

檔案與串流

人工智慧與無線感應設備開發專班

湜憶電腦知訊顧問股份有限公司

馬傳義

前言

- 從鍵盤、滑鼠或其他周邊設備所輸入的資料，都暫時儲存於主記憶體中。
- 當結束程式執行時，這些資料都會消失，下次要應用時需重新輸入。
- 為了要節省重新輸入資料的時間或避免重要資料遺失，最好將這些資料存放在檔案中。
- 有需要的時候，程式還可以將檔案內的資料讀取出來，或再將更新過的資料存入檔案中。
- Java與檔案之間的溝通，是讓一串串的資料在電腦的主記憶體中進進出出，形成了「串流」（stream）。

前言

■ Java把「磁碟」及「網路資料」存取採「串流」的方式來處理。

◆ 好處是不管存取的資料，是來自「磁碟」或「網路」，撰寫的Java程式碼都是一樣的。

檔案概論

■ 程式與資料最好分別存檔。

■ 當資料有異動時，只要修改資料檔案的內容即可，不會影響程式檔案執行；同時相同的程式檔案也可以處理不同資料檔案。

例如：

- 學校成績系統可以處理各科系、各班級的成績資料。
- 系統只有一個，但是不同的老師都可以利用此系統來處理自己科目的成績資料。

■ 檔案可分成「文字檔」與「非文字檔」兩種。

檔案概論

■ 「文字檔」是人可以直接閱讀的檔案（即人「看得懂」的檔案）。

例如：

程式的原始碼（*.java、*.py、...等）。

網頁的原始碼（*.htm、*.html、...等）。

■ 「非文字檔」是人無法直接閱讀的檔案（即人「看不懂」的檔案）。

例如：

影音檔（*.mp4、*.mp3、...等）。

圖形檔（*.gif、*.jpg、...等）。

編譯後的類別檔（*.class、...等）。

串流概論

■ Java是以「串流」的方式來處理輸入與輸出的資料。

■ 串流可以分成「輸入串流」與「輸出串流」。

◆ 輸入串流：

- 包括從鍵盤輸入的資料、從檔案讀取的輸入資料、...等。

◆ 輸出串流：

- 包括將資料處理的結果存入檔案、將資料處理的結果列印出來、將資料處理的結果顯示在螢幕上、...等。

串流概論

■ Java在資料處理上，可分為由字元（char）所組成的「字元串流」與由位元組（byte）所組成「位元組串流」。

◆ 字元串流：

- 是用來處理16個位元（16 bit）的Unicode資料。

◆ 位元組串流：

- 是用來處理8個位元（8 bit）的資料。

■ 在撰寫Java程式碼時，若需要使用串流，則會用到java.io套件裡的類別，因此在程式碼的開頭要載入java.io套件，如下：

```
import java.io.*;
```

串流概論

■ 無論處理的是輸入串流或是輸出串流，都會有需要處理「例外」的問題。

- ◆ 若是無法順利取得輸入串流或是輸出串流時，會拋出Exception的例外，此時就要用「try ... catch」來捕捉，或用「throws Exception」來處理。

注意：

如果要詳細的區分，處理輸入或輸出時所會發生的例外，應該是用IOException的例外，但是所有例外均是繼承自Exception類別（包含IOException類別），所以通常會改用Exception。

File（檔案／資料夾）類別

- 當使用純文字編輯文件時，該文件最終會以「*.txt」的格式儲存成純文字文件檔案。
- 該檔案在儲存前，使用者必須要先尋找或建立一個資料夾來存放這個檔案。
- 上面的敘述提到了資料夾與檔案，在Java語言中，可以使用File類別的物件來指定或建立資料夾與檔案實體。
- 嚴格說來，File類別它並不屬於串流的類別，
 - ◆ File類別中的方法並不會牽涉到檔案的讀寫。
 - ◆ File類別會被用來提供檔案或是目錄的相關資訊，
 - 包括了檔案的建立、大小、修改日期、存取權限等資訊、...等。

File（檔案／資料夾）類別

注意：

File類別並不提供「開啟檔案」、「關閉檔案」或進行檔案「讀、寫資料」的功能。

File（檔案／資料夾）類別

■ 建構子：

◆ File(String pathname)

