8장 파일 입출력

2018년도 1학기 윈도우 프로그래밍

학습 목표

- 학습목표
  - 표준 입출력이 아닌 API에서 제공하는 파일 입출력을 배운다.
  - 공용 대화상자를 이용하여 파일 입출력을 배운다.
  - 그 외에 유용한 함수에 관하여 배운다.

- 내용
  - 파일 다루기
  - 공용 대화상자

1. 파일 다루기

- API 이용한 표준 입출력 및 파일 사용 방법
  - 파일을 만들고 열어준다. 열 때는 읽기용인지 쓰기용인지 명시
  - 열린 파일에는 텍스트를 쓰거나 읽는다.
  - 작업 후에는 파일을 닫는다.

기능	C언어 표준 라이브러리 함수	Win32 API함수
파일 열기	fopen()	CreateFile()
파일 닫기	fclose()	CloseHandle()
파일 포인터 위치 변경/획득	fseek()	SetFilePointer()
파일 읽기	fread()	ReadFile()
파일 쓰기	fwrite()	WriteFile()

파일 생성/열기 함수

- 파일 생성 함수

파일 생성 함수 CreateFile()

```
HANDLE CreateFile(
      LPCTSTR    lpFileName,
      DWORD      dwDesiredAccess,
      DWORD      dwShareMode,
      LPSECURITY_ATTRIBUTES lpSecurityAttributes,
      DWORD      dwCreationDistribution,
      DWORD      dwFlagsAndAttributes,
      HANDLE     hTemplateFile
);
```

