마우스와 키보드

(10주차)

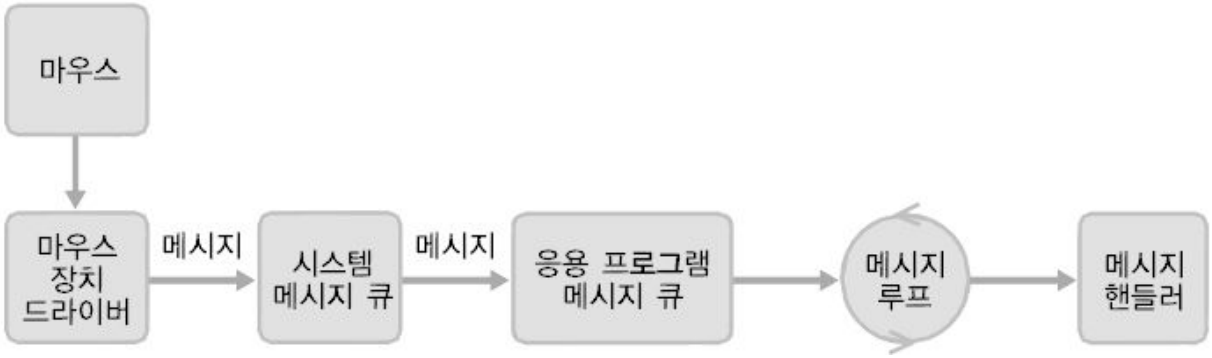
학습개요

- 학습 목표
  - 마우스 메시지 처리 기법을 익힌다.
  - 키보드 메시지 처리 기법을 익힌다.

- 학습 내용
  - 마우스
  - 키보드
  - 실습

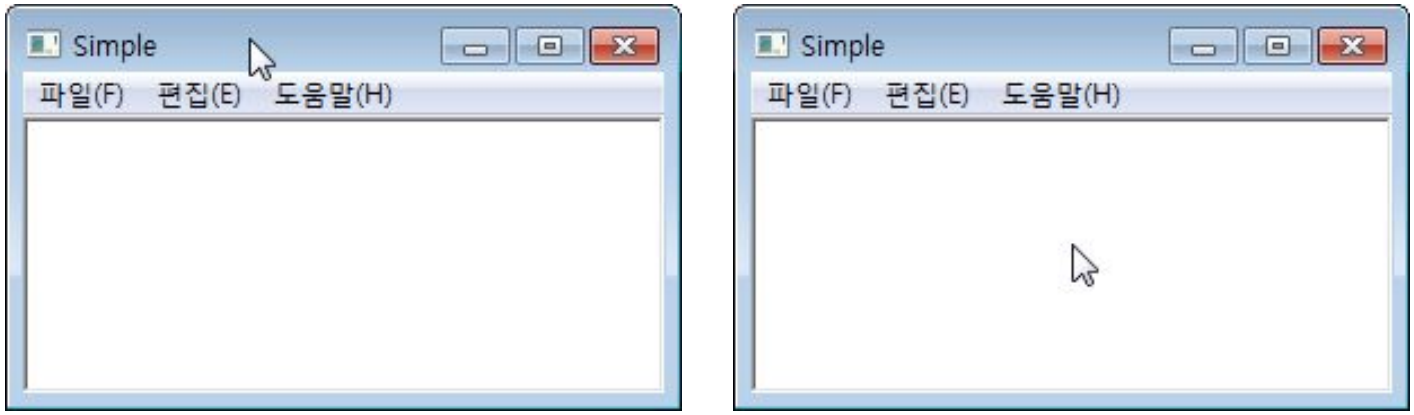
마우스 기초 (1)

- 마우스 처리
  - 윈도우 운영체제는 마우스와 관련된 모든 변화를 메시지 형태로 프로그램에 전달한다.



마우스 기초 (2)

- 마우스 메시지 전달
  - 마우스 메시지는 원칙적으로 마우스 커서 밑에 있는 윈도우가 받는다.



