Chapter 13

File and Multimedia Design

13-1 File and Stream

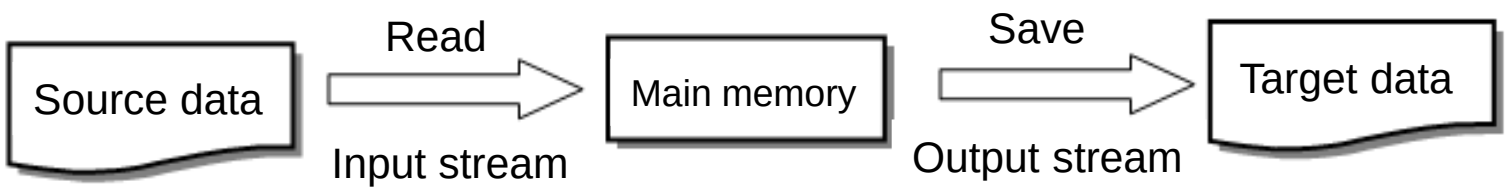
- Data process in previous chapters
  ⇨ data and source code in the same place, get input data from keyboard, output data is not saved
- Disadvantages of putting data and code together
  ⇨ have to modify data in the source code
  ⇨ key in data every time the program runs
  ⇨ execution is required to see the result
- Advantages of separating data and code
  ⇨ use program to modify data files
  ⇨ one program can process many data files

13-1 File and Stream

- C# uses stream to process data input and output
- Stream is like a tap that water flows from higher place to lower place
  ⇨ single direction, the processed character cannot be processed again
- Stream is assembled by characters or bytes
- Data process can be divided into “character stream” and “data stream”

13-1 File and Stream

- **Input stream**
  data from keyboard or file reading
- **Output stream**
  result of data process is saved into file, printed or shown on the screen



13-1 File and Stream

Types of data file:

1. Text file

- **Save character data, every byte is character**
- **Character stream**
- **Process character by character in single direction**

2. Binary file

- **Saved data is process as number**
- **Also called non-character file, Bytes data not processed**
- **Binary stream**
- **As scrambled code in text editor**

13-2 System.IO Namespace

- **When C# program uses stream, there is a code required:**
  using System.IO;
- **System.IO classes for file and directory process**

Class	Description
Directory	Providing directory of creating, deleting, moving and showing. All methods are static methods, object realization is not required
DirectoryInfo	Providing directory of creating, deleting and so on. Methods are identical to methods of Directory, but object realization is required
File	Providing file of creating, opening, duplication, deleting and moving. All methods are static, object realization is not required

Class	Description
FileInfo	Providing file creating, opening and so on. Methods are identical to methods of File, but object realization is required
FileStream	Providing file I/O process. Synchronous and asynchronous file reading and writing are supported
StreamWriter	Character stream file writer
StreamReader	Character stream file reader
BinaryWriter	Binary stream file writer
BinaryReader	Binary stream file reader

13-3 Directory Operation

- **Create, delete, move and get working directory in C#:**

  1. Directory class

  2. DirectoryInfo class

Directory Class

- All methods are static methods
- Ex: create “my” folder and its sub folder “test” in C: root directory

  Directory.CreateDirectory("c:\\my\\test\\")
- Add @ symbol before path string for ignoring \ as an escape character

  Directory.CreateDirectory(@"c:\my\test\")

DirectoryInfo Class

- All methods are NOT static methods
- The way to create “D:\my\test1” with Directory class is rewritten in DirectoryInfo class:

```
DirectoryInfo dirInfo = new  DirectoryInfo(@"D:\my\test1\");
dirinfo.Create();
```

Directory Static Methods

1. **CreateDirectory(path)**
   create the folder of designated path. The parent folders are also created if they do not exist

Ex: create directory “test” under “D:\my”, there are 2 usages

```
Directory.CreateDirectory(@"D:\my\test\");
Directory.CreateDirectory("D:\\my\test\\");   ⇦ 省略@
```

Directory Static Methods

2. Exists(path)
examine whether the directory of path is in existence or not. Return true for existence and return false for not

Ex: examine "D:\my" is in existence or not. Create D:\my if the directory does not exist

```
string path = @"D:\my\";
if (Directory.Exists(path) == false)        // if path does not exist
    Directory.CreateDirectory(path);        // create path
```

if (Directory.Exists(path) == false) can be rewritten in
if (!Directory.Exists(path))

Directory Static Methods

3. Delete (path, recursive)

- Delete the folder designated path
- If recursive = true, the sub folders and files are also deleted
- If recursive = false and there are folders or files in the directory, deleting is cancelled

Ex: delete all sub folders and files in D:\my directory

```
Directory.Delete(@"D:\my\", true);
```

Directory Static Methods

4. Move(sourceDir, destDir)

- Move all files and folders in sourceDir to destDir
- Ex: move folders and files in D:\my to D:\you

```
Directory.Move(@"D:\my", @"D:\you");
```

Directory Static Methods

5. **GetDirectories(path)**
   get list of child directories of the path's designated folder, return string array

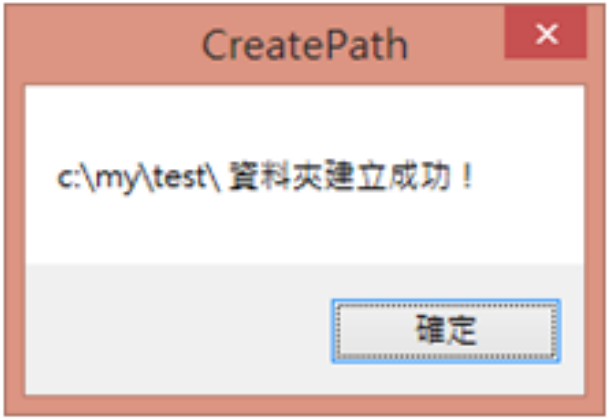
6. **GetFiles(path)**
   get list of child files of the path's designated folder, return string array

Practice(CreatePath):

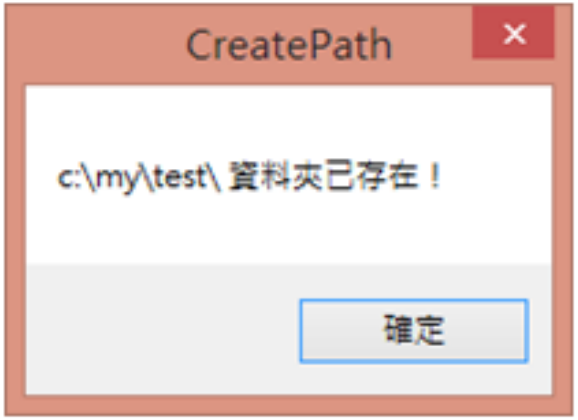
Design a program to create folders. Requirements:
Create C:\my\test folder in C: disc

1. If the folder exists, show message “C:\my\test資料夾已存在”
2. If the folder does not exist, create the folder and show message “C:\my\test資料夾建立成功”

Result:



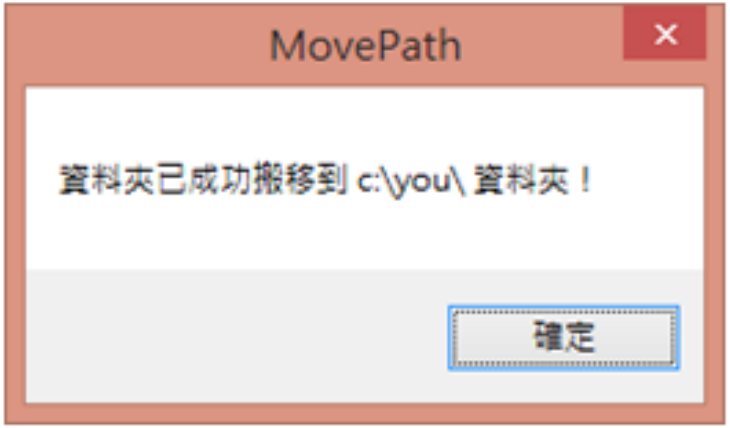
或



Practice(MovePath):

Design a program to move folders. Requirements:
Move folder “C:\my”, its sub folders and files to “C:\you”

Result:



或

Directory moving successfully

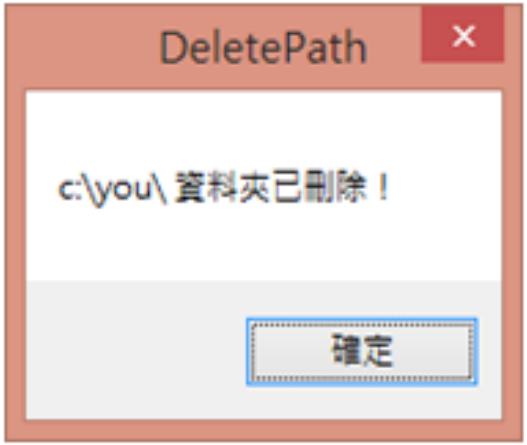
Directory don’t exist

Practice(DeletePath):

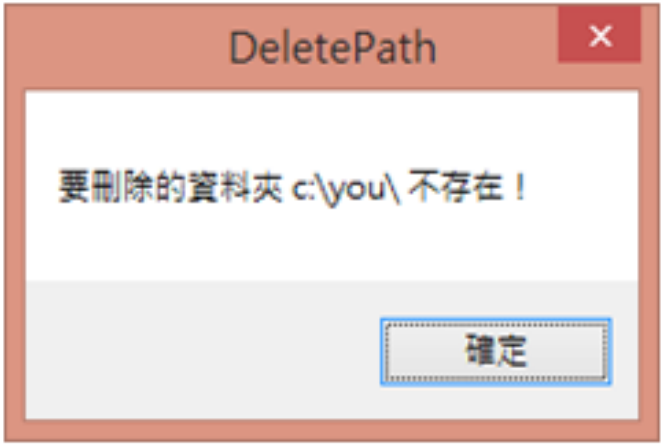
From previous example, design a program to delete folders. Requirements:
Delete C:\you folder, its sub folders and files