- 參數是用來以「絕對路徑」的方式，建立資料夾或檔案物件。

注意：

若未指定絕對路徑，則會在程式執行時所在的位置下，建立資料夾或檔案物件。

例如：

➤ File myDir = new File("C:\\Javatest");

說明：

建立資料夾物件myDir。

myDir物件會被建立在指定的路徑下（此例為在「C:\磁碟機」中建立「Javatest」資料夾）。

File（檔案／資料夾）類別

➤ `File myFile = new File("C:\\Javatest.txt");`

說明

建立檔案物件myFile。

myFile物件會被建立在指定的路徑下（此例為在「C:\磁碟機」中建立「Javatest.txt」檔案）。

注意：

- ◆ 在Windows系統下的目錄符號「\」，剛好是Java中跳脫字元的前置符號，所以使用「\\」（「\\」的第一個「\」為跳脫字元，第二個「\」為反斜線字元）來代替實際的「\」符號。
- ◆ 如果是要在Linux平台下執行Java程式，那目錄符號要更改為「/」。

File（檔案／資料夾）類別

◆ File(File parent, String child)

- 在上層資料夾物件中，建立子資料夾或檔案物件。

例如：

➤ `File childDir = new File("C:\\Javatest", "Javatest2");`

說明：

建立檔案物件childDir。

childDir物件會被建立在指定的路徑下（此例為在「C:\\Javatest」下建立「Javatest2」子資料夾）。

➤ `File childFile = new File("C:\\Javatest ", "Javatest2.txt");`

說明：

建立檔案物件childFile。

childFile物件會被建立在指定的路徑下（此例為在「C:\\Javatest」下建立「Javatest2.txt」檔案）。

File（檔案／資料夾）類別

◆ File (String parent, String child)

- 在即有的父資料夾路徑中建立子資料夾或檔案物件。

例如：

➤ `File childDir = new File("C:\\Javatest", "next2");`

說明：

建立子資料夾物件`childDir`，指定子資料夾標題文字為『next2』。

這子資料夾實體會被建立在已存在的C:\\Javatest路徑中。


➤ `File childFile = new File("C:\\Javatest", "Javatest2.txt");`

說明：

建立檔案物件`childFile`，指定檔案標題文字為『Javatest2.txt』。

這檔案實體會被建立在已存在的C:\\Javatest路徑中。

File（檔案／資料夾）類別

 常用的方法：

◆ boolean mkdir()

- 在指定的路徑中建立資料夾。
- 若路徑不存在，則建立失敗。
- 若資料夾已存在，不重新建立。
- 當建立成功時，傳回true；建立失敗時，傳回false。

◆ boolean mkdirs()

- 在指定的路徑中建立資料夾，若路徑不存在，則先行建立該路徑，再建立本資料夾。
- 若資料夾已存在，不重新建立。
- 當建立成功時，傳回true；建立失敗時，傳回false。

File（檔案／資料夾）類別

◆ boolean createNewFile()

- 在指定的路徑中建立檔案，若路徑不存在，則出現錯誤。
- 若檔案已存在，不重新建立。
- 當建立成功時，傳回true；建立失敗時，傳回false。

◆ boolean delete()

- 刪除指定的檔案。

◆ boolean renameTo(File pathname)

- 更改資料夾或檔案的標題文字。
- 若有資料夾或檔案，傳回true；否則，傳回false。

File（檔案／資料夾）類別

◆ boolean exists()

- 檢查資料夾或檔案物件是否已建立。
- 若存在，傳回true；否則，傳回false。

◆ boolean isDirectory()

- 檢查是否為已建立的資料夾物件。
- 若是，傳回true；否則，傳回false。

◆ boolean isFile()

- 檢查是否為已建立的檔案物件。
- 若是，傳回true；否則，傳回false。

◆ String getName()

- 以字串型別取得資料夾或檔案標題文字，不包含路徑。

File（檔案／資料夾）類別

◆ String getPath()

- 以字串型別取得資料夾或檔案包含路徑的完整標題文字。

◆ String getParent()