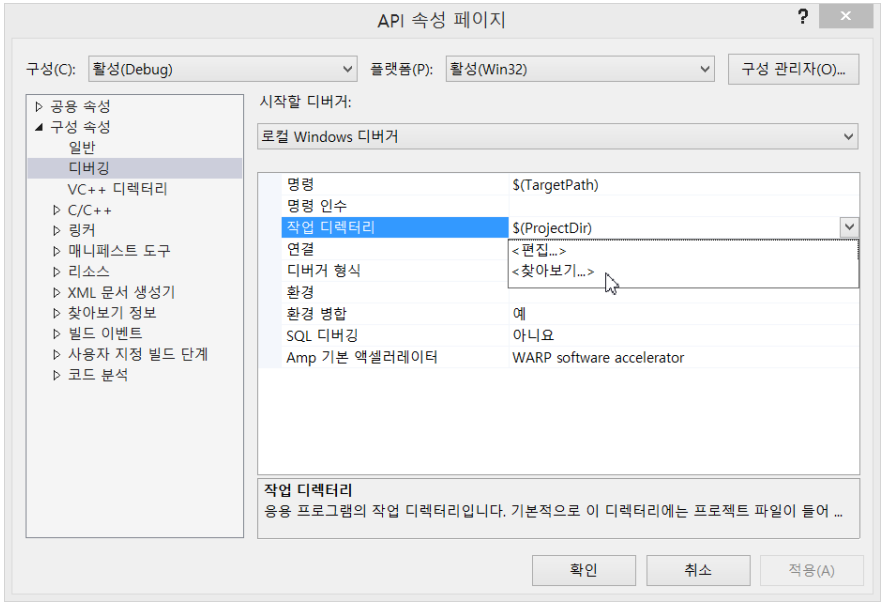
- 파일은 작업 디렉터리에 생성됨

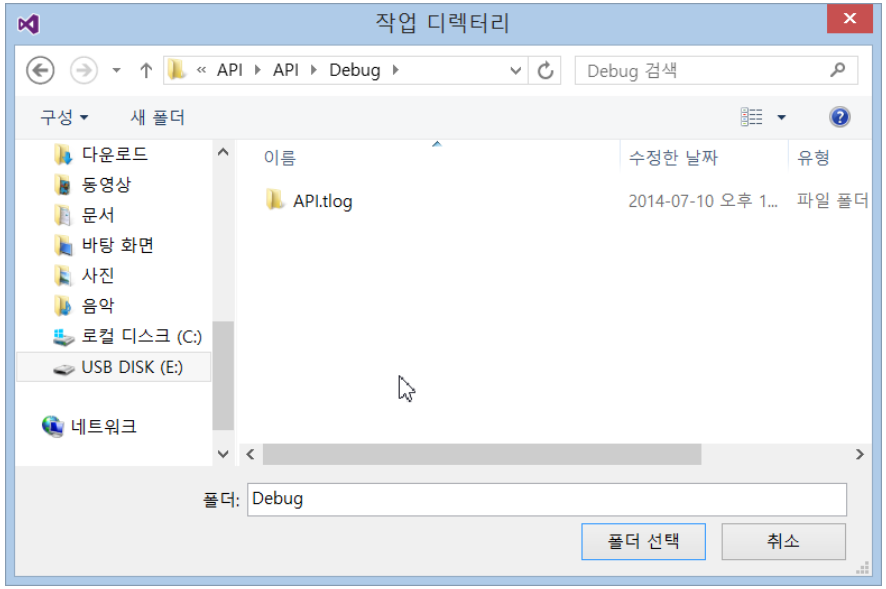
속성값

- lpFileName: 만들 파일 이름
- dwDesiredAccess: 읽기/쓰기 모드 (아래의 3 모드 중 1개 지정)
  - 읽기: GENERIC_READ
  - 쓰기: GENERIC_WRITE
  - 읽기 및 쓰기: GENERIC_READ| GENERIC_WRITE
- dwShareMode: 공유 모드 (파일 공유 여부 명시)
  - 읽기 공유허용: FILE_SHARE_READ
  - 쓰기 공유허용: FILE_SHARE_WRITE
- lpSecurityAttributes: 보안 속성 (자녀 프로세스에 상속 여부 설정), NULL 이면 상속 안됨
- dwCreationDistribution:파일 생성 모드
  - 새로 만들기, 이미 있으면 에러 메시지: CREATE_NEW
  - 항상 새로 만들기, 파일이 있어도 파괴하고 새로 만듦: CREATE_ALWAYS
  - 기존파일 열기, 파일이 없으면 에러 메시지: OPEN_EXISTING
  - 항상 열기: OPEN_ALWAYS
- dwFlagsAndAttributes: 파일 속성 (읽기 전용 파일, 시스템 파일, 숨겨진 파일 등 지정)
  - 일반적인 파일: FILE_ATTRIBUTE_NORMAL
- nFlagTemplate: 기존에 존재하는 파일과 동등한 특성을 가지는 파일을 만들기

파일 생성/열기 함수

- 작업 디렉터리 위치 확인





파일 생성/열기 함수

- 파일 읽기 함수

파일 읽기 함수 ReadFile()

```
BOOL ReadFile(
      HANDLE hFile,
      LPVOID lpBuffer,
      DWORD nNumberOfBytesToRead,
      LPDWORD lpNumberOfBytesRead,
      LPOVERLAPPED lpOverlapped
);
```

- hFile: 데이터를 읽을 파일 핸들
- lpBuffer: 읽은 자료를 넣을 버퍼
- nNumberOfBytesToRead: 읽고자 하는 바이트 크기
- lpNumberOfBytesRead: 실제 읽은 바이트
- lpOverlapped: NULL

파일 생성/열기 함수

- 파일 쓰기 함수

파일 쓰기 함수 WriteFile()

```
BOOL WriteFile(
      HANDLE    hFile,
      LPVOID    lpBuffer,
      DWORD     nNumberOfBytesToWrite,
      LPDWORD   lpNumberOfBytesWritten,
      LPOVERLAPPED lpOverlapped
);
```

  - hFile: 데이터를 저장할 파일 핸들
  - lpBuffer: 쓸 자료가 저장된 버퍼
  - nNumberOfBytesToWrite: 쓸 자료의 바이트 크기
  - lpNumberOfBytesWritten: 실제 쓴 바이트
  - lpOverlapped: NULL

