【第十章】

檔案輸入與輸出

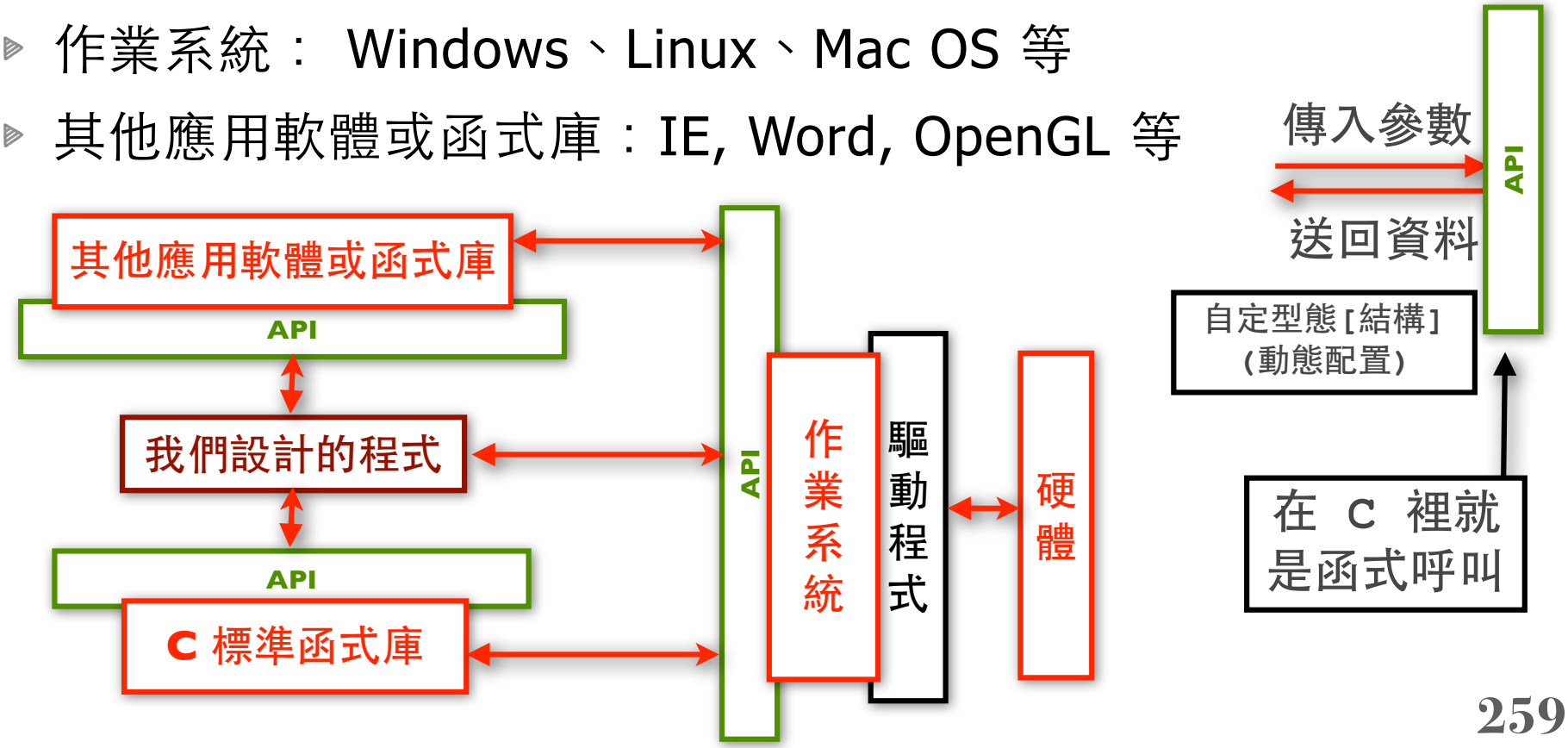
講師: 李根逸 (Ken-Yi Lee), E-mail: feis.tw@gmail.com

課程大綱

- 與作業系統或其他軟體溝通 **(API)** [P259]
- **<stdio.h>** 檔案相關函式表 [P260]
  - 開啟與關閉檔案 (**fopen, fclose**)
  - 讀寫純文字檔 (**fscanf, fprintf**)
  - 讀寫二進位檔 (**fread, fwrite**)
- 前置處理器：**#include** [P266]
- 專案：多個檔案編譯 [P268]
- 前置處理器：**#define** [P269]