1. If the folder is deleted successfully, show the left figure
2. If the folder you are going to delete does not exist, show the right figure

Result:



或



13-4 File Operation

- C# uses File class and FileInfo class to access file data
- File class
  all methods are static methods
  ⇨ use File.methodName to call the method
- FileInfo class
  realization is required

FileInfo Constructor

1. FileInfo(String path)

- Create FileInfo object
- Parameter path is path of directory or file
- Ex: create a FileInfo object called finfo and open “D:\my\htc.txt”

```
FileInfo finfo = new FileInfo(@“d:\my\htc.txt”);
      or
FileInfo finfo = new FileInfo(“d:\\my\\htc.txt”);
```

FileInfo Properties

Property	Description
Exists	Acquire whether the designated file is in existence or not true: exist, false: do not exist. Ex:finfo.Exists
FullName	Get full path of the file, the path includes directories ex: finfo.FullName
Name	Get the file name with the extension Ex: finfo.Name
Extension	Get the extension of the file Ex: finfo.Extension
DirectoryName	Get the path of the directory which the file belongs to. Ex: finfo.DirectoryName
Length	Get the size of file Ex: finfo.Length

FileInfo Methods

1. Create()

- To create and open the file assigned by FileInfo object
- The path has to exist or the error occurs
- Ex: finfo is a FileInfo object and finfo stands for "D:\my\htc.txt"

```
FileInfo finfo = new FileInfo("d:\\my\\htc.txt");
FileStream fs = finfo.Create();          // create htc.txt
```

FileInfo Methods

2. Close()

- To close the file referred by the opened FileInfo object
- File stream in the same function has to be closed if the stream is no longer to be used
  ⇨ or repeated realization error occurs
- Ex: close the file referred by FileInfo object fs
  fs.Close();

FileInfo Methods

3. CopyTo (String path, Boolean)

- Copy the current file to the file assigned by path, the source file still exists
- Copy to bin\Debug of project directory if the path is not defined
- The path of the target directory has to be created in advance, or the error occurs when copying
- If the second parameter is true, overwrite the file if it is already in existence, and false for not overwriting

FileInfo Methods

Ex: copy the file “D:\my\htc.txt” assigned by finfo to “D:\you\newhtc.txt”

```
FileInfo finfo = new FileInfo(“d:\\my\\htc.txt”);
finfo.CopyTo(“d:\\you\\newhtc.txt”, true);
```

FileInfo Methods

4. MoveTo(String path)

- Move the current file to the file assigned by path, the source file is removed
- Moving does not have overwriting function, the target directory has to be created and the target file cannot exist before moving

Ex: move the file “D:\my\htc.txt” assigned by finfo to “D:\you\newhtc.txt”

```
FileInfo finfo = new FileInfo(“d:\\my\\htc.txt”);
finfo.MoveTo(“d:\\you\\newhtc.txt”);
```

FileInfo Methods

5. Delete()

- **Delete the designated file referred by FileInfo object**
- **Ex: delete the file “D:\you\newhtc.txt” assigned by finfo object**

```
FileInfo finfo = new FileInfo(“d:\\you\\newhtc.txt”);
finfo.Delete();
```

FileInfo Methods

6. CreateText()

- Create and open a new text file assigned by FileInfo object
- The new file has no data, StreamWriter is required for writing data
- If the file exists, clear the content. If the file does not exist, create a new file
- The designated path of the directory has to be created in advance or the errors would occur

FileInfo Methods

- **Ex: create a new file called “htc.txt” in D:\my directory and assign StreamWriter “sw” as output data stream to write output data into the file**

```
FileInfo finfo = new FileInfo(“d:\\my\\htc.txt”);
StreamWriter sw = finfo.CreateText();
```

FileInfo Methods

7. AppendText()

- Use StreamWriter to append the data to the end of the text file
- If the file exists ⇨ open the file
- If the file does not exist ⇨ create a new file
- The directory has to be created in advance, or the error occurs

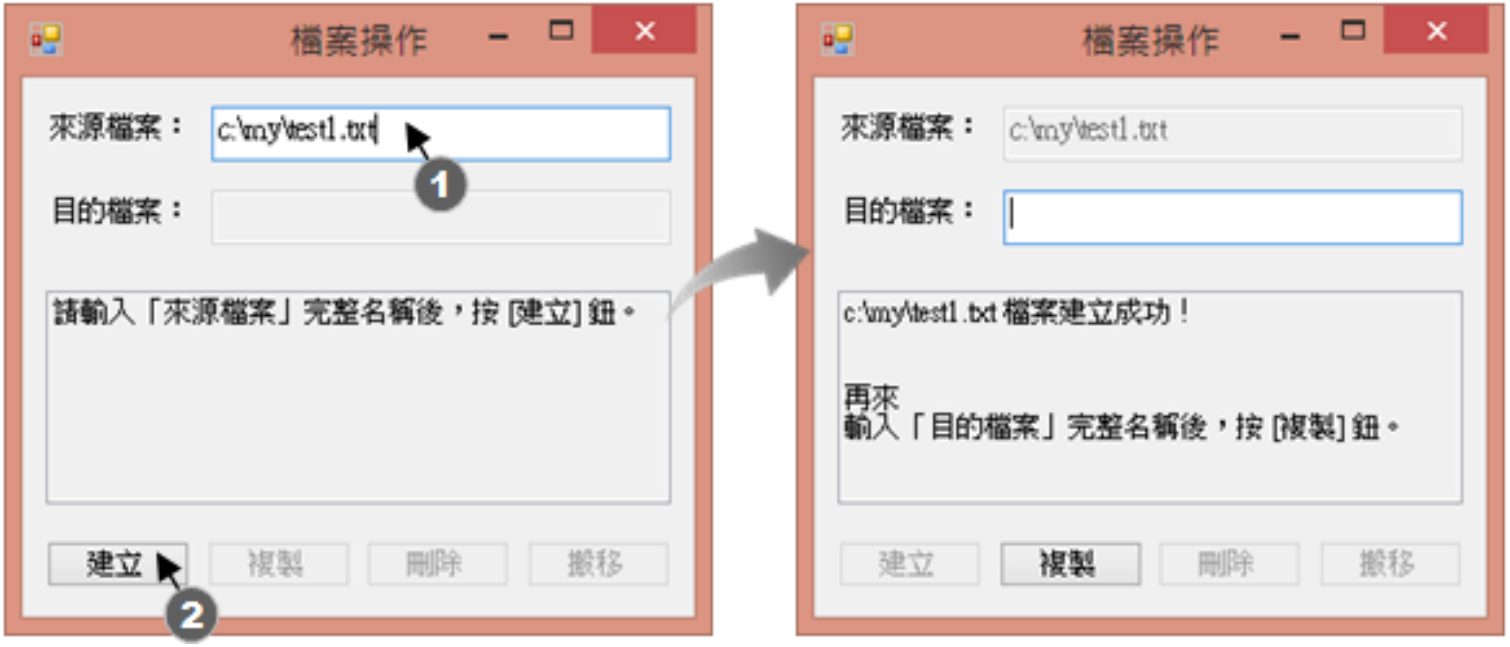
Ex: open the text file "D:\my\htc.txt" and assign StreamWriter sw as output data stream. Write output data at the end of file with output data stream

```
FileInfo finfo = new FileInfo("d:\\my\\htc.txt");
StreamWriter sw = finfo.AppendText();
```

Practice(FileOperate):

Design a program which can create, copy, delete and move data file.
Requirements:

1. The program starts as shown in figure 1. “建立” button is available but others are not
2. Input “C:\my\test1.txt” in source file, and press “建立” button to show message “C:\my\test1.txt檔案建立成功!” as shown in figure 2. In the meantime, “複製” button is available and others are not




檔案操作
來源檔案：
c:\my\test1.txt
目的檔案：
c:\you\test2.doc
3
c:\my\test1.txt 檔案建立成功！
再來
輸入「目的檔案」完整名稱後，按 [複製] 鈕。
建立
複製
刪除
搬移
4

圖 3

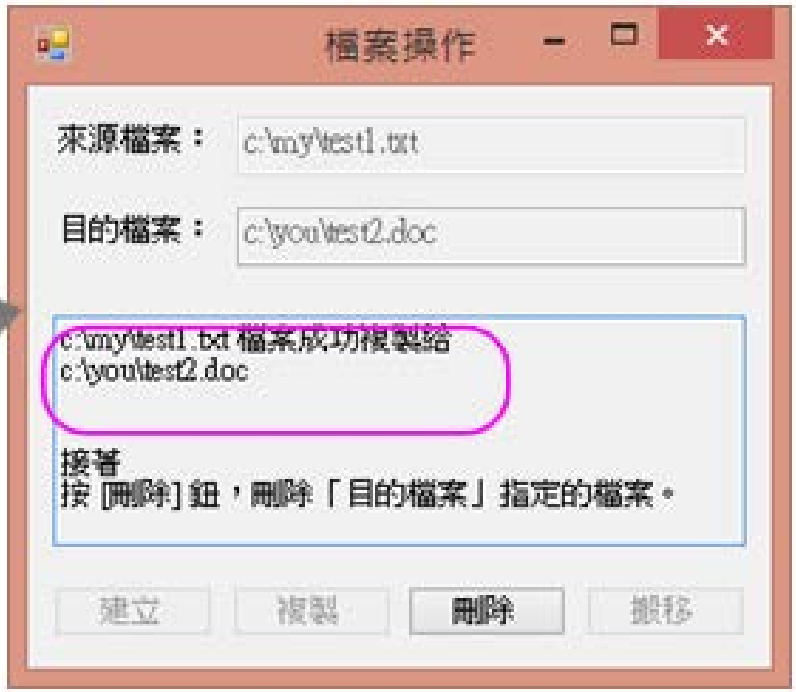

檔案操作
來源檔案：
c:\my\test1.txt
目的檔案：
c:\you\test2.doc
c:\my\test1.txt 檔案成功複製給
c:\you\test2.doc
接著
按 [刪除] 鈕，刪除「目的檔案」指定的檔案。
建立
複製
刪除
搬移