- 以字串型別取得資料夾或檔案的上層路徑標題文字。
 -

File（檔案／資料夾）類別

◆ 程式（建立資料夾）：

```
package CH11_01;
```

```
import java.io.*;
```

```
public class CH11_01  
{
```

```
    public static void main(String[] args)  
    {
```

```
        String path_name="c:\\javadir";
```

```
        File myDir = new File(path_name);
```

```
        if(myDir.exists())
```

```
            System.out.println(path_name + "已經存在");
```

```
        else
```

```
            System.out.println(path_name + "尚未建立");
```

```
        if(myDir.mkdir())
```

```
            System.out.println(path_name + "資料夾建立成功");
```

File（檔案／資料夾）類別

```
        else  
            System.out.println(path_name + "資料夾建立失敗");  
    }  
}
```

File（檔案／資料夾）類別

```
1 package CH11_01;
2
3 import java.io.*;
4
5 public class CH11_01
6 {
7     public static void main(String[] args)
8     {
9         String path_name="c:\\javadir";
10         File myDir = new File(path_name);
11
12         if(myDir.exists())
13             System.out.println(path_name + "已經存在");
14         else
15             System.out.println(path_name + "尚未建立");
16
17         if(myDir.mkdir())
18             System.out.println(path_name + "資料夾建立成功");
19         else
20             System.out.println(path_name + "資料夾建立失敗");
21     }
22 }
```

File（檔案／資料夾）類別

◆ 執行結果：

```
c:\javadir尚未建立  
c:\javadir資料夾建立成功
```

◆ 說明：

- 行01：
 - ◆ 定義「套件（package）」。
- 行03：
 - ◆ 載入「java.io.*」套件。
- 行09：
 - ◆ 宣告字串變數，並指定初值。
- 行10：
 - ◆ 利用File(String pathname)建構子，建立myDir物件。
 - ◆ 即在『C磁碟機』中，建立名為『javadir』的資料夾。

File（檔案／資料夾）類別

- 行12~行15：
 - ◆ 判斷在C磁碟機下，是否有javadoc資料夾。
- 行17~行20：
 - ◆ 判斷資料夾物件是否建立成功。
 - ◆ 雖然mkdir()是建立資料夾的方法，但要建立的資料夾已經存在時，則不會重新建立，且因為建立資料夾失敗時，會傳回false，故用此方法來判斷資料夾是否建立成功。

File（檔案／資料夾）類別

◆ 程式（建立檔案）：

```
package CH11_02;
```

```
import java.io.*;
```

```
public class CH11_02  
{
```

```
    public static void main(String[] args)  
    {
```

```
        String file_name="c:\\javadir\\javatest.txt";  
        File myFile = new File(file_name);
```

```
        if(myFile.exists())
```

```
            System.out.println(file_name + "已經存在");
```

```
        else
```

```
            System.out.println(file_name + "尚未建立");
```

```
        try  
        {
```


File（檔案／資料夾）類別

```
        if(myFile.createNewFile())
            System.out.println(file_name + "檔案建立成功");
        else
            System.out.println(file_name + "檔案建立失敗");
    }
    catch(IOException e)
    {
        System.out.println(e);
    }
}
}
```

File（檔案／資料夾）類別

```
1 package CH11_02;
2
3 import java.io.*;
4
5 public class CH11_02
6 {
7     public static void main(String[] args)
8     {
9         String file_name="c:\\javadir\\javatest.txt";
10         File myFile = new File(file_name);
11
12         if(myFile.exists())
13             System.out.println(file_name + "已經存在");
14         else
15             System.out.println(file_name + "尚未建立");
16
17         try
18         {
19             if(myFile.createNewFile())
20                 System.out.println(file_name + "檔案建立成功");
21             else
22                 System.out.println(file_name + "檔案建立失敗");
23         }
24         catch(IOException e)
25         {
26             System.out.println(e);
```

File（檔案／資料夾）類別

```
27 }  
28 }  
29 }
```

◆ 執行結果：

```
c:\javadoc\javatest.txt尚未建立  
c:\javadoc\javatest.txt檔案建立成功
```

◆ 說明：

- 行01：
 - ◆ 定義「套件（package）」。
- 行03：
 - ◆ 載入「java.io.*」套件。
- 行09：
 - ◆ 宣告字串變數，並指定初值。