與作業系統或其他軟體函式庫溝通

- 為了便利程式設計師容易設計與維護複雜的電腦軟體。一般的開發環境下我們會遇到下面幾類元件
  - 硬體：磁碟、鍵盤和滑鼠等
  - 作業系統： Windows、Linux、Mac OS 等
  - 其他應用軟體或函式庫：IE, Word, OpenGL 等



<stdio.h> 檔案相關函式表

- FILE 是個定義在 <stdio.h> 內的結構，用來描述檔案資訊

函式宣告	說明
`FILE *fopen(const char *fn, const char *m);`	用 m 模式開啟名為 fn 的檔案
`int fclose(FILE *fp);`	關閉 fp 檔案
`int fscanf(FILE *fp, const char *fmt, ...)`	依 fmt 格式從 fp 檔案讀入資料
`int fprintf(FILE * fp, const char * fmt, ...);`	依 fmt 格式對 fp 檔案寫入資料
`size_t fread(void *ptr, size_t sz, size_t cnt, FILE *fp );`	從 fp 檔案讀入每筆大小為 sz 的資料，共 cnt 筆，存放在 ptr
`size_t fwrite(const void *ptr, size_t sz, size_t cnt, FILE *fp);`	對 fp 檔案寫入 ptr 所指向每筆大小為 sz 的資料，共 cnt 筆
`int fseek(FILE *fp, long int offset, int origin);`	在 fp 檔案中，將位置移動到相對於 origin 點 offset 的位置
`int feof(FILE *fp);`	檢查 fp 是否已經讀到檔尾 (EOF)

文字檔與二進位檔

- **文字檔是將記憶體內容經由轉換成文字後用 ASCII 碼儲存，優點是對人來說可讀性高，缺點是存取較慢且可能會浪費檔案空間**
  - 常見的文字檔案有：記事本文件檔 (.txt)，網頁檔 (.html) 與 CSV 檔 (.csv)
- **二進位檔是將記憶體內容直接儲存至檔案中，優點是讀取與空間效率高，缺點是對人來說可讀性較差**
  - 常見的二進位檔有: word 檔 (.doc)，PNG 圖片檔 (.png)，JPEG 圖片檔 (.jpg) 與 BMP 圖片檔 (.bmp)
  - 絕大部分應用程式的檔案都是二進位檔！
- **範例： 怎麼在檔案儲存 12345678 這個整數？**

fopen 函式

- `FILE *fopen(const char *filename, const char *mode);`
  - `filename` 指定欲讀入的檔名字串
  - `mode` 指定開啟該檔案所使用的模式
    - "`r`" 讀取, "`w`" 寫入, "`a`" 附加
    - 寫入一個已經存在的檔案會將原檔案內容清除
    - 如果開啟檔案失敗 (例如檔名錯誤) 則 `fopen` 會傳回一個 `NULL` 值 (空指標)
      - `#define NULL 0`
    - 開啟檔案成功時 `fopen` 會傳回一個指向 `FILE` 結構的位址

《範例》寫純文字檔

- 請參考 **write_text_file.cpp**：
  - 當 **fopen** 開啟檔案失敗時會回傳 **NULL** 值 (空指標)

```
int main() {
    FILE *fp = fopen("grade.txt", "w");
     int num;
    printf("請輸入學生人數: ");
    scanf("%d", &num);
    for (int i = 1; i <= num; ++i) {
       char username[100];
       int grade;
       printf("請輸入姓名與成績: ");
       scanf("%99s", username);
       scanf("%d", &grade);
       fprintf(fp, "%s %d\n", username, grade);
    }
    fclose(fp);
    system("pause");
    return 0;
}
```

《範例》讀純文字檔

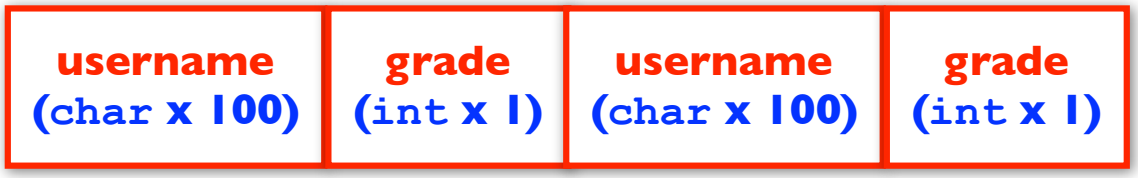
- 請參考 **read_text_file.cpp**：
  - 我們可以用 **int feof(FILE *fp)** 來檢查檔案是否已經到了結尾

```
int main() {
    FILE *fp = fopen("grade.txt", "r");
    while (1) {
        char username[100];
        int grade;
        fscanf(fp, "%s", username);
        fscanf(fp, "%d", &grade);
        if (feof(fp)) { break; }
        printf("姓名: %s (%d)\n", username, grade);
    }
    fclose(fp);
    system("pause");
    return 0;
}
```