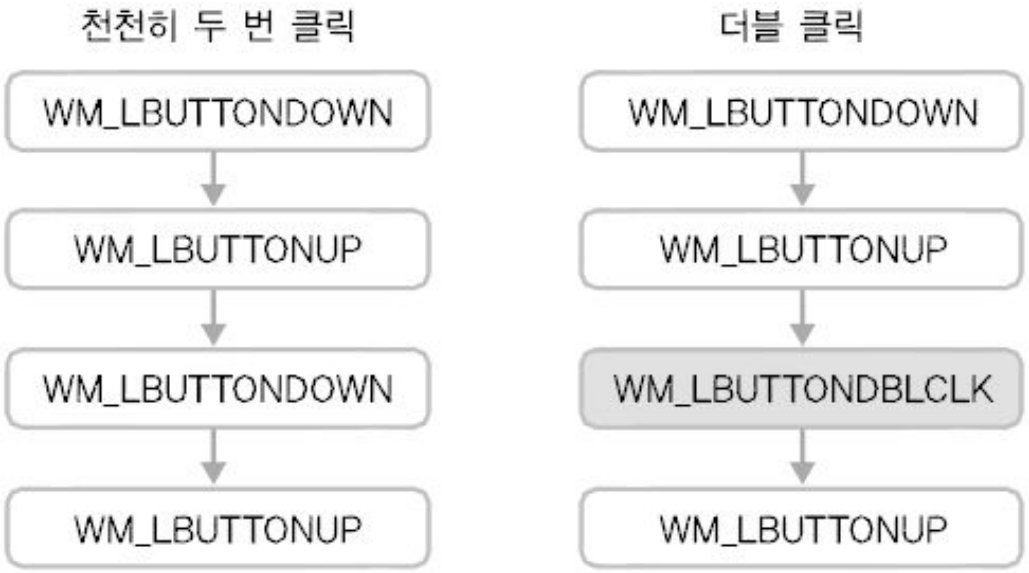
클라이언트 영역 마우스 메시지 (1)

- 클라이언트 영역 마우스 메시지

메시지	발생 시점
WM_LBUTTONDOWN	마우스 왼쪽 버튼을 누를 때
WM_LBUTTONUP	마우스 왼쪽 버튼을 뗄 때
WM_LBUTTONDBLCLK	마우스 왼쪽 버튼을 더블 클릭할 때
WM_MBUTTONDOWN	마우스 가운데 버튼을 누를 때
WM_MBUTTONUP	마우스 가운데 버튼을 뗄 때
WM_MBUTTONDBLCLK	마우스 가운데 버튼을 더블 클릭할 때
WM_RBUTTONDOWN	마우스 오른쪽 버튼을 누를 때
WM_RBUTTONUP	마우스 오른쪽 버튼을 뗄 때
WM_RBUTTONDBLCLK	마우스 오른쪽 버튼을 더블 클릭할 때
WM_MOUSEMOVE	마우스를 움직일 때

클라이언트 영역 마우스 메시지 (2)

- 마우스 왼쪽 버튼 두 번 클릭 VS 더블 클릭
  메시지 발생순서 비교



클라이언트 영역 마우스 메시지 (3)

- 클라이언트 영역 마우스 메시지 핸들러

메시지	메시지 맵 매크로	메시지 핸들러
WM_LBUTTONDOWN	ON_WM_LBUTTONDOWN()	OnLButtonDown()
WM_LBUTTONUP	ON_WM_LBUTTONUP()	OnLButtonUp()
WM_LBUTTONDBLCLK	ON_WM_LBUTTONDBLCLK()	OnLButtonDblClk()
WM_MBUTTONDOWN	ON_WM_MBUTTONDOWN()	OnMButtonDown()
WM_MBUTTONUP	ON_WM_MBUTTONUP()	OnMButtonUp()
WM_MBUTTONDBLCLK	ON_WM_MBUTTONDBLCLK()	OnMButtonDblClk()
WM_RBUTTONDOWN	ON_WM_RBUTTONDOWN()	OnRButtonDown()
WM_RBUTTONUP	ON_WM_RBUTTONUP()	OnRButtonUp()
WM_RBUTTONDBLCLK	ON_WM_RBUTTONDBLCLK()	OnRButtonDblClk()
WM_MOUSEMOVE	ON_WM_MOUSEMOVE()	OnMouseMove()

클라이언트 영역 마우스 메시지 (4)

- 메시지 핸들러 형태

```
afx_msg void On*(UINT nFlags, CPoint point);
   ①              ②            ③
```

  - nFlags
    - 메시지가 생성될 당시의 키보드나 마우스 버튼의 상태를 나타내는 비트 마스크

비트 마스크	의미
MK_CONTROL	[Ctrl] 키 누름
MK_SHIFT	[Shift] 키 누름
MK_LBUTTON	마우스 왼쪽 버튼 누름
MK_MBUTTON	마우스 가운데 버튼 누름
MK_RBUTTON	마우스 오른쪽 버튼 누름

