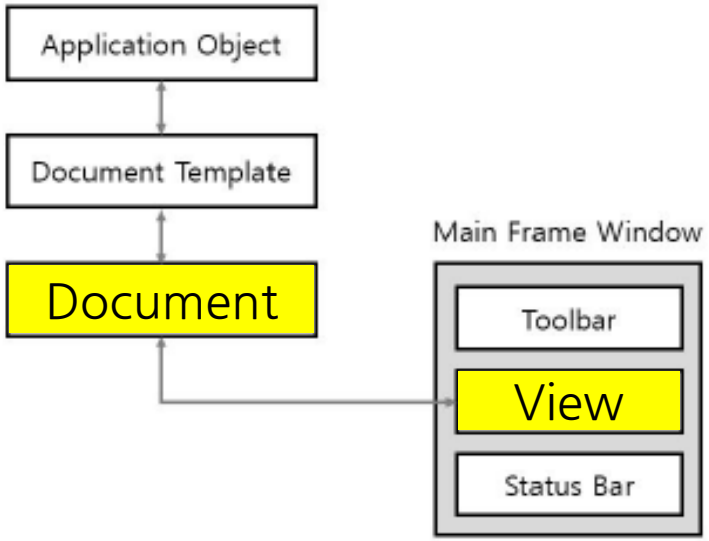
Dialog-based

SDI 프로그램

- SDI(단일 문서 인터페이스) 프로그램은 한 번에 하나의 문서 작업만이 가능함.
- CSingleDocTemplate 클래스를 이용해서 Document Template 객체를 구성함.
- Document 객체가 1개임.



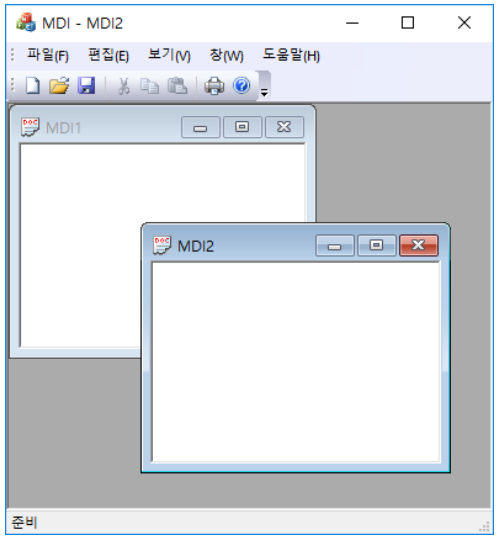
SDI



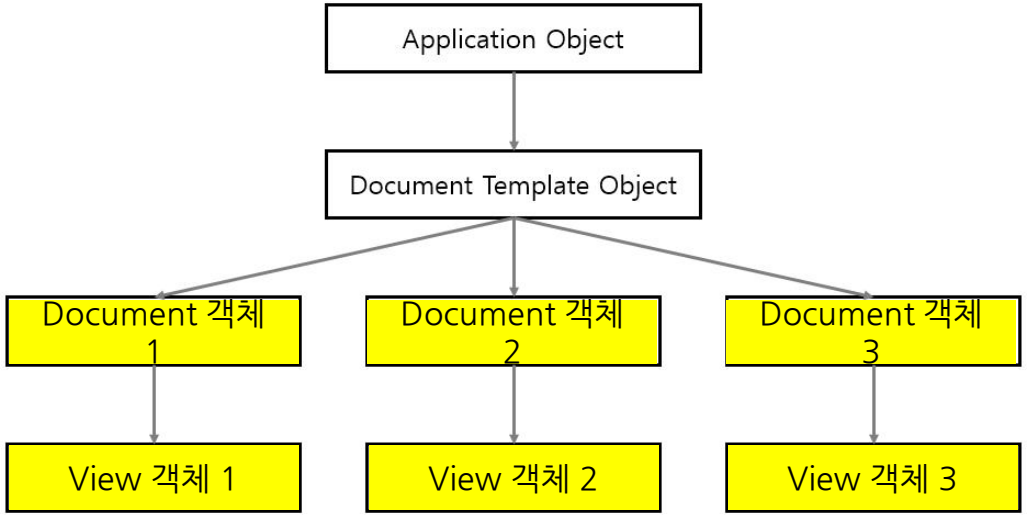
객체 구성 관계

MDI 프로그램

- MDI(다중 문서 인터페이스) 프로그램은 여러 개의 문서를 동시에 작업할 수 있음.
- CMultiDocTemplate 클래스를 이용해서 Document Template 객체를 구성함.
- 새 문서가 생성될 때 View와 Document 객체가 새로 생성됨.
- Document 객체가 여러 개일 수 있음.