- 파일 닫기 함수

파일 닫기 함수 CloseHandle()

```
CloseHandle(HANDLE hFile);
```

파일 입출력 예

```
LRESULT CALLBACK WndProc (HWND hwnd, UINT iMsg, WPARAM wParam, LPARAM lParam)
{
        HDC hdc;
        HANDLE hFile;
        TCHAR InBuff[1000];
        TCHAR OutBuff[1000] = "₩nAPI 파일 입출력 테스트입니다.";
        int size = 1000, read_size;

        switch (iMsg) {
                case WM_LBUTTONDOWN:
                hFile = CreateFile ("test.txt", GENERIC_READ|GENERIC_WRITE,
                                        FILE_SHARE_READ|FILE_SHARE_WRITE, NULL, OPEN_EXISTING, 0, 0);
                memset (InBuf, 0, 999*sizeof(char));
                ReadFile (hFile, InBuff, size, &read_size, NULL);        // hFile에서 size 만큼 읽어 InBuff에 저장
                InBuff[size] = '₩0';

                hdc = GetDC(hwnd);
                TextOut (hdc, 0, 0, InBuff, strlen(InBuff));                    // InBuff 에 있는 내용을 화면에 출력
                ReleaseDC (hwnd, hdc);

                SetFilePointer (hFile, 0, NULL, FILE_END);
                WriteFile (hFile, OutBuff, strlen(OutBuff), &size, NULL);        // OutBuff의 내용을 hFile의 끝에 저장

                CloseHandle (hFile);
        break;
```

임의 접근

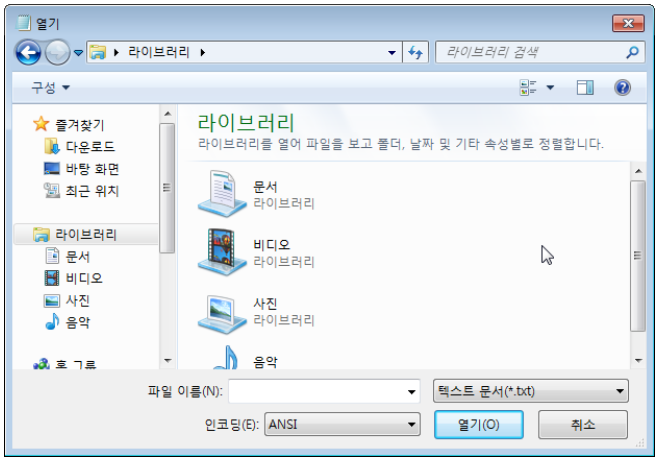
- 파일 엑세스할 때 대상 파일 위치 (File Pointer) 결정
  - 최초로 파일이 열렸을 때: FP는 파일의 선두 위치, 파일을 읽거나 쓰면 그만큼 파일 포인터가 이동 → 순차적 접근
  - 파일의 임의의 위치에서 원하는 만큼 읽는다. → 임의 접근

  - 임의 접근 함수:

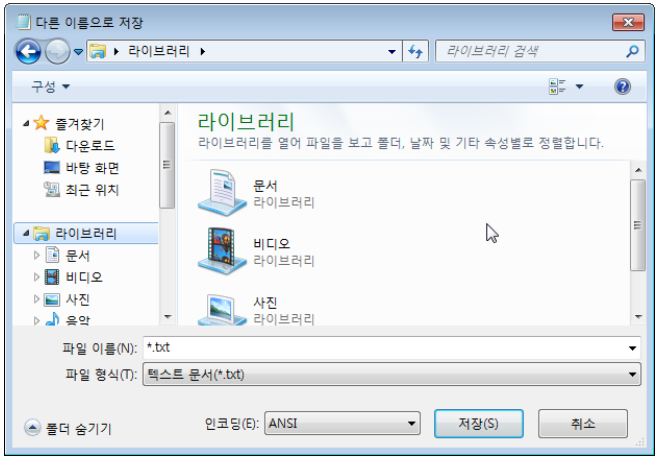
```
DWORD SetFilePointer(
    HANDLE hFile,                       // 파일 핸들
    LONG lDistanceToMove,               // 파일 포인터 이동할 위치
    PLONG lpDistanceToMoveHigh,         // 파일 크기가 2GB 이상일 경우 옮길 위치
    DWORD dwMoveMethod                  // 파일 포인터의 이동 시작 위치 지정
                                        // FILE_BEGIN / FINE_CURRENT / FILE_END
);
```