클라이언트 영역 마우스 메시지 (5)

- nFlags와 비트 마스크 연산 예

```
void CChildView::OnLButtonDown(UINT nFlags, CPoint point)
{
  if(nFlags & MK_SHIFT){ // [Shift] 키가 눌렸다면
    ...
  }
}
```

- point
  - 메시지가 생성될 당시의 마우스 커서 위치(클라이언트 좌표)

클라이언트 영역 마우스 메시지 (6)

- 사용 예

```
void CChildView::OnLButtonDown(UINT nFlags, CPoint point)
{
    CClientDC dc(this);
    dc.SetMapMode(MM_LOMETRIC); // 매핑 모드를 변경한다.
    CPoint pt = point; // point 객체를 복사한다.
    dc.DPtoLP(&pt); // 장치 좌표를 논리 좌표로 변환한다.
    dc.Rectangle(pt.x-100, pt.y+100, pt.x+100, pt.y-100);
}
```

클라이언트 영역 마우스 메시지 (7)

- 마우스 캡처
  - 용도
    - 마우스 캡처를 하면 마우스 커서의 위치에 관계없이 모든 마우스 메시지를 특정 윈도우가 받을 수 있다.
  - 관련 함수

API 함수	MFC 함수	의미
SetCapture()	CWnd::SetCapture()	마우스 캡처를 시작한다.
ReleaseCapture()	없음	마우스 캡처를 해제한다.
GetCapture()	CWnd::GetCapture()	어느 윈도우가 현재 마우스 캡처를 하고 있는지 알아낸다.

비클라이언트 영역 마우스 메시지 (1)

- 비클라이언트 영역 마우스 메시지

메시지	발생 시점
WM_NCLBUTTONDOWN	왼쪽 버튼을 누를 때
WM_NCLBUTTONUP	왼쪽 버튼을 뗄 때
WM_NCLBUTTONDBLCLK	왼쪽 버튼을 더블 클릭할 때
WM_NCMBUTTONDOWN	가운데 버튼을 누를 때
WM_NCMBUTTONUP	가운데 버튼을 뗄 때
WM_NCMBUTTONDBLCLK	가운데 버튼을 더블 클릭할 때
WM_NCRBUTTONDOWN	오른쪽 버튼을 누를 때
WM_NCRBUTTONUP	오른쪽 버튼을 뗄 때
WM_NCRBUTTONDBLCLK	오른쪽 버튼을 더블 클릭할 때
WM_NCMOUSEMOVE	마우스를 움직일 때

비클라이언트 영역 마우스 메시지 (2)

- 비클라이언트 영역 마우스 메시지 핸들러

메시지	메시지맵 매크로	메시지 핸들러
WM_NCLBUTTONDOWN	ON_WM_NCLBUTTONDOWN()	OnNcLButtonDown()
WM_NCLBUTTONUP	ON_WM_NCLBUTTONUP()	OnNcLButtonUp()
WM_NCLBUTTONDBLCLK	ON_WM_NCLBUTTONDBLCLK()	OnNcLButtonDblClk()
WM_NCMBUTTONDOWN	ON_WM_NCMBUTTONDOWN()	OnNcMButtonDown()
WM_NCMBUTTONUP	ON_WM_NCMBUTTONUP()	OnNcMButtonUp()
WM_NCMBUTTONDBLCLK	ON_WM_NCMBUTTONDBLCLK()	OnNcMButtonDblClk()
WM_NCRBUTTONDOWN	ON_WM_NCRBUTTONDOWN()	OnNcRButtonDown()
WM_NCRBUTTONUP	ON_WM_NCRBUTTONUP()	OnNcRButtonUp()
WM_NCRBUTTONDBLCLK	ON_WM_NCRBUTTONDBLCLK()	OnNcRButtonDblClk()
WM_NCMOUSEMOVE	ON_WM_NCMOUSEMOVE()	OnNcMouseMove()

비클라이언트 영역 마우스 메시지 (3)

- 메시지 핸들러 형태

```
afx_msg void OnNc*(UINT nHitTest, CPoint point);
                        ①          ②
```

  - nHitTest
    - 메시지가 생성될 당시의 마우스 커서 위치를 나타내는 상수값 → 다음 페이지 표 참조
  - point
    - 메시지가 생성될 당시의 마우스 커서 위치(스크린 좌표)
      - 클라이언트 좌표로 변환하려면 CWnd::ScreenToClient() 함수를 사용