MDI



객체 구성 관계

SDI 프로그램 작성

- 만들고자 하는 SDI 프로그램
  - 마우스를 클릭한채로 움직여서 그림을 그리는 프로그램
  - 사용하는 윈도우 메시지
    - WM_MOUSEMOVE
      - 마우스가 움직일 때 발생하는 메시지
    - WM_LBUTTONDOWN
      - 마우스 왼쪽 버튼을 클릭했을 때 발생하는 메시지

멤버 변수 선언

- View 클래스 내에 변수를 선언함.

```
public:
    int m_ptX;
    int m_ptY;
    int m_crColor;
    CRect m_reRect;
```

멤버 변수 초기화

- 선언한 변수는 클래스의 생성자에서 초기화함.
  - 생성자는 객체가 생성될 때 한번 실행됨.

```
CSDrawingView::CSDrawingView()
    : m_ptX(0)
    , m_ptY(0)
    , m_crColor(BLACK_BRUSH)
    , m_reRect(100,100,300,300)
{
    // TODO: 여기에 생성 코드를 추가합니다.

}
```

생성자 초기화 목록

WM_MOUSEMOVE 메시지 처리

- 메시지 처리 함수로 nFlags와 point가 넘어옴.
  - nFlags
    - 메시지 발생시의 키보드와 마우스의 상태 값이 저장되어 있음.
  - point
    - 메시지 발생시의 마우스 좌표 값(x, y)이 저장되어 있음.

```
void CSDrawingView::OnMouseMove(UINT nFlags, CPoint point)
{
    // TODO: 여기에 메시지 처리기 코드를 추가 및/또는 기본값을 호출합니다.
    if (nFlags & MK_LBUTTON)
    {
        CClientDC dc(this);

        dc.MoveTo(m_ptX, m_ptY);
        dc.LineTo(point.x, point.y);

        m_ptX = point.x;
        m_ptY = point.y;
    }

    CView::OnMouseMove(nFlags, point);
}
```

```
void CSDrawingView::OnMouseMove(UINT nFlags, CPoint point)
{
    // TODO: 여기에 메시지 처리기 코드를 추가 및/또는 기본값을 호출합니다.
    if (nFlags & MK_LBUTTON)
    {
        CClientDC dc(this);

        dc.MoveTo(m_ptX, m_ptY);
        dc.LineTo(point.x, point.y);

        m_ptX = point.x;
        m_ptY = point.y;
    }

    CView::OnMouseMove(nFlags, point);
}
```

마우스 왼쪽 버튼이 눌렸으면 참이 됨.

Value	Meaning
MK_CONTROL 0x0008	The CTRL key is down.
MK_LBUTTON 0x0001	The left mouse button is down.
MK_MBUTTON 0x0010	The middle mouse button is down.
MK_RBUTTON 0x0002	The right mouse button is down.
MK_SHIFT 0x0004	The SHIFT key is down.
MK_XBUTTON1 0x0020	The first X button is down.
MK_XBUTTON2 0x0040	The second X button is down.

- 마우스 왼쪽 버튼이 눌렸으면 if문의 괄호 부분이 if (MK_LBUTTON)이 되는데
  C 언어와 C++ 언어에서는 0이 아닌 값은 참으로 판정하기 때문에, 결국 if (MK_LBUTTON)은
  if (참)이 됨.

```
void CSDrawingView::OnMouseMove(UINT nFlags, CPoint point)
{
    // TODO: 여기에 메시지 처리기 코드를 추가 및/또는 기본값을 호출합니다.
    if (nFlags & MK_LBUTTON)
    {
        CClientDC dc(this);

        dc.MoveTo(m_ptX, m_ptY);
        dc.LineTo(point.x, point.y);

        m_ptX = point.x;
        m_ptY = point.y;
    }

    CView::OnMouseMove(nFlags, point);
}
```

선 시작 위치 설정.

시작 위치부터 (x, y)까지 선을 긋기.
교재 152쪽 참고.

마우스 좌표 값을 변수에 저장하여 다음 번 메시지 처리 시 시작 위치로 사용함.

WM_LBUTTONDOWN 메시지 처리

```
void CSDrawingView::OnLButtonDown(UINT nFlags, CPoint point)
{
    // TODO: 여기에 메시지 처리기 코드를 추가 및/또는 기본값을 호출합니다.
    m_ptX = point.x;
    m_ptY = point.y;

    if (m_reRect.PtInRect(point))
    {
        if (m_crColor == BLACK_BRUSH)
        {
            m_crColor = WHITE_BRUSH;
        }
        else
        {
            m_crColor = GRAY_BRUSH;
        }
        InvalidateRect(m_reRect);
    }

    CView::OnLButtonDown(nFlags, point);
}
```

마우스 좌표 값을 변수에 저장함.

point가 m_reRect 영역의 내부이면 참을 반환하고
내부가 아니면 거짓을 반환함.
교재 153쪽 참고.

m_reRect 영역만 화면을 갱신함.
교과서 154쪽 참고.

OnDraw() 함수

- SDI/MDI 프로그램에는 WM_PAINT 메시지 처리함수로 OnDraw() 함수가 준비되어 있음.

```
void CSDrawingView::OnDraw(CDC* pDC)
{
    CSDrawingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    if (!pDoc)
        return;

    // TODO: 여기에 원시 데이터에 대한 그리기 코드를 추가합니다.
    pDC->TextOutW(150, 50, _T("Hello, MFC 2017 !"));
    pDC->SelectStockObject(m_crColor);
    pDC->Rectangle(m_reRect);
}
```

DC를 사용하려면 주석을 해제함.