《範例》寫二進位檔

- 請參考 **write_bin_file.cpp**

```
int main() {
  FILE *fp = fopen("bin.txt", "wb");
  int num;
  printf("請輸入學生人數: ");
  scanf("%d", &num);
  for (int i = 1; i <= num; ++i) {
    int grade;
    char username[100];
    printf("請輸入姓名與成績: ");
    scanf("%99s", username);
    scanf("%d", &grade);
    fwrite(username, sizeof(username), 1, fp);
    fwrite(&grade, sizeof(grade), 1, fp);
  }
  fclose(fp);
  system("pause");
  return 0;
}
```

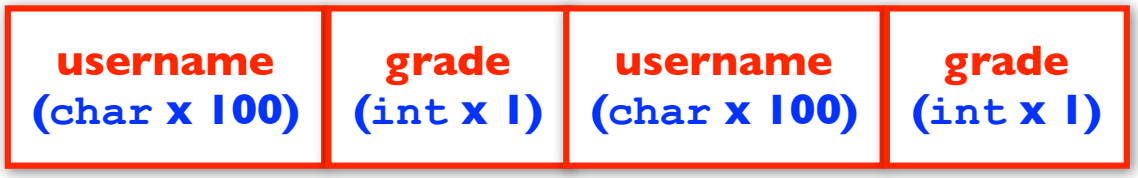


《範例》讀二進位檔

- 請參考 **read_bin_file.cpp**

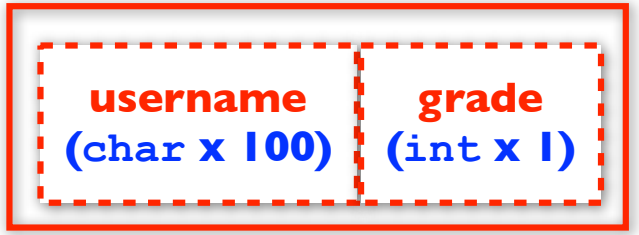
```
int main() {
  FILE *fp = fopen("bin.txt", "rb");
  while (1) {
    char username[100];
    int grade;
    fread(username, sizeof(username), 1, fp);
    fread(&grade, sizeof(grade), 1, fp);

    if (feof(fp)) { break; }
    printf("姓名: %s (%d)\n", username, grade);
  }
  fclose(fp);
  system("pause");
  return 0;
}
```



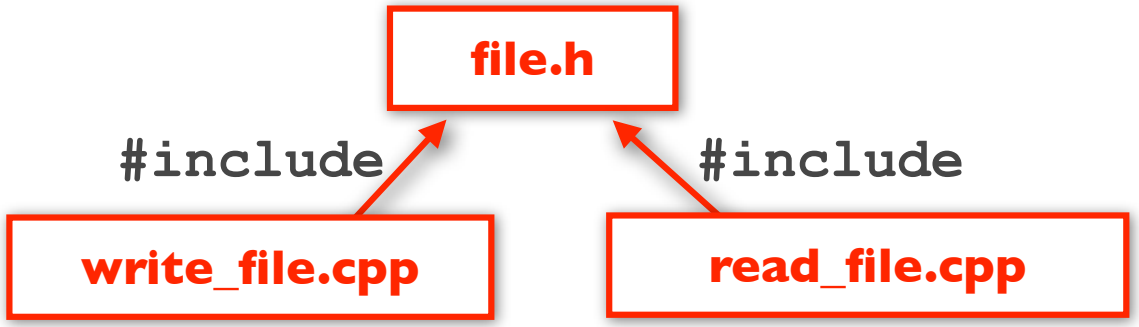
《範例》使用結構寫檔與讀檔

- 結構通常可以搭配 **fwrite** 寫檔與 **fread** 讀檔，與 **fprintf** 和 **fscanf** 不同，此時檔案通常並不是作為純文字檔在讀寫，而是類似將記憶體內容直接寫入或讀出，所以直接用文字模式觀看檔案會有些看起來近似亂碼的內容
  - 寫二進位檔請參考 **write_struct_file.cpp**
  - 讀二進位檔請參考 **read_struct_file.cpp**
- 用結構與二進位檔的優點？
  - **fseek** 可移動檔案至指定位元組
    - 隨機存取 (random access)