圖 4

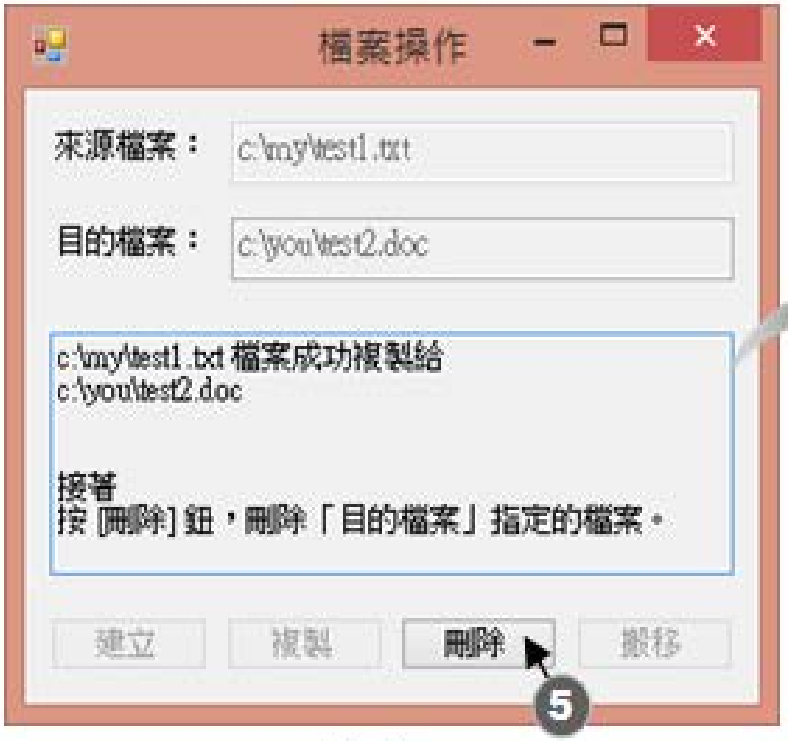

檔案操作
來源檔案：
c:\my\test1.txt
目的檔案：
c:\you\test2.doc
c:\my\test1.txt 檔案成功複製給
c:\you\test2.doc
接著
按 [刪除] 鈕，刪除「目的檔案」指定的檔案。
建立
複製
刪除
搬移
5

圖 5

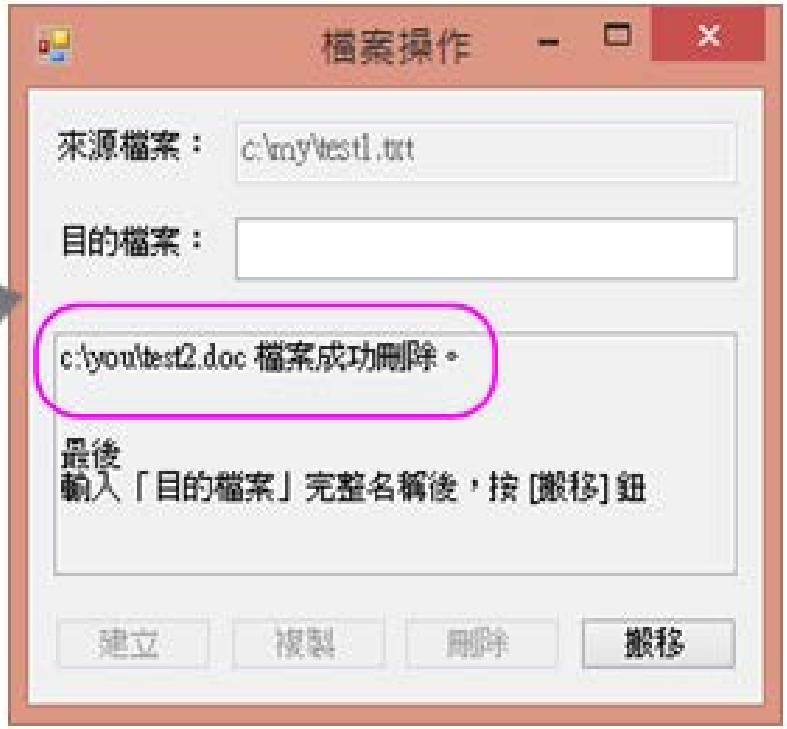

檔案操作
來源檔案：
c:\my\test1.txt
目的檔案：
c:\you\test2.doc 檔案成功刪除。
最後
輸入「目的檔案」完整名稱後，按 [搬移] 鈕
建立
複製
刪除
搬移

圖 6

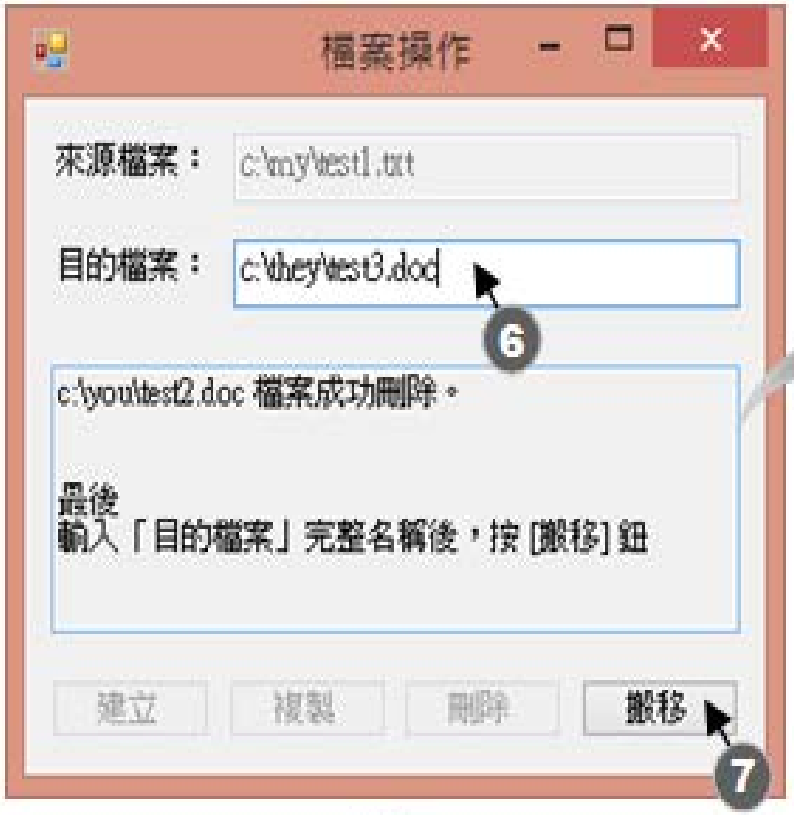

檔案操作
來源檔案：
c:\my\test1.txt
目的檔案：
c:\they\test3.doc
6
c:\you\test2.doc 檔案成功刪除。
最後
輸入「目的檔案」完整名稱後，按 [搬移] 鈕
建立
複製
刪除
搬移
7