비클라이언트 영역 마우스 메시지 (4)

- nHitTest

상수값	의미
HTCAPTION	타이틀 바
HTCLIENT	클라이언트 영역
HTCLOSE	종료 버튼
HTHSCROLL	가로 스크롤 바
HTMENU	메뉴
HTMAXBUTTON 또는 HTZOOM	최대화 버튼
HTMINBUTTON 또는 HTREDUCE	최소화 버튼
HTSYSMENU	시스템 메뉴
HTVSCROLL	세로 스크롤 바

마우스 커서

- 마우스 커서 변경

```
HCURSOR SetCursor(HCURSOR hCursor);
```

  - hCursor
    - 커서 리소스를 가리키는 핸들값
    - 다음 함수의 리턴값을 대입
      - CWinApp::LoadStandardCursor()
      - CWinApp::LoadCursor()

마우스 커서 위치 추적 (1)

- 마우스 커서 위치 추적
  - TrackMouseEvent() 함수

```
BOOL TrackMouseEvent(LPTRACKMOUSEEVENT lpEventTrack);
```

  - TRACKMOUSEEVENT 구조체

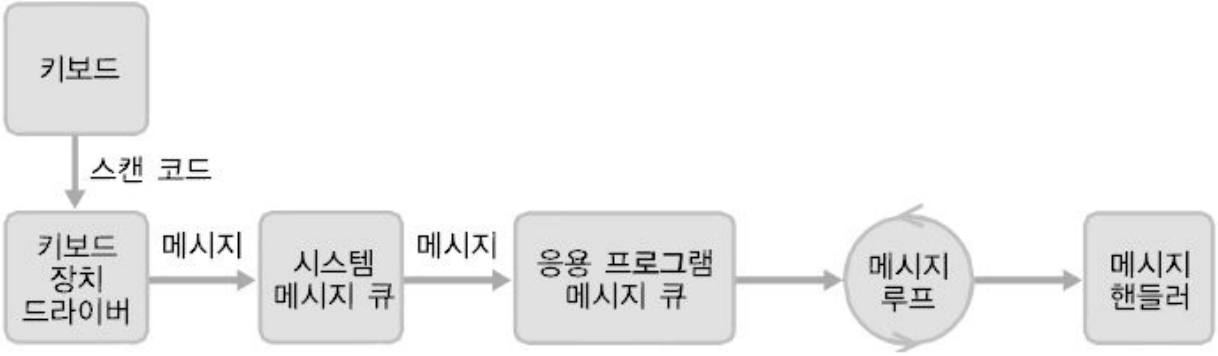
```
typedef struct tagTRACKMOUSEEVENT {
    DWORD cbSize;
    DWORD dwFlags;
    HWND hwndTrack;
    DWORD dwHoverTime;
} TRACKMOUSEEVENT, *LPTRACKMOUSEEVENT;
```

마우스 커서 위치 추적 (2)

- 마우스 커서 위치 추적하기
  - 주의점
    - TrackMouseEvent() 함수로 요청한 WM_MOUSELEAVE 메시지는 한 번만 발생하므로 필요하다면TrackMouseEvent() 함수를 다시 호출해야 한다.

키보드 포커스 (1)

- 윈도우의 키보드 메시지 처리
  - 윈도우 운영체제는 키보드와 관련된 모든 이벤트를 프로그램에 메시지 형태로 전달한다.