File（檔案／資料夾）類別

- 行10：
 - ◆ 利用File(String pathname)建構子，建立myFile物件。
 - ◆ 即在『c:\javadir』的路徑下，建立名為『javatest.txt』的檔案。
- 行12~行15：
 - ◆ 判斷在C磁碟機下的javadir資料夾中，是否有javatest.txt檔案。
- 行17~行27：
 - ◆ 判斷檔案物件是否建立成功。
 - ◆ 雖然createNewFile()是建立檔案的方法，但要建立的檔案已經存在時，則不會重新建立，且因為建立檔案失敗時，會傳回false，故用此方法來判斷檔案是否建立成功。
 - ◆ 若發生IOException，則顯示該例外。

寫入字元資料

- 字元資料流為16 bits的Unicode文字串流，即雙位元組資料。
- 將字元資料寫入檔案，需使用FileWriter與BufferedWriter類別。
- 這種處理字元串流的類別物件稱為「串流物件」。
- FileWriter建構子：
 - ◆ FileWriter(File myFile)
 - 參數是用來以「絕對路徑」的方式，將資料寫入指定的檔案中。
 - myFile為檔案類別的物件名稱。

寫入字元資料

例如：

```
File myFile = new File("C:\\javadir\\stu.txt");  
FileWriter fileWrite = new FileWriter(myFile);
```

說明：

- ◆ 上述兩行是在指定的路徑下，建立檔案物件myFile，並建立fileWrite的寫入串流物件。

◆ FileWriter(String filename)

- 建立一個可供寫入字元資料的輸出串流物件，用來開啟輸出的字元資料檔。
- filename為檔案名稱，可包含檔案所在磁碟機、資料夾與檔名。

注意：

若在相同的路徑下，有相同名稱的檔案，則相同檔名的檔案會被覆蓋掉。

寫入字元資料

例如：

```
FileWriter fileWrite = new FileWriter("c:\\javadir\\stu.txt");
```

說明：

- ◆ fileWrite 為串流物件。
- ◆ 建立stu.txt 字元資料檔案。
- ◆ 如果該檔案已存在，會被覆蓋。

◆ FileWriter(String filename, Boolean append)

- append若設為true，則寫入的資料會加在原檔案後面；若設為false，則原檔案內容會被覆蓋不見，新資料由最前面開始寫入。
- 預設值為false。

寫入字元資料

BufferedWriter建構子：

◆ BufferedWriter(FileWriter fWriter)

- 建立記憶體緩衝區，準備輸出串流到檔案，讓資料的操作增加效能。

例如：

```
FileWriter fileWrite = new FileWriter("c:\\javadir\\stu.txt");
```

```
BufferedWriter fileOut = new BufferedWriter(fileWrite);
```

說明：

- ◆ 上述兩行是開啟緩衝區輸出串流。

◆ BufferedWriter(FileWriter fWriter, int size)

- 建立記憶體緩衝區輸出串流，並設定緩衝區佔用記憶體的大小，預設值為512Bytes。

寫入字元資料

常用的方法：

◆ void write(String str)

- 將str字串寫入輸出串流。
- 過程是先存於緩衝區，再由緩衝區寫入檔案中。

◆ void write(int c)

- 將單一字元寫入輸出串流。

◆ void newLine()

- 寫入換行字元。

◆ void flush()

- 清理緩衝區並釋放記憶空間，若緩衝區內還有資料，會全部寫入檔案中。

寫入字元資料

◆ void close()

- 關閉輸出串流，即關閉檔案。

寫入字元資料

◆ 程式：

```
package CH11_03;
```

```
import java.io.*;
```

```
public class CH11_03  
{
```

```
    public static void main(String[] args)  
    {
```

```
        File myDir = new File("c:\\javadir");  
        myDir.mkdirs();
```

```
        File childFile = new File(myDir, "stu.txt");
```

```
        try  
        {
```

```
            FileWriter filewrite = new FileWriter(childFile);  
            BufferedWriter fileout = new BufferedWriter(filewrite);  
            fileout.write("張三\t85\t90");
```

寫入字元資料

```
        fileout.newLine();  
        fileout.write("李四\t65\t67");  
        fileout.newLine();  
        fileout.flush();  
        filewrite.close();  
    }  
    catch(IOException e)  
    {  
        System.out.println("檔案處理有誤！");  
    }  
}  
}
```