圖 7

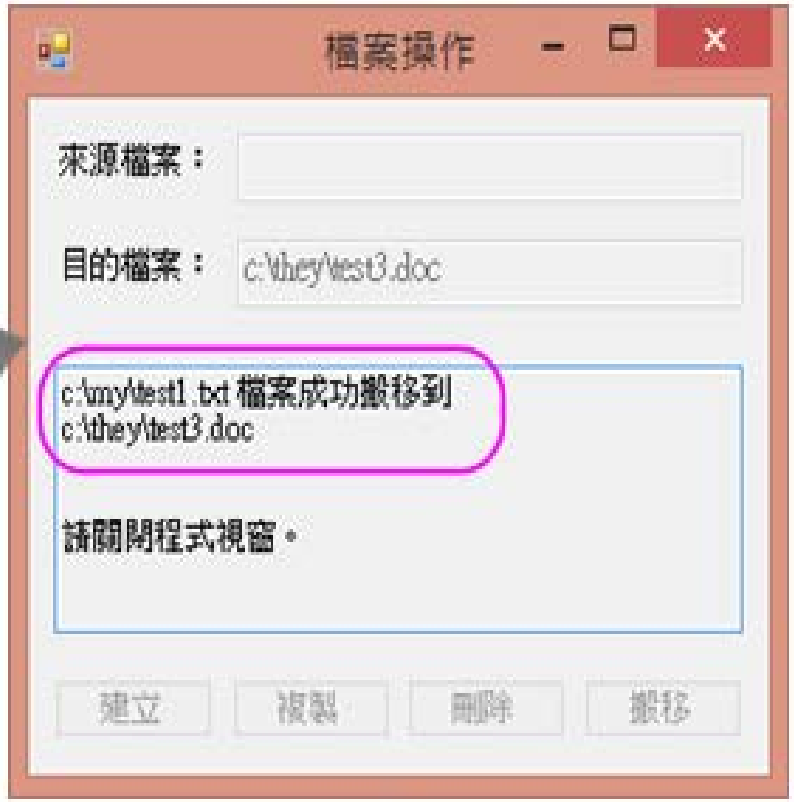
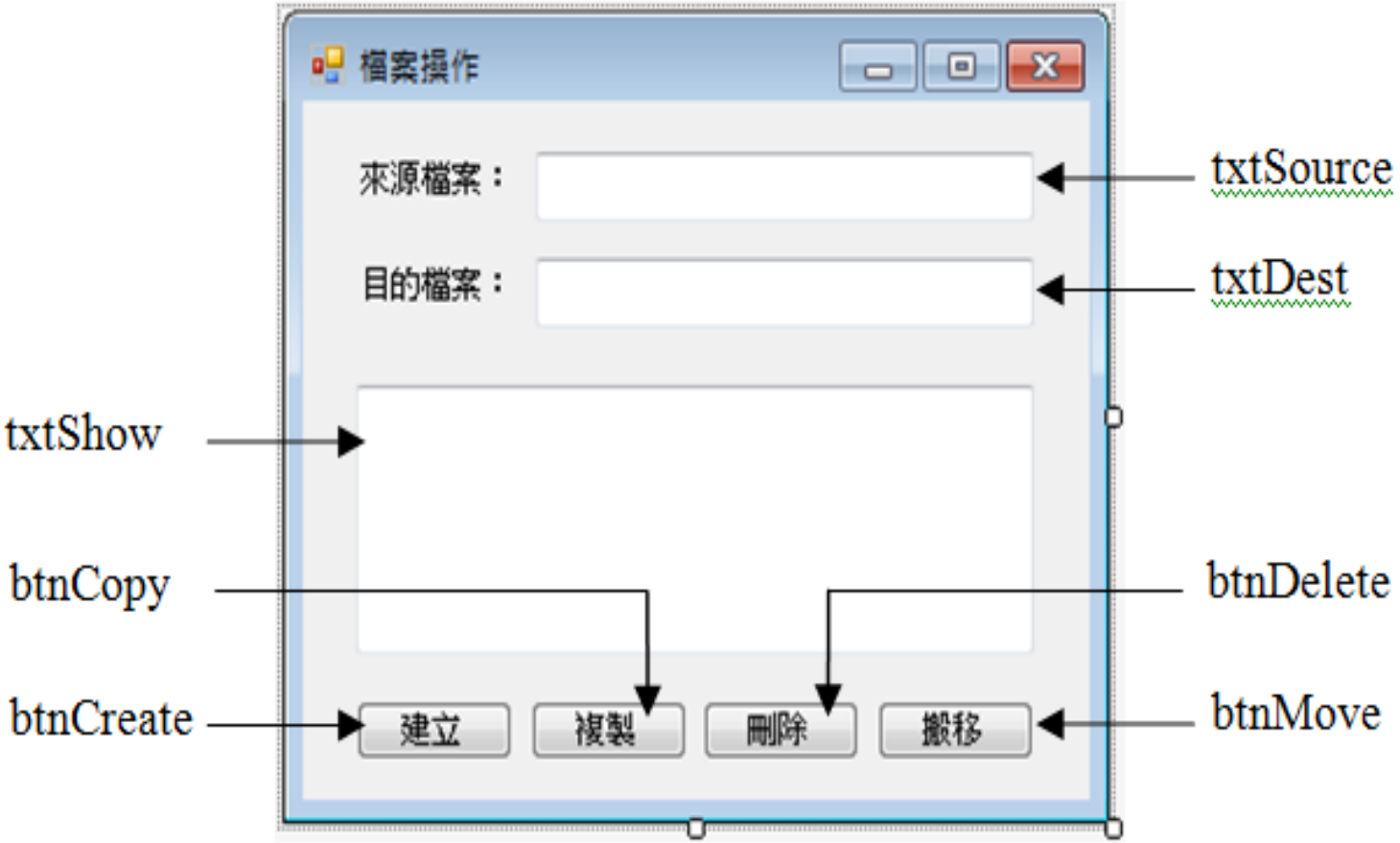

檔案操作
來源檔案：
目的檔案：
c:\they\test3.doc
c:\my\test1.txt 檔案成功搬移到
c:\they\test3.doc
請關閉程式視窗。
建立
複製
刪除
搬移

圖 8

Design User Interface



13-5 Text Data Read & Write

- **StreamWriter Class**
  ⇨ process text data writing
  StreamReader Class
  ⇨ process text data reading
- **StreamWriter Methods:**

Method	Description
Write(String)	Write string data into the file and attach to the end of file
WriteLine(String)	Write string data and new line character into the file
Flush()	Clear buffer data
Close()	Close data stream object

Ex: use the methods provided by FileInfo and StreamWriter to write data. The data is written into D:\my\apple.txt through StreamWriter sw

Steps to Write Text File

Step 1 create FileInfo object

```
FileInfo finfo = new FileInfo(@“d:\my\apple.txt”);
FileInfo finfo = new FileInfo(“d:\\my\\apple.txt”);
```

Step 2 choose the way to open data file

1. use CreateText()

```
StreamWriter sw = finfo.CreateText();
```

2. use AppendText()

```
StreamWriter sw = finfo.AppendText();
```

Step 3 write data

1. sw.Write(string)

write the string and attach to the end of file

2. sw.Writline(string)

write the string with new line character

Step 4 put data of output stream into the file and erase the buffer

```
sw.Flush();
```

Step 5 Close the file

```
sw.Close();
```

Ex1: put input data of textBox1 into the string array called product line by line

```
int k = 0;
foreach (string item in textBox1.Lines )
{
        product[k] = item ;
        k++ ;
}
```

Ex2: show every element of product string array on label1. The AutoSize property of label1 should set to false before execution, so can enlarge manually for multi-line display

```
foreach (string item in product )
        lblShow.Text += item + "\n";
```

Ex3: create FileInfo object called finfo, the object opens D:\my\apple.txt text file. The file is created if the file does not exist

```
string filename = @"D:\my\apple.txt";
FileInfo finfo = new FileInfo(filename);
if (!Directory.Exists(finfo.DirectoryName))
        Directory.CreateDirectory(finfo.DirectoryName);
```

Ex4: create and open the text file referred by finfo object. The data output stream sw is also created. To output data, use the data output stream to write data into the file referred by finfo object

```
StreamWriter sw = finfo.CreateText();
```

Ex5: write every element of product string array into the file assigned by sw stream object

```
foreach (string item in product)
        sw.WriteLine(item);
```

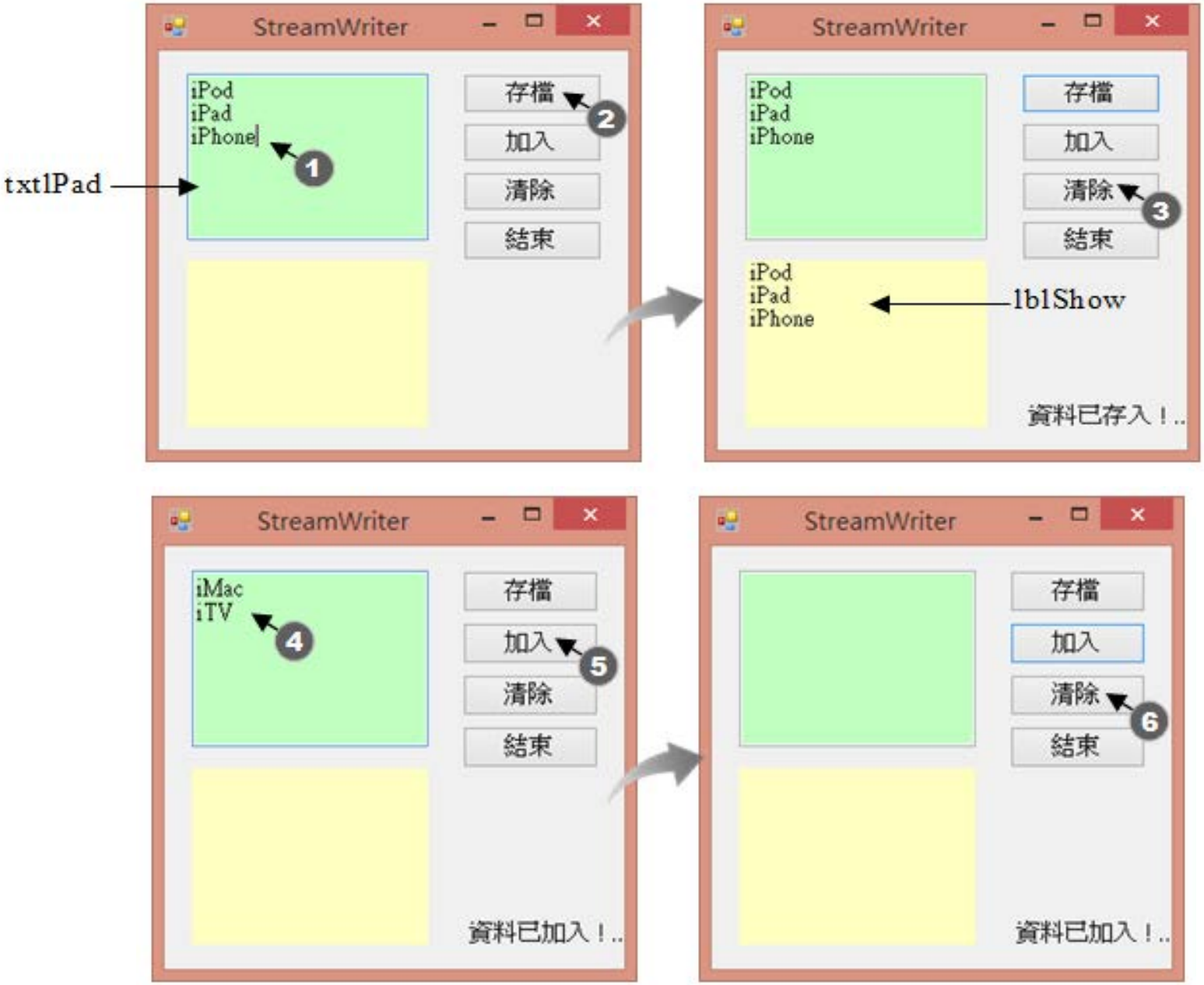
Ex6: write all data of textBox1 into the file assigned by sw stream object