2. 공용대화상자

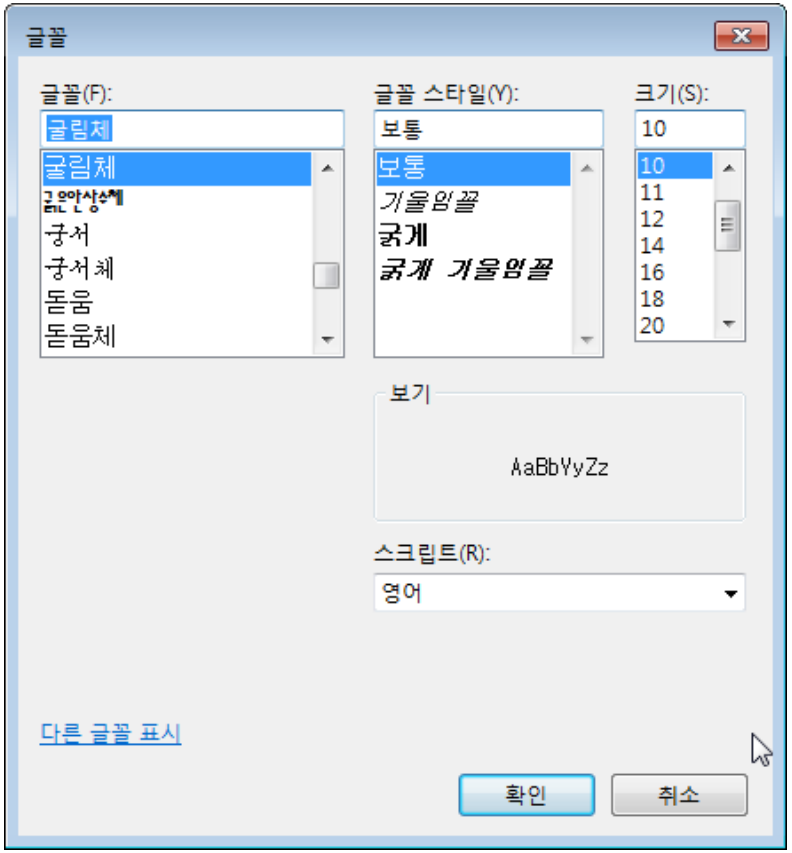
- 윈도우의 공용 대화상자



파일 열기



파일 저장하기



글꼴 선택하기

공용대화상자 – 파일 열기

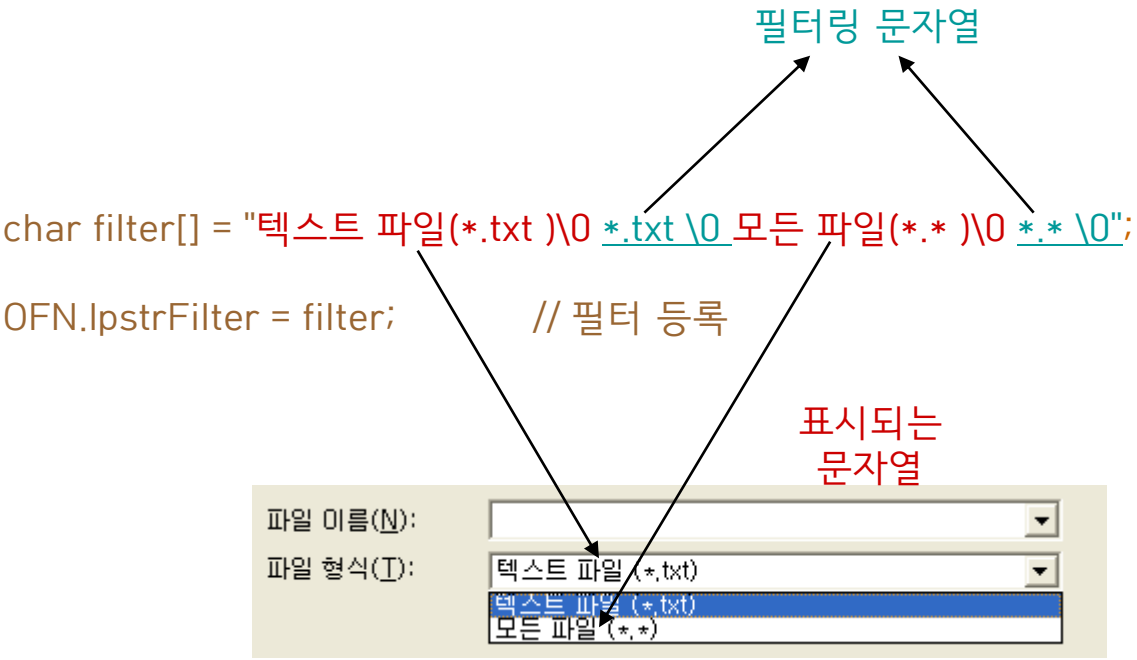
- 파일열기 처리절차
  - **OPENFILENAME** 구조체 할당
  - 열기함수 호출 -> 파일이름 획득