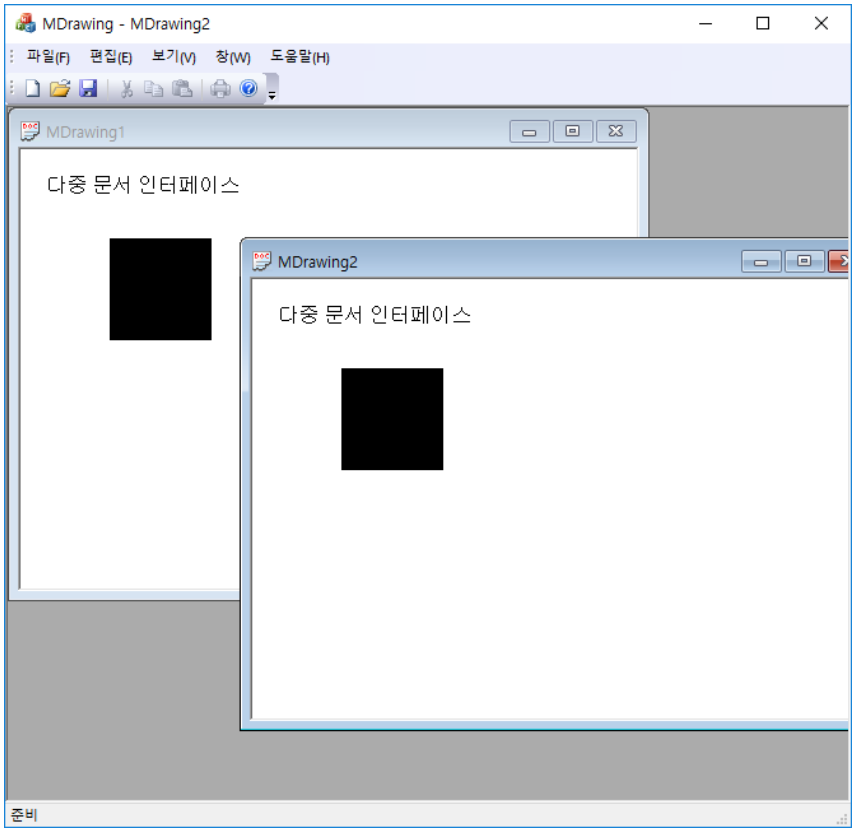
DC를 이용해 글자를 출력함.

미리 정의된 GDI 객체를 DC에 적용시킴.
교과서 153쪽 참고.

DC를 이용해 사각형을 그림.

MDI 프로그램 작성

- 만들고자 하는 MDI 프로그램
  - "새로 만들기"를 누를 때마다 기본 글자와 그림이 그려진 자식 창(Child Window)이 열리는 프로그램



멤버 변수 선언

- MDrawingDoc 클래스에 추가할 변수
  - MDrawingDoc.h 파일의 “특성입니다. public:” 밑에 추가

```
int m_ptX;
int m_ptY;
```

- MDrawingView 클래스에 추가할 변수
  - MDrawingView.h 파일의 “특성입니다. public:” 밑에 추가

```
int m_ptX;
int m_ptY;
```

멤버 변수 초기화

- 만들고자 하는 프로그램
  - "새로 만들기"를 누를 때마다 기본 글자와 그림이 그려진 자식 창(Child Window)이 열리는 프로그램
- MDrawingDoc 클래스의 멤버 변수 초기화
  - OnNewDocument() 함수
    - "새로 만들기"를 누를 때 호출되는 함수

```
m_ptX = 100;
m_ptY = 100;
```

- MDrawingView 클래스의 멤버 변수 초기화
  - OnInitialUpdate() 함수
    - View가 표시되기 전, View가 Document와 연결된 직후에 호출되는 함수

```
CMDrawingDoc* pDoc = GetDocument();

m_ptX = pDoc->m_ptX;
m_ptY = pDoc->m_ptY;
```

View와 연결된 Document의 포인터를 반환함.

OnDraw() 함수

- SDI/MDI 프로그램에는 WM_PAINT 메시지 처리함수로 OnDraw() 함수가 준비되어 있음.

```
void CMDrawingView::OnDraw(CDC* pDC)
{
    CMDrawingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);
    if (!pDoc)
        return;

    // TODO: 여기에 원시 데이터에 대한 그리기 코드를 추가합니다.
    RECT rect;
    rect.left = m_ptX - 30;
    rect.top = m_ptY - 30;
    rect.bottom = m_ptX + 50;
    rect.right = m_ptY + 50;
    pDC->SelectStockObject(BLACK_BRUSH);
    pDC->Rectangle(&rect);

    pDC->TextOutW(20, 20, _T("다중 문서 인터페이스"));
}
```

DC를 사용하려면 주석을 해제함.

구조체 변수 rect를 선언함.

변수 rect에 값을 저장함.

미리 정의된 GDI 객체를 DC에 적용시킴.
교과서 153쪽 참고.

DC를 이용해 사각형을 그림.

DC를 이용해 문자열을 출력함.

질문

Q&A