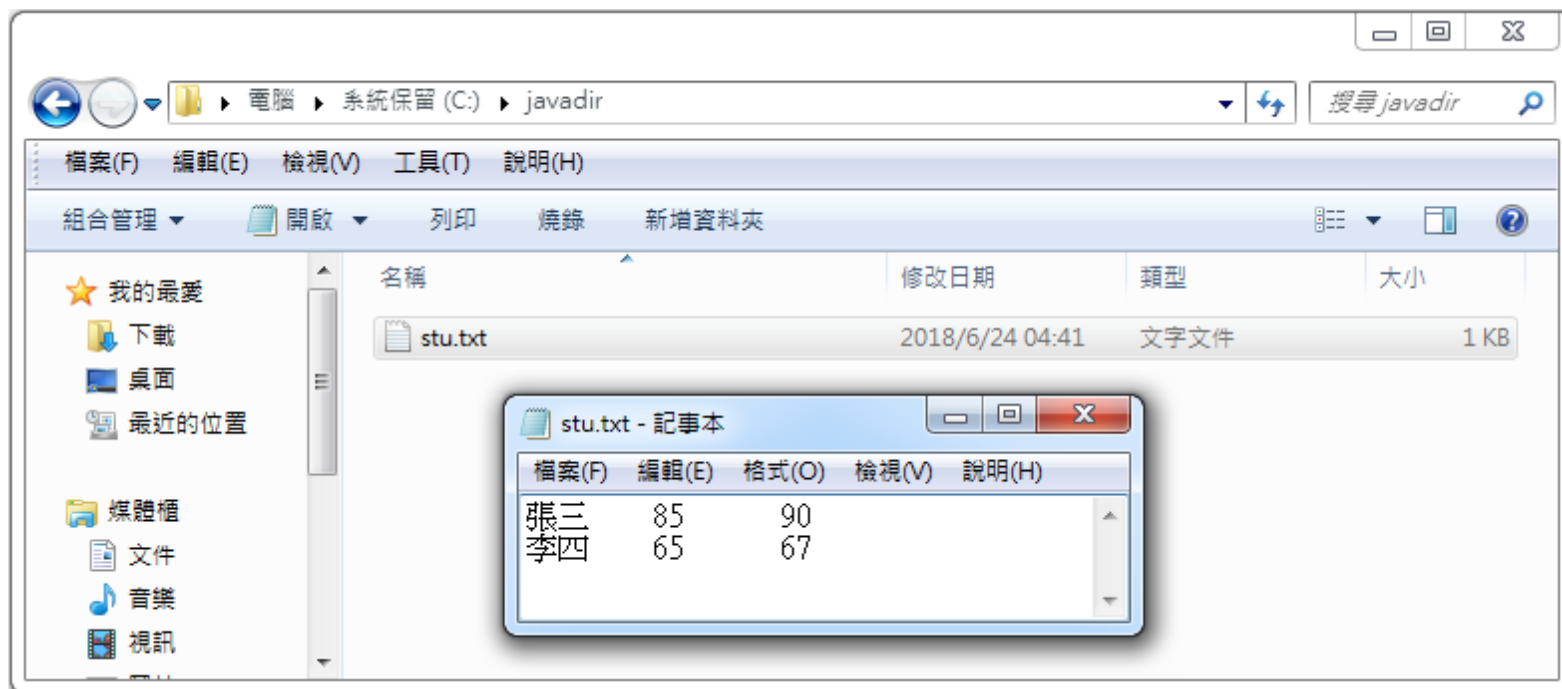
寫入字元資料

```
1 package CH11_03;
2
3 import java.io.*;
4
5 public class CH11_03
6 {
7     public static void main(String[] args)
8     {
9         File myDir = new File("c:\\javadir");
10        myDir.mkdirs();
11
12        File childFile = new File(myDir, "stu.txt");
13
14        try
15        {
16            FileWriter filewrite = new FileWriter(childFile);
17            BufferedWriter fileout = new BufferedWriter(filewrite);
18            fileout.write("張三\t85\t90");
19            fileout.newLine();
20            fileout.write("李四\t65\t67");
21            fileout.newLine();
22            fileout.flush();
23            filewrite.close();
24        }
25        catch(IOException e)
26        {
```