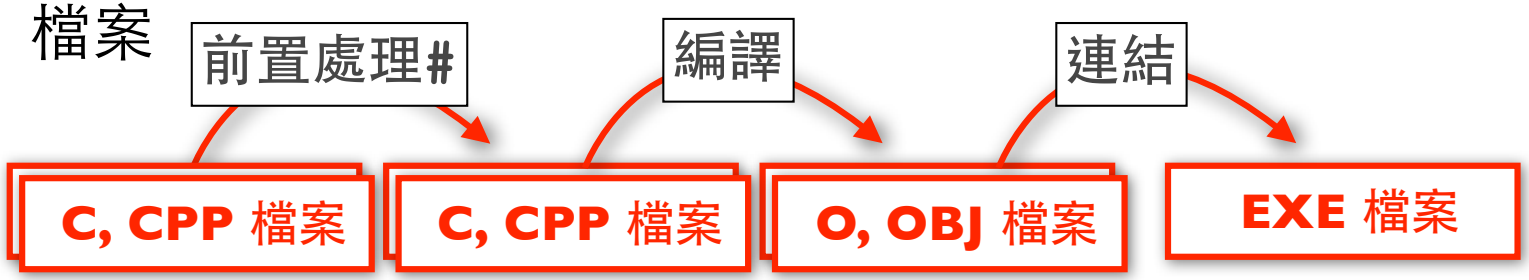
#include

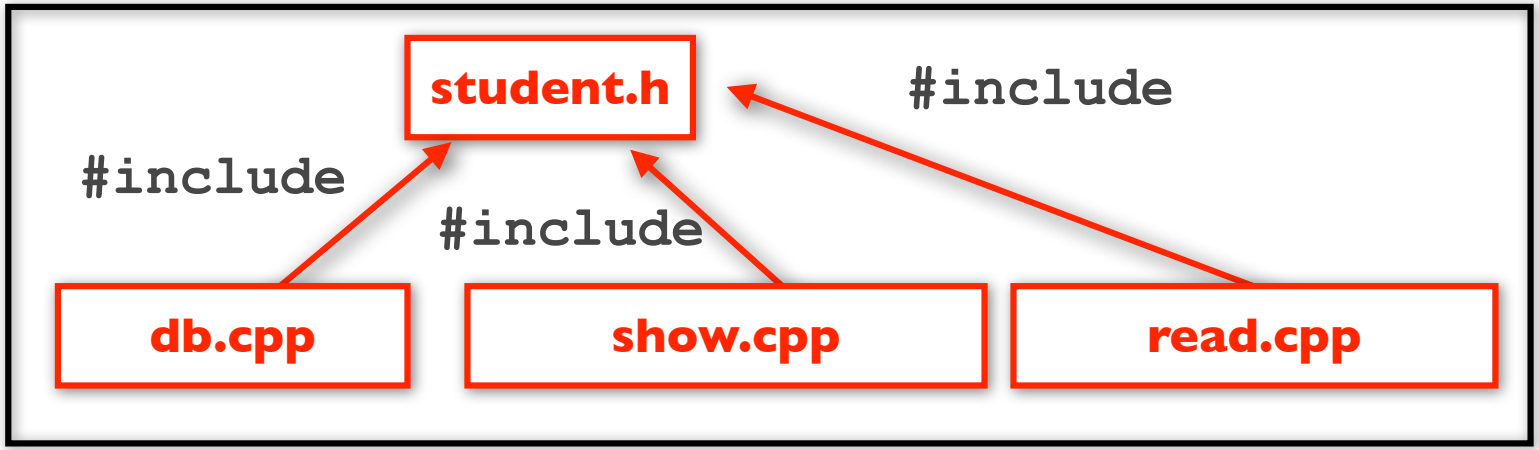
- **#include “標頭檔名稱” 可以引入標頭檔。標頭檔一般是放置函式、結構和型別宣告的地方。**
  - 這是屬於 C 前置處理器的功能
  - 使用 #include 可以讓不同的檔案使用相同的宣告
  - 概念上 #include 會將指定的標頭檔內容複製貼上於 #include 的地方
    - 標頭檔名稱用 “” 括住會優先於目前的目錄下找此檔案
    - 標頭檔名稱用<>括住會優先於系統的目錄下找此檔案