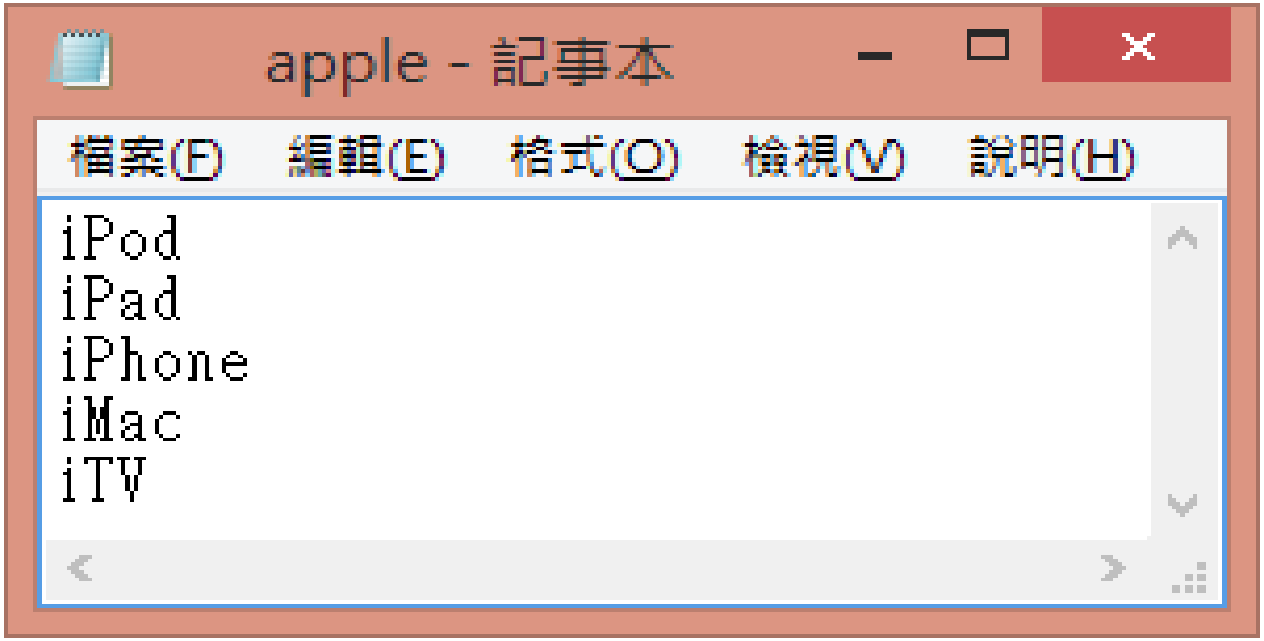
```
sw.WriteLine(textBox1.Text) ;
```

Practice(StreamWriter1):

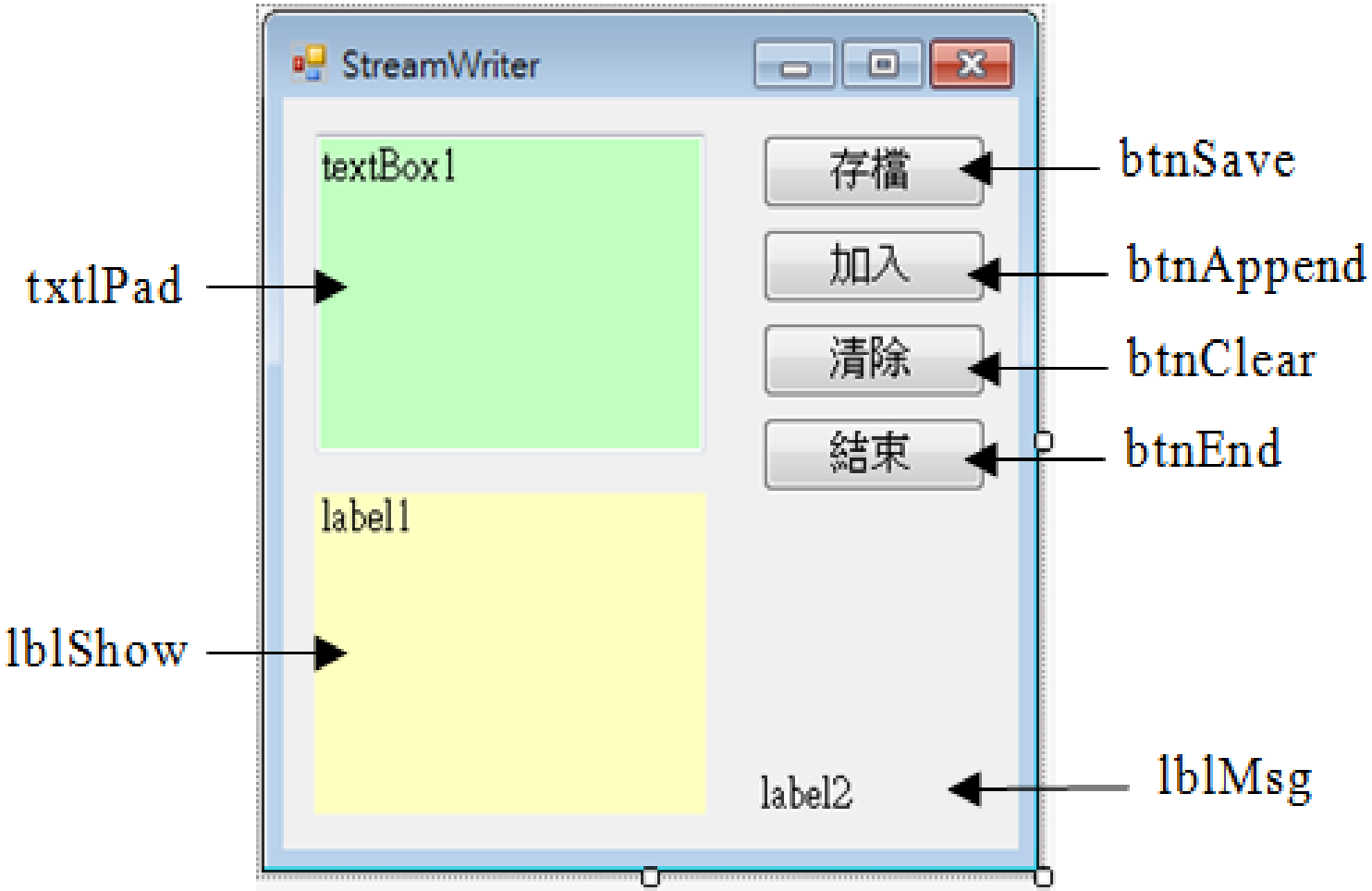
Design a program to write text inputted by keyboard into the text file.
Requirements:

1. Input 3-line text into txtIPad text box with keyboard. Use enter to change to new line.
2. Press “存檔” button to verify and save 3-line text into d:\my\apple.txt, then show the message in lblShow label


StreamWriter
iPod
iPad
iPhone
txtIPad
存檔
加入
清除
結束
1
2
StreamWriter
iPod
iPad
iPhone
存檔
加入
清除
結束
3
iPod
iPad
iPhone
lblShow
資料已存入！..
StreamWriter
iMac
iTV
存檔
加入
清除
結束
4
5
資料已加入！..
StreamWriter
存檔
加入
清除
結束
6
資料已加入！..


apple - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)
iPod
iPad
iPhone
iMac
iTV

Design User Interface



Read Character Stream Data from Text File

- Use StreamReader class to read character stream data from the text file
- When reading:
  ① read a character at a time
  ② read a line of string at a time
  ③ read whole text

StreamReader Methods

Method	Description
Read()	Read a character or a Chinese word from text file
ReadLine()	Read a line of string without new line character from text file. Return null if read to the end of data stream
ReadToEnd()	Read from the beginning of file to the end of file
Peek()	Examine the next character for reading, return -1 if it is the end of the file

Steps to Read Text File

Step1 **create stream reader**

StreamReader sr = new StreamReader(path);

Step2 **read data**

1. read a character every time

Ex: in the loop, use Read() to get a character every time, and use Peek() to check the document is fully read or not. Peek() returns -1 when reading is finished and exit the loop

```
do
{
    ch = (char)sr.Read();
    if (sr.Peek() == -1)
        break;
    txtShow.Text += ch;
} while(true);
```

Use MessageBox to check the read character

```
DialogResult result;
result = MessageBox.Show("按<是>鈕繼續輸出 按<否>鈕離開, "是否繼續",
            MessageBoxButtons.YesNo, MessageBoxIcon.Exclamation);
if (result == DialogResult.No) // 按 <否>鈕
{
        txtShow.Text = "";
        return;
}
```