寫入字元資料

```
27     System.out.println("檔案處理有誤！");  
28 }  
29 }  
30 }
```

◆ 執行結果：



寫入字元資料

◆ 說明：

- 行01：
 - ◆ 定義「套件（package）」。
- 行03：
 - ◆ 載入「java.io.*」套件。
- 行09：
 - ◆ 利用File(String pathname)建構子，建立myDir物件。
- 行10：
 - ◆ 利用myDir物件，建立資料夾。
 - ◆ 不確定在指定的路徑是否存在，故使用需使用 mkdirs()（若路徑不存在，先建立路徑，再建立資料夾）。

寫入字元資料

- 行12：
 - ◆ 利用File(File parent, String child)建構子，建立childFile物件。
 - ◆ 即在父資料夾（myDir）物件中，建立檔案物件childFile，其檔案名稱為『stu.txt』。
- 行14~行28：
 - ◆ 寫入資料到檔案。
 - ◆ 行16
 - ◆ 利用FileWriter(File myFile)建構子，建立filewrite物件。
 - ◆ 即建立一個可供寫入字元資料的輸出串流物件，用來開啟輸出的字元資料檔。
 - ◆ 行17：
 - ◆ 利用BufferedWriter(FileWriter fWriter)建構子，建立fileout物件。

寫入字元資料

- ◆ 即建立記憶體緩衝區，準備輸出串流到檔案，讓資料的操作增加效能。
- ◆ 行18、行20：
 - ◆ 寫入一筆資料。
- ◆ 行19、行21：
 - ◆ 寫入換行字元，使往下移一行。
- ◆ 行22：
 - ◆ 清理緩衝區。
 - ◆ 若緩衝區內還有資料，會全部寫入檔案中。
- ◆ 行23：
 - ◆ 關閉檔案。
- ◆ 行27：
 - ◆ 若路徑有錯或無法寫入此檔案，會出現「檔案處理有誤！」訊息。

讀取字元資料

■ 若要由檔案中讀出字元資料，需使用FileReader與BufferedReader類別。

■ FileReader建構子：

◆ FileReader(File myFile)

- 參數是用來以「絕對路徑」的方式，將資料寫入指定的檔案中。
- myFile為檔案類別的物件名稱。

例如：

```
File myFile = new File("c:\\javadir\\stu.txt");  
FileReader fileRead = new FileReader(myFile);
```

說明：

- ◆ 上述兩行是資料寫入指定的檔案中。

讀取字元資料

◆ FileReader(String filename)

- 建立一個可供讀取字元資料的輸入串流物件，用來從指定的檔案取出資料。

例如：

```
FileWriter fileWrite = new FileWriter("c:\\javadir\\stu.txt");
```

■ BufferedReader建構子：

◆ BufferedReader(FileReader fRead)

- 建立記憶體緩衝區，準備輸入串流給程式處理，讓資料的操作增加效能。

例如：

```
FileReader fileRead = new FileReader("c:\\javadir\\stu.txt");
```

```
BufferedReader fileIn = new BufferedReader(fileRead);
```

- ◆ 上述兩行是開啟緩衝區輸入串流。

讀取字元資料

◆ `BufferedReader(FileWriter fRead, int size)`

- 建立記憶體緩衝區輸入串流，並設定緩衝區大小，預設值為512 Bytes。

■ 常用方法：

◆ `String readLine()`

- 讀取一行字串。
- 資料由檔案取出到緩衝區，再由緩衝區取出給程式處理。

◆ `int read()`

- 讀取單一字元。

◆ `long skip(long n)`

- 跳過n個字元。

讀取字元資料

◆ void close()

- 關閉輸入串流，即關閉檔案。

讀取字元資料

◆ 程式：

```
package CH11_04;
```

```
import java.io.*;
```

```
public class CH11_04  
{
```

```
    public static void main(String[] args)  
    {
```

```
        File myFile = new File("c:\\javadir\\stu.txt");
```

```
        if(! myFile.exists())
```

```
            System.out.println("檔案不存在！");
```

```
        else
```

```
        {
```

```
            String data;
```

```
            try
```

```
            {
```