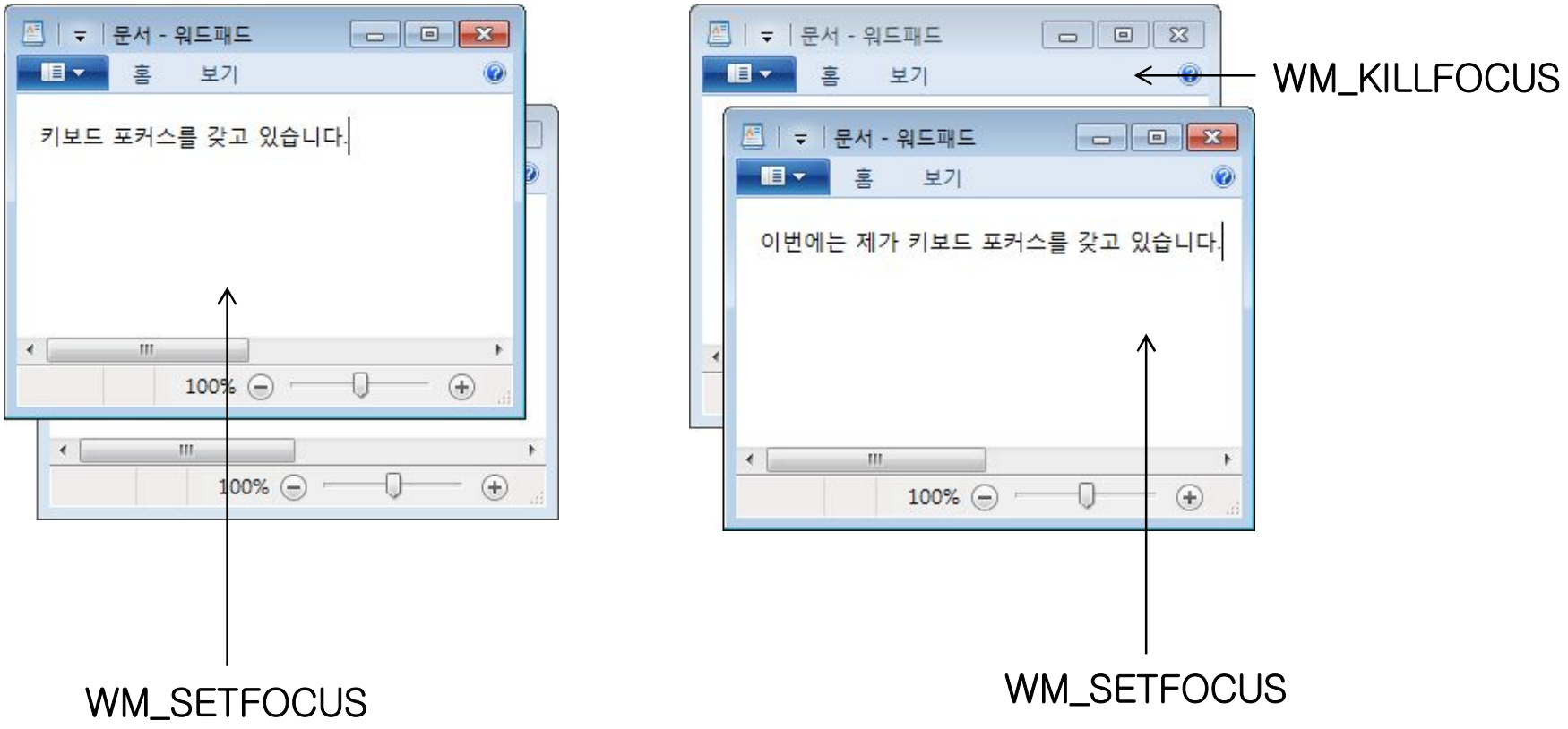


키보드 포커스 (2)

- 키보드 메시지 전달
  - 키보드 메시지는 키보드 포커스를 가진 윈도우가 받음
- 키보드 포커스
  - 활성 윈도우 또는 활성 윈도우의 자식 윈도우가 가지는 일종의 속성

키보드 포커스 (3)

- 키보드 포커스 변화



키보드 포커스 (4)

- 캐럿 함수 - MFC

함수 이름	기능
CreateCaret()	비트맵을 이용하여 캐럿을 생성한다.
CreateGrayCaret()	회색 직사각형 캐럿을 생성한다.
CreateSolidCaret()	검은색 직사각형 캐럿을 생성한다.
ShowCaret()	캐럿이 보인다.
HideCaret()	캐럿을 숨긴다.
GetCaretPos()	캐럿의 위치(클라이언트 좌표)를 얻는다.
SetCaretPos()	캐럿의 위치(클라이언트 좌표)를 설정한다.

키보드 포커스 (5)

- 캐럿 함수 - API

함수 이름	기능
::DestroyCaret()	캐럿을 파괴한다.
::GetCaretBlinkTime()	캐럿이 깜박이는 간격을 얻는다.
::SetCaretBlinkTime()	캐럿이 깜박이는 간격을 설정한다.