2. read a line of string every time

Ex: in the loop, use ReadLine() to get a string which has no new line character every time, add “\r\n” manually each time when displayed. If already read to the end of data stream, return null to check whether reading has ended.

```
do
{
        data = sr.ReadLine();
        if (data == null) break;
        txtShow.Text += data + "\r\n";
} while(true);
```

3. read one character at a time

use ReadToEnd() to read text from the current position of the cursor to the end of file

```
txtShow.Text = sr.ReadToEnd();
```

Step3 close data stream

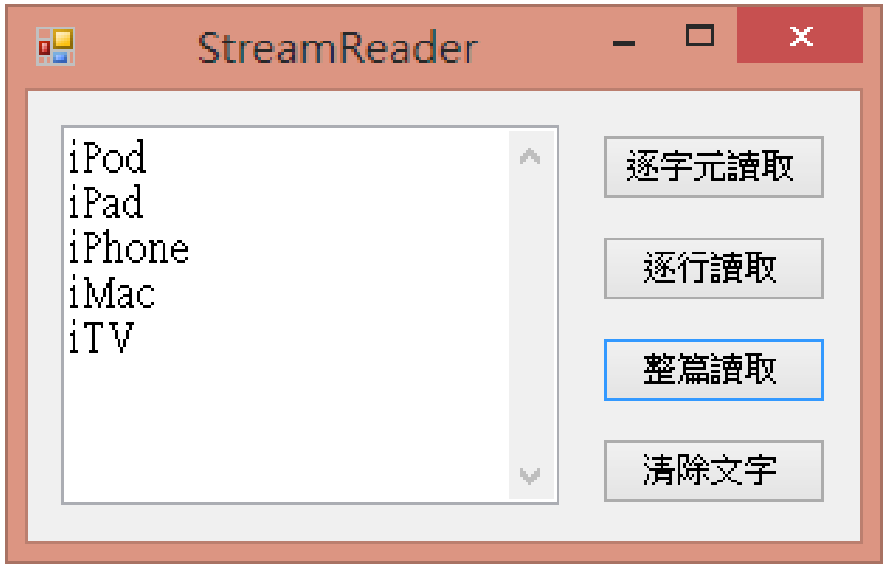
sr.Close();

```
sr.Close();
```

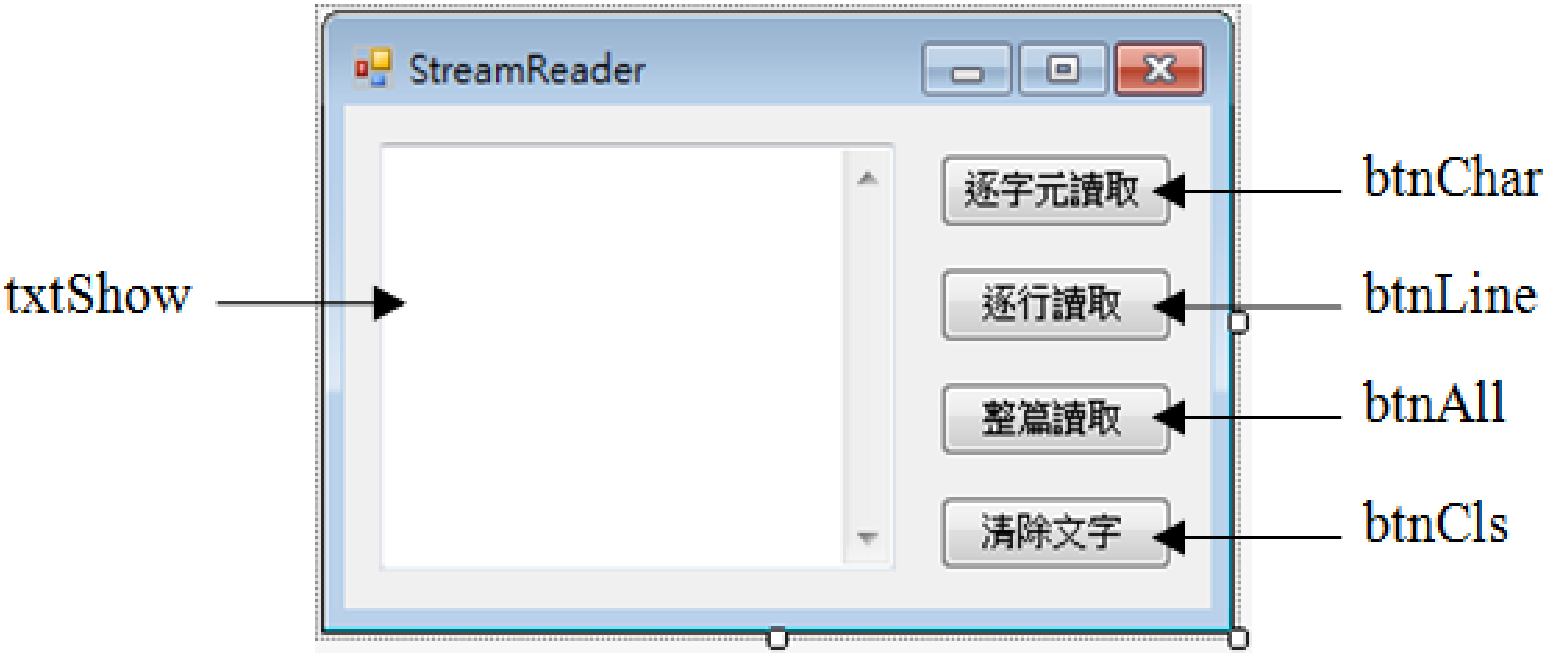
Practice(StreamReader1):

Design a program to load text file. The functions include “逐字元讀取”, “逐行讀取” and “整篇文章讀取”. The example file is D:\my\apple.txt. The program reads the characters by the chosen function.


StreamReader
iPod
iPod
逐字元讀取
逐行讀取
整篇讀取
清除文字
是否繼續
按 <是> 鈕 繼續輸出
按 <否> 鈕 離開
是(Y)
否(N)