OPENFILENAME 구조체의 예

```
01 static char filepath[100], filename[100];
02 OPENFILENAME OFN;
03 memset(&OFN, 0, sizeof(OPENFILENAME));
04 OFN.lStructSize = sizeof(OPENFILENAME);
05 OFN.hwndOwner = hwnd;
06 OFN.lpstrFile = filepath
07 OFN.nMaxFileTitle = 100;
08 OFN.lpstrFileTitle = filename;
09 OFN.nMaxFile = 100;
10 GetOpenFileName(&OFN);
```

필터 지정 방법

- 필터의 용도
  - 표시되는 파일이름을 걸러 줌
  - 정의시 공문자 삽입 안 하도록
  - 매 필터마다 널 문자로 종료하며 하나의 필터는 **"파일형식\0필터"**로 표시한다.
  - 여러 개의 패턴 지정하려면 ;로 연결



필터 지정 방법

- OPENFILENAME 구조체

```
typedef struct tagOFN{                          // ofn
    DWORD           lStructSize;                //구조체 크기
    HWND            hwndOwner;                  //오너 윈도우 핸들
    HINSTANCE       hInstance;                  //인스턴스 핸들
    LPCTSTR         lpstrFilter;                //파일 형식 콤보 박스에 나타낼 필터
    LPTSTR          lpstrCustomFilter;          //커스텀 필터를 저장하기 위한 버퍼
    DWORD           nMaxCustFilter;             //커스텀 필터 버퍼의 길이
    DWORD           nFilterIndex;               //파일 형식 콤보 박스에서 사용할 필터의 인덱스
    LPTSTR          lpstrFile;                  //파일 이름 에디트에 처음 나타낼 파일명
                                                //최종적으로 선택된 파일이름이 저장된다.
    DWORD           nMaxFile;                   //lpstrFile 멤버의 길이
    LPTSTR          lpstrFileTitle;             //선택한 파일명을 리턴받기 위한 버퍼 (경로X)
    DWORD           nMaxFileTitle;              //lpstrFileTitle 멤버의 길이
    LPCTSTR         lpstrInitialDir;            //파일 찾기를 시작할 디렉토리
    LPCTSTR         lpstrTitle;                 //대화상자의 캡션
    DWORD           Flags;                      //대화상자의 모양과 동작을 지정하는 플래그
    WORD            nFileOffset;                //lpstrFile 버퍼 내의 파일명 오프셋
    WORD            nFileExtension;             //lpstrFile 버퍼 내의 파일 확장자 오프셋
    LPCTSTR         lpstrDefExt;                //디폴트 확장자
    DWORD           lCustData;                  //훅 프로시저로 보낼 사용자 정의 데이터
    LPOFNHOOKPROC   lpfnHook;                   //훅 프로시저명
    LPCTSTR         lpTemplateName;             //템플리트명
}OPENFILENAME;
```

파일 열기

```
OPENFILENAME OFN;
TCHAR str[100], lpstrFile[100] = "";
TCHAR filter[100] = "Every File(*.*)\0*.*\0Text File \0 *.txt;*.doc \0 ";

switch (iMsg)
{
case WM_COMMAND:
      switch(LOWORD(wParam)) {
         case ID_FILEOPEN:            // 메뉴 선택
            memset(&OFN, 0, sizeof(OPENFILENAME));   // 초기화
            OFN.lStructSize = sizeof(OPENFILENAME);
            OFN.hwndOwner = hwnd;
            OFN.lpstrFilter = filter;
            OFN.lpstrFile = lpstrFile;
            OFN.nMaxFile = 256;
            OFN.lpstrInitialDir = ".";              // 초기 디렉토리
            if (GetOpenFileName (&OFN)!=0) {
                        wsprintf (str, "%s 파일을 여시겠습니까 ?", OFN.lpstrFile);
                        MessageBox (hwnd, str, "열기 선택", MB_OK);
            }
            break;
```