키 누름 메시지 (1)

- 키 누름 메시지(Keystroke Message)
  - 키보드를 누르거나 떼는 동작에 의해 발생하는 메시지
- 키 누름 메시지 종류

메시지	발생 시점
WM_KEYDOWN	[F10], [Alt] 이외의 키를 누를 때
WM_KEYUP	[F10], [Alt] 이외의 키를 뗄 때
WM_SYSKEYDOWN	[F10], [Alt], [Alt]+[키 조합]을 누를 때
WM_SYSKEYUP	[F10], [Alt], [Alt]+[키 조합]을 뗄 때

키 누름 메시지 (2)

- 키 누름 메시지 핸들러 형태

```
afx_msg void On*(UINT nChar, UINT nRepCnt, UINT nFlags);
                      ①          ②          ③
```

  - nChar
    - 키에 할당된 가상 키 코드값
  - nRepCnt
    - 키를 계속 누르고 있을 경우 1보다 큰 값을 가진다.
  - nFlags
    - 키와 관련된 부가적인 정보를 담고 있다.

키 누름 메시지 (3)

- 가상 키코드

가상 키 코드	해당 키	가상 키 코드	해당 키
VK_CANCEL	Ctrl-Break	VK_HOME	Home
VK_BACK	Backspace	VK_LEFT	←
VK_TAB	Tab	VK_UP	↑
VK_RETURN	Enter	VK_RIGHT	→
VK_SHIFT	Shift	VK_DOWN	↓
VK_CONTROL	Ctrl	VK_SNAPSHOT	Print Screen
VK_MENU	Alt	VK_INSERT	Insert
VK_PAUSE	Pause	VK_DELETE	Delete
VK_CAPITAL	Caps Lock	'0' ~ '9'	0 ~ 9
VK_ESCAPE	Esc	'A' ~ 'Z'	A ~ Z
VK_SPACE	Spacebar	VK_F1 ~ VK_F12	F1 ~ F12
VK_PRIOR	PgUp	VK_NUMLOCK	Num Lock
VK_NEXT	PgDn	VK_SCROLL	Scroll Lock
VK_END	End		

문자 메시지 (1)

- 문자 메시지 필요성
  - [R] 키를 누른 경우?

문자	가상 키 코드 조합
r	영문 입력 모드에서 [R] 또는 [Caps Lock]+[Shift]+[R] 키를 누른 경우
R	영문 입력 모드에서 [Caps Lock]+[R] 또는 [Shift]+[R] 키를 누른 경우
ㄱ	한글 입력 모드에서 [R] 키를 누른 경우
ㄲ	한글 입력 모드에서 [Shift]+[R] 키를 누른 경우

문자 메시지 (2)

- 문자 메시지 발생 시나리오

문자 메시지 (3)

- 문자 메시지 핸들러 형태

```
afx_msg void OnChar(UINT nChar, UINT nRepCnt, UINT nFlags) ;
                            ①          ②           ③
afx_msg void OnSysChar(UINT nChar, UINT nRepCnt, UINT nFlags) ;
                               ①          ②           ③
```

  - nChar
    - 키에 해당하는 문자 코드값을 가진다.
  - nRepCnt
    - 키를 계속 누르고 있을 경우 1보다 큰 값을 가진다.
  - nFlags
    - 키와 관련된 부가적인 정보를 담고 있다.

학습정리

- 마우스 메시지는 원칙적으로 마우스 커서 밑에 있는 윈도우가 받습니다.
- 클라이언트 영역 마우스 메시지 핸들러는 첫번째 인자로 메시지가 생성될 당시의 키보드나 마우스 버튼의 상태를 나타내는 비트 마스크 정보를 전달받으며, 두번째 인자로 메시지가 생성될 당시의 마우스 커서 위치(클라이언트 좌표)를 전달 받습니다.
- SetCapture(), ReleaseCapture() 함수를 이용하면 마우스 캡처를 통해 마우스 커서의 위치에 관계없이 모든 마우스 메시지를 특정 윈도우가 받을 수 있습니다.
- 비클라이언트 영역 마우스 메시지 핸들러는 첫번째 인자로 메시지가 생성될 당시의 마우스 커서 위치를 나타내는 상수값을 전달받고, 두번째 인자로 메시지가 생성될 당시의 마우스 커서 위치(스크린 좌표)를 전달 받습니다.
- 마우스 커서의 위치를 추적하기 위해 TrackMouseEvent() 함수로 요청한 WM_MOUSELEAVE 메시지는 한 번만 발생하므로 필요할 때마다 TrackMouseEvent() 함수를 다시 호출해야 한다.
- 키 누름 메시지는 키보드를 누르거나 떼는 동작에 의해 발생하며 메시지 핸들러의 첫번째 인자로 가상키 값이 전달되며, 문자 메시지는 메시지 핸들러의 첫번째 인자로 문자 코드값이 전달됩니다 .