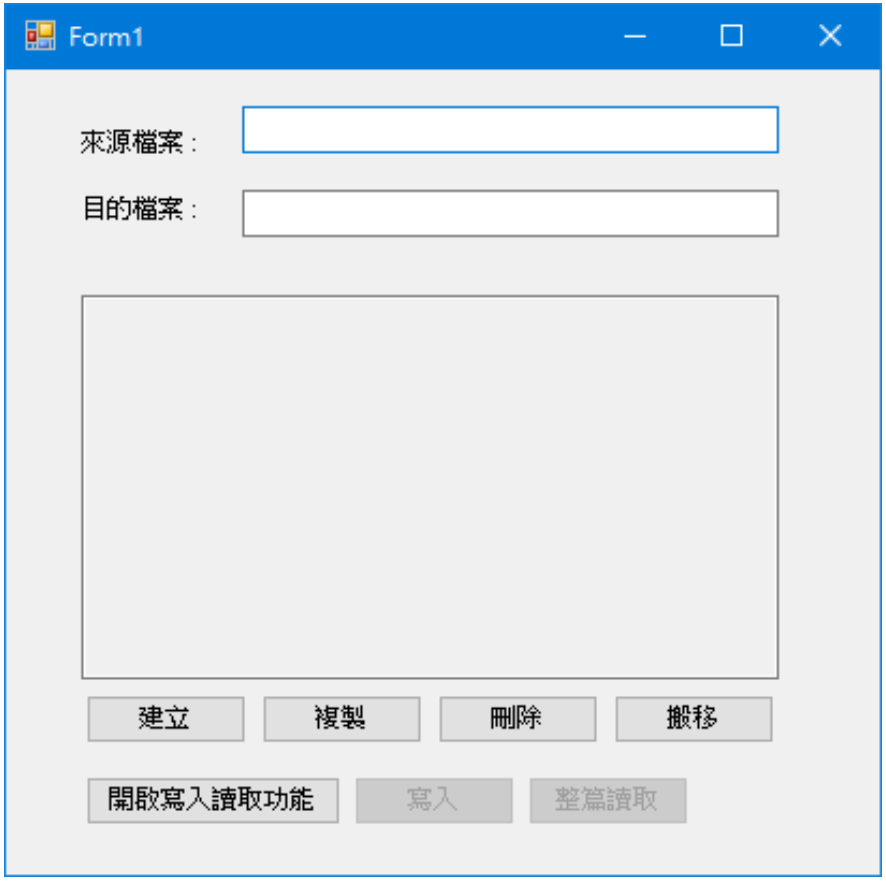

StreamReader
iPod
iPad
iPhone
iMac
iTV
逐字元讀取
逐行讀取
整篇讀取
清除文字

Design User Interface

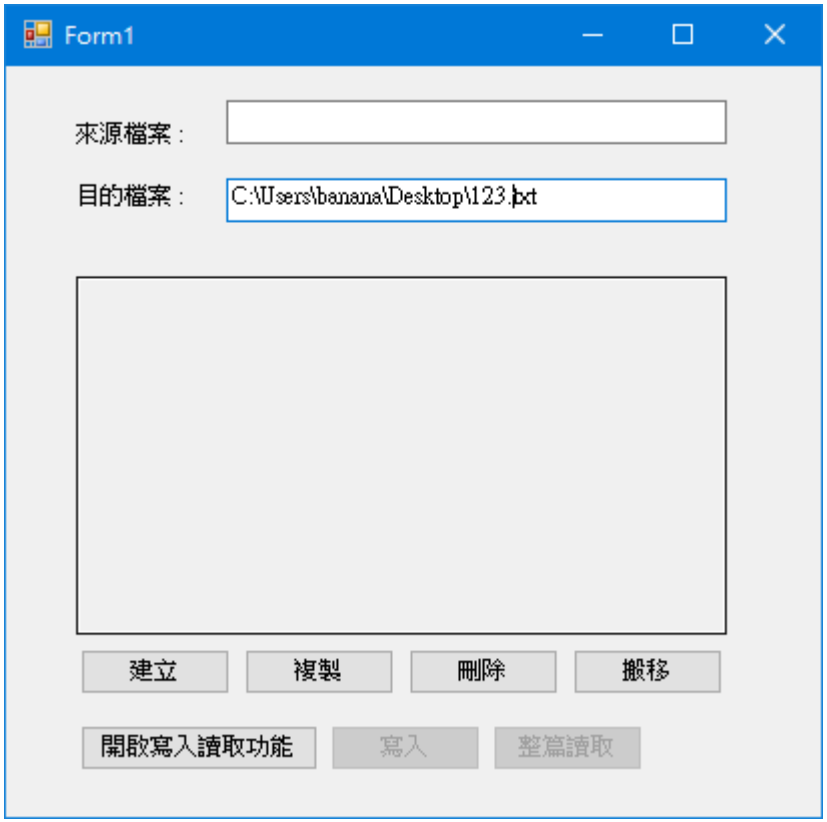


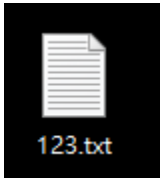
Practice(load):

Design a program to load text file. The functions include “建立”, “複製”, “刪除”, “搬移”, “寫入” and “整篇讀取”.

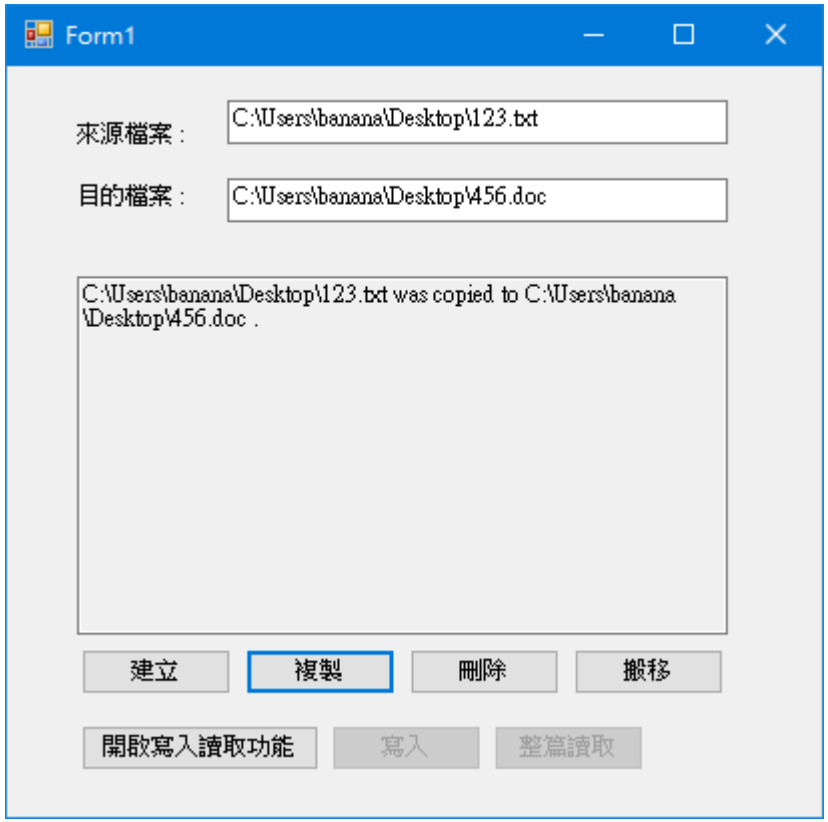


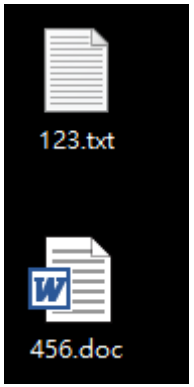
Build


Form1
來源檔案：
目的檔案：
C:\Users\banana\Desktop\123.txt
建立
複製
刪除
搬移
開啟寫入讀取功能
寫入
整篇讀取

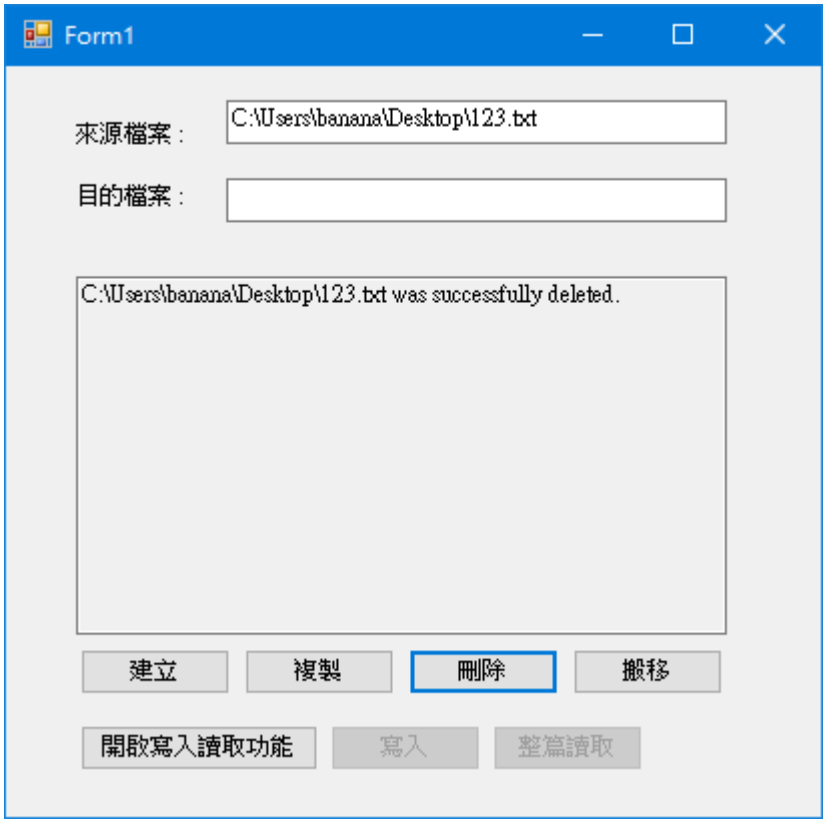

123.txt

Copy


Form1
來源檔案：
C:\Users\banana\Desktop\123.txt
目的檔案：
C:\Users\banana\Desktop\456.doc
C:\Users\banana\Desktop\123.txt was copied to C:\Users\banana\Desktop\456.doc .
建立
複製
刪除
搬移
開啟寫入讀取功能
寫入
整篇讀取

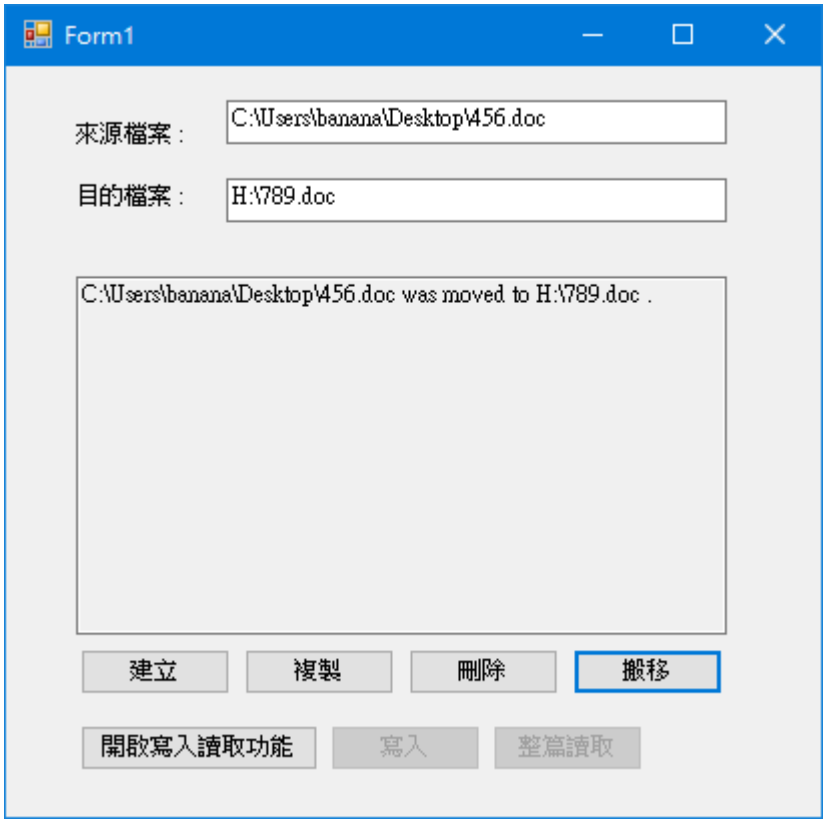

123.txt
456.doc

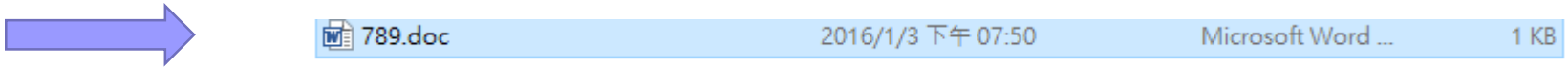
Delete


Form1
來源檔案：
C:\Users\banana\Desktop\123.txt
目的檔案：
C:\Users\banana\Desktop\123.txt was successfully deleted.
建立
複製
刪除
搬移
開啟寫入讀取功能
寫入
整篇讀取