파일 저장하기

```
OPENFILENAME SFN;                                    // 파일열기와 저장하기는 동일한 구조체 사용
TCHAR str[100], lpstrFile[100] = "";
TCHAR filter[100] = "Every File(*.*)\0*.*\0Text File \0 *.txt;*.doc \0 ";

switch (iMsg)
{
case WM_COMMAND:
      switch(LOWORD(wParam)) {
         case ID_FILESAVE:              // 메뉴 선택
               memset (&OFN, 0, sizeof(OPENFILENAME));  // 초기화
               SFN.lStructSize = sizeof(OPENFILENAME);
               SFN.hwndOwner = hwnd;
               SFN.lpstrFilter = filter;
               SFN.lpstrFile = lpstrFile;
               SFN.nMaxFile = 256;
               SFN.lpstrInitialDir = ".";
               if (GetSaveFileName (&SFN)!=0) {
                        wsprintf (str, "%s 파일에 저장하시겠습니까 ?", SFN.lpstrFile);
                        MessageBox (hwnd, str, "저장하기 선택", MB_OK);
               }
               break;
```

실습 8-1

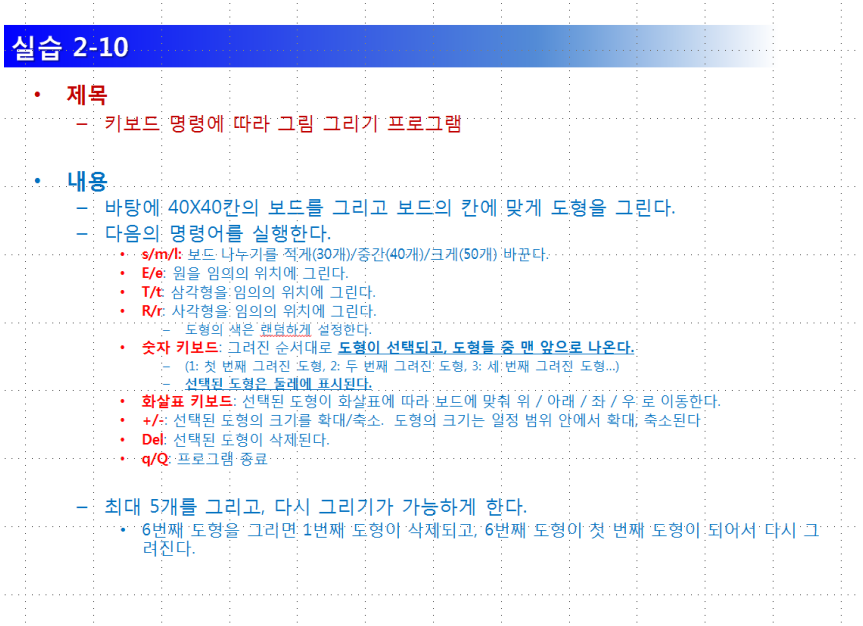
- 제목
  - 메모장에 파일 입출력 기능 추가하기

- 내용
  - 실습 2-6에서 구현한 캐럿이 있는 10라인 메모장 실습에 파일 입출력 기능을 추가한다
  - 파일 공용 대화상자를 띄워서 입출력 할 파일을 선택하도록 한다.

실습 8-2

- 제목
  - 그래픽 데이터 파일에 저장하기

- 내용
  - 실습 2-10을 사용하여 화면에 그린 도형들을 저장하고 다시 읽을 수 있는 기능을 추가하도록 한다.
    - 저장할 내용:
      - 객체의 개수
      - 객체의 종류 (원, 삼각형, 사각형)
      - 객체의 좌표 및 크기
      - 객체의 색



실습 8-3