將程式分散在多個檔案

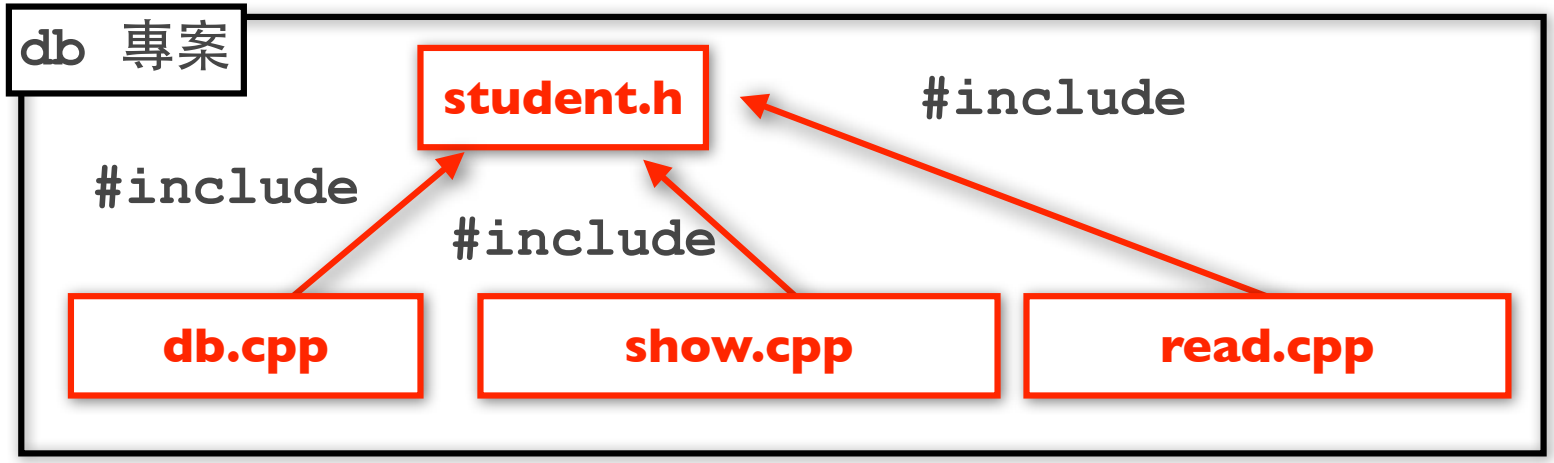
- 當我們要將函式定義放置在不同的檔案時就會面臨連結失敗 **(linking error)** 的問題
  - 試著將 db.cpp 的 show 跟 read 函式定義放置在不同檔案





專案 (Project)

- 為了在連結時知道哪些檔案是要一起考慮的，我們需要專案 **(Project)** 來幫我們
  - 一個專案內一般來說只會有一個主函式 (main)
  - 在 Dev C++ 中:
    - 使用「File > New > Project > Empty Project」新增空白專案
    - 使用「Project > Add to Project」或「Project > New File」來將程式檔案加入專案內



#define

- 前置處理器還有一個常見的語法就是 **#define**

```
#define M_PI  3.14159265358979323846
int main() {
  printf("PI = %f\n", M_PI);
  return 0;
}
```

  - 經過前置處理器後會變成：

```
int main() {
  printf("PI = %f\n", 3.14159265358979323846);
  return 0;
}
```

  - 好處是不會真的存在 M_PI 這個變數

習題 (1)

- **[E1001]** 試寫一程式 **(keyin)** 讓使用者輸入學生人數後，再輸入每個學生的座號和三個不同科目 **(Chinese, Math** 和 **English)** 的成績，並將這些資料寫入檔案 **(db2.txt)**。
  - 在這裡我們可以假設最多 60 個學生
- **[E1002]** 再寫一程式 **(query)** 自動從檔案 **(db2.txt)** 讀出資料後，讓使用者輸入座號後顯示該學生的各科成績