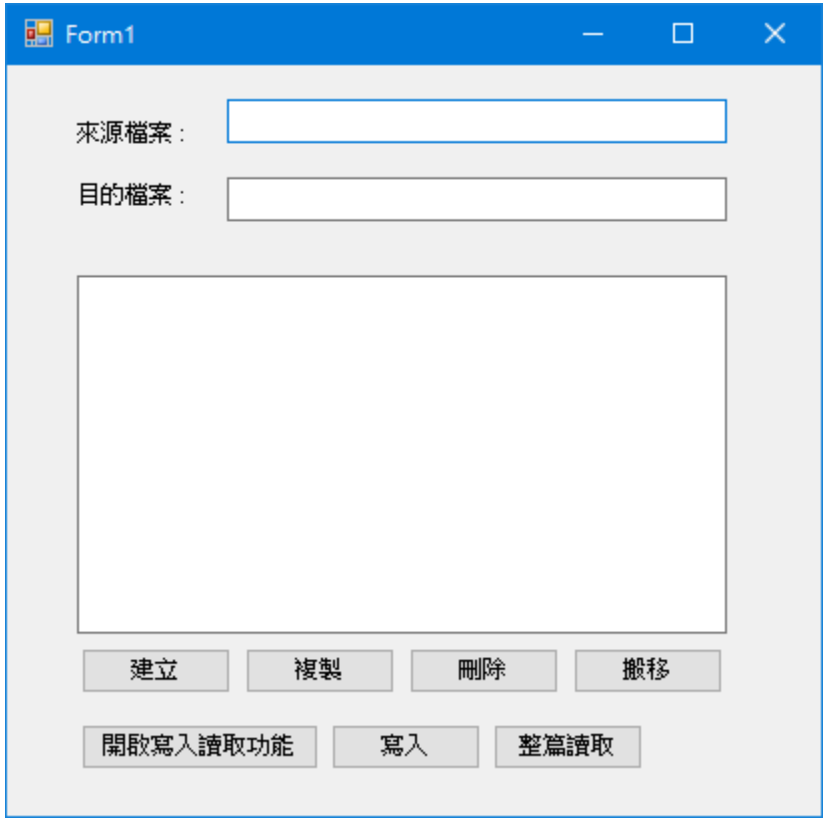

456.doc

Delete

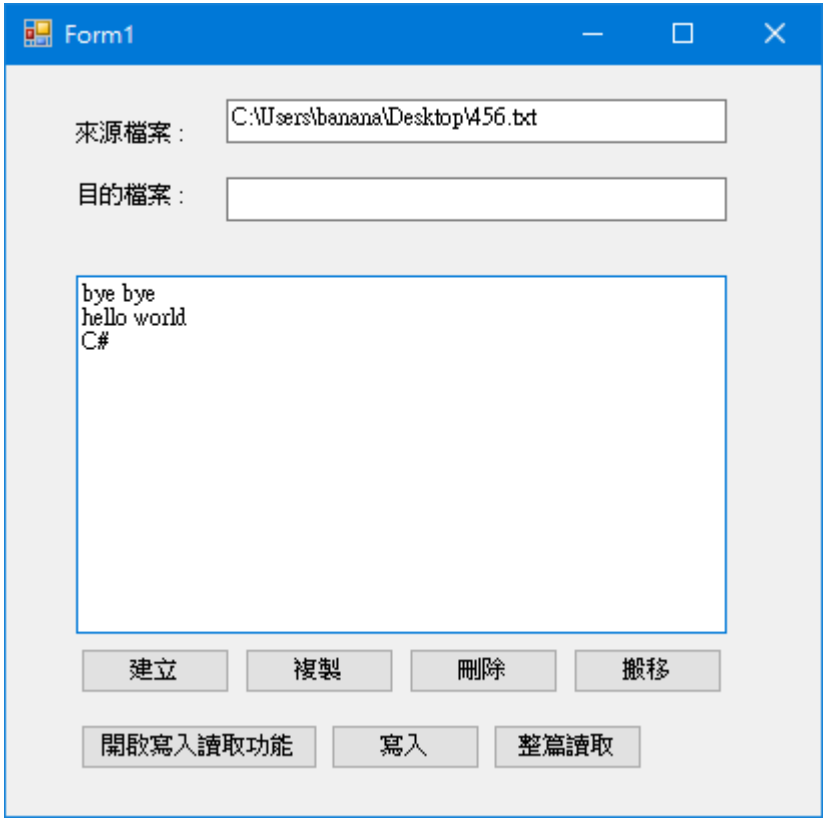

Form1
來源檔案：
C:\Users\banana\Desktop\456.doc
目的檔案：
H:\789.doc
C:\Users\banana\Desktop\456.doc was moved to H:\789.doc .
建立
複製
刪除
搬移
開啟寫入讀取功能
寫入
整篇讀取

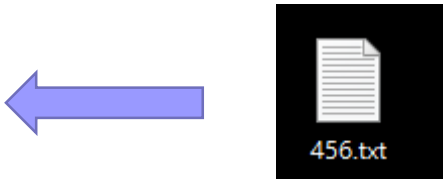

789.doc
2016/1/3 下午 07:50
Microsoft Word ...
1 KB

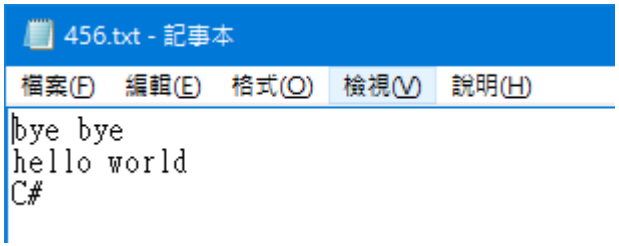
Open the store function



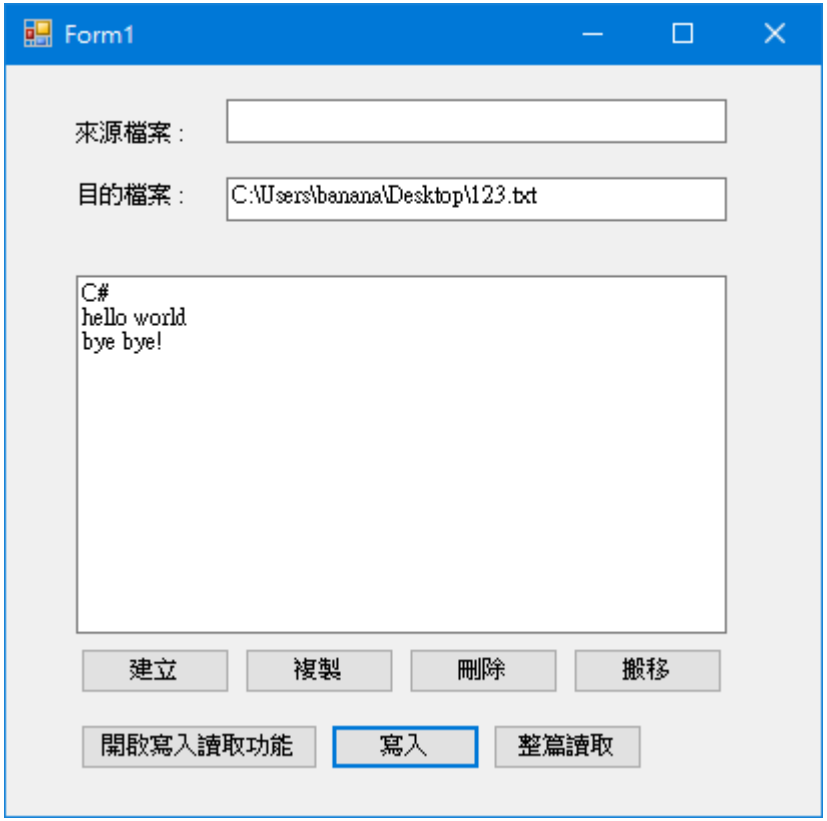
Store

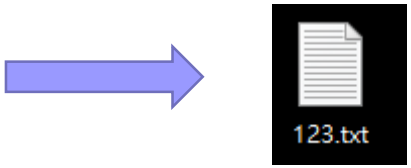

Form1
來源檔案：
C:\Users\banana\Desktop\456.txt
目的檔案：
bye bye
hello world
C#
建立
複製
刪除
搬移
開啟寫入讀取功能
寫入
整篇讀取

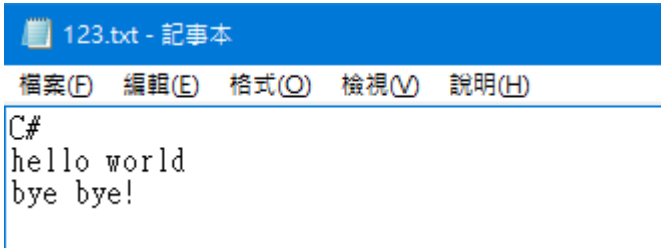

456.txt


456.txt - 記事本
檔案(F)
編輯(E)
格式(O)
檢視(V)
說明(H)
bye bye
hello world
C#

Store


Form1
來源檔案：
目的檔案：
C:\Users\banana\Desktop\123.txt
C#
hello world
bye bye!
建立
複製
刪除
搬移
開啟寫入讀取功能
寫入
整篇讀取


123.txt


123.txt - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)
C#
hello world
bye bye!

The End

Take a Break …..