讀取字元資料

```
FileReader fileread = new FileReader(myFile);
BufferedReader filein = new BufferedReader(fileread);

do
{
    data = filein.readLine();
    if(data == null)
        break;

    System.out.println(data);
}while(true);

fileread.close();
}
catch(IOException e)
{
    System.out.println("檔案處理有誤！");
}
}
```

讀取字元資料

}

}

讀取字元資料

```
1 package CH11_04;
2
3 import java.io.*;
4
5 public class CH11_04
6 {
7     public static void main(String[] args)
8     {
9         File myFile = new File("c:\\javadir\\stu.txt");
10
11         if(! myFile.exists())
12             System.out.println("檔案不存在！");
13         else
14         {
15             String data;
16
17             try
18             {
19                 FileReader fileread = new FileReader(myFile);
20                 BufferedReader filein = new BufferedReader(fileread);
21
22                 do
23                 {
24                     data = filein.readLine();
25                     if(data == null)
26                         break;
```

讀取字元資料

```
27
28         System.out.println(data);
29     } while(true);
30
31     fileread.close();
32 }
33 catch(IOException e)
34 {
35     System.out.println("檔案處理有誤!");
36 }
37 }
38 }
39 }
```

◆ 執行結果：

張三	85	90
李四	65	67

◆ 說明：

• 行01：

- ◆ 定義「套件（package）」。

讀取字元資料

- 行03：
 - ◆ 載入「java.io.*」套件。
- 行09：
 - ◆ 利用File(String pathname)建構子，建立myDir物件。
- 行11~行37：
 - ◆ 判斷要讀取資料的檔案是否存在。
 - ◆ 行12：
 - ◆ 若不存在，顯示訊息。
 - ◆ 行15：
 - ◆ 宣告字串變數。
 - ◆ 行19：
 - ◆ 利用FileReader(File myFile)建構子，建立fileread物件。

讀取字元資料

- ◆ 即 建立一個可供讀取字元資料的輸入串流物件，用來開啟輸入的字元資料檔。
- ◆ 行20：
 - ◆ 利用BufferedReader(FileReader fRead)建構子，建立filein物件。
 - ◆ 即 建立記憶體緩衝區，準備輸入串流給程式處理，讓資料的操作增加效能。
- ◆ 行22~行29：
 - ◆ 讀出資料。
 - ◆ 行24：

讀取一行字串資料，並指定給字串變數。
 - ◆ 行25~行26：

若所字串變數的內容為空字串，表示後面沒有資料，則離開迴圈。

讀取字元資料

- ◆ 行28：
 - ◆ 顯示讀出的資料。
- ◆ 行31：
 - ◆ 關閉檔案。
- ◆ 行35：
 - ◆ 若路徑有錯或無法讀出此檔案，會出現「檔案處理有誤！」訊息。

位元組資料的存取

- 位元組資料流為8 bits的位元組串流。
- 讀出與寫入需要使用FileInputStream與FileOutputStream類別。
- 建構子：
 - ◆ FileInputStream(String filename)
 - 建立一個可供讀取位元組資料的串流物件。
 - ◆ FileOutputStream(String filename)
 - 建立一個可供寫入位元組資料的串流物件。
 - 同名稱的檔案會被覆蓋掉。

位元組資料的存取

常用方法：

◆ `int available()`

- 取得輸入位元組資料串流所佔的位元組大小。

◆ `int read(byte[] b)`

- 將所讀取的輸入位元組資料串流存放到位元組陣列 `b` 中。

◆ `int write(byte[] b)`

- 將存放在位元組陣列 `b` 內的資料寫入到輸出串流物件。

◆ `void close()`

- 關閉位元組串流。

位元組資料的存取

◆ 程式（複製圖形檔；需在CMD中執行）：

```
import java.io.*;
```

```
public class CH11_05
```

```
{
```

```
    public static void main(String[] args)
```

```
    {
```

```
        String file_name = "images.jpg";
```

```
        try
```

```
        {
```

```
            FileInputStream filein = new FileInputStream(file_name);
```

```
            byte data[] = new byte[filein.available()];
```

```
            filein.read(data);
```

```
            filein.close();
```

```
            FileOutputStream fileout = new FileOutputStream("複製_" + file_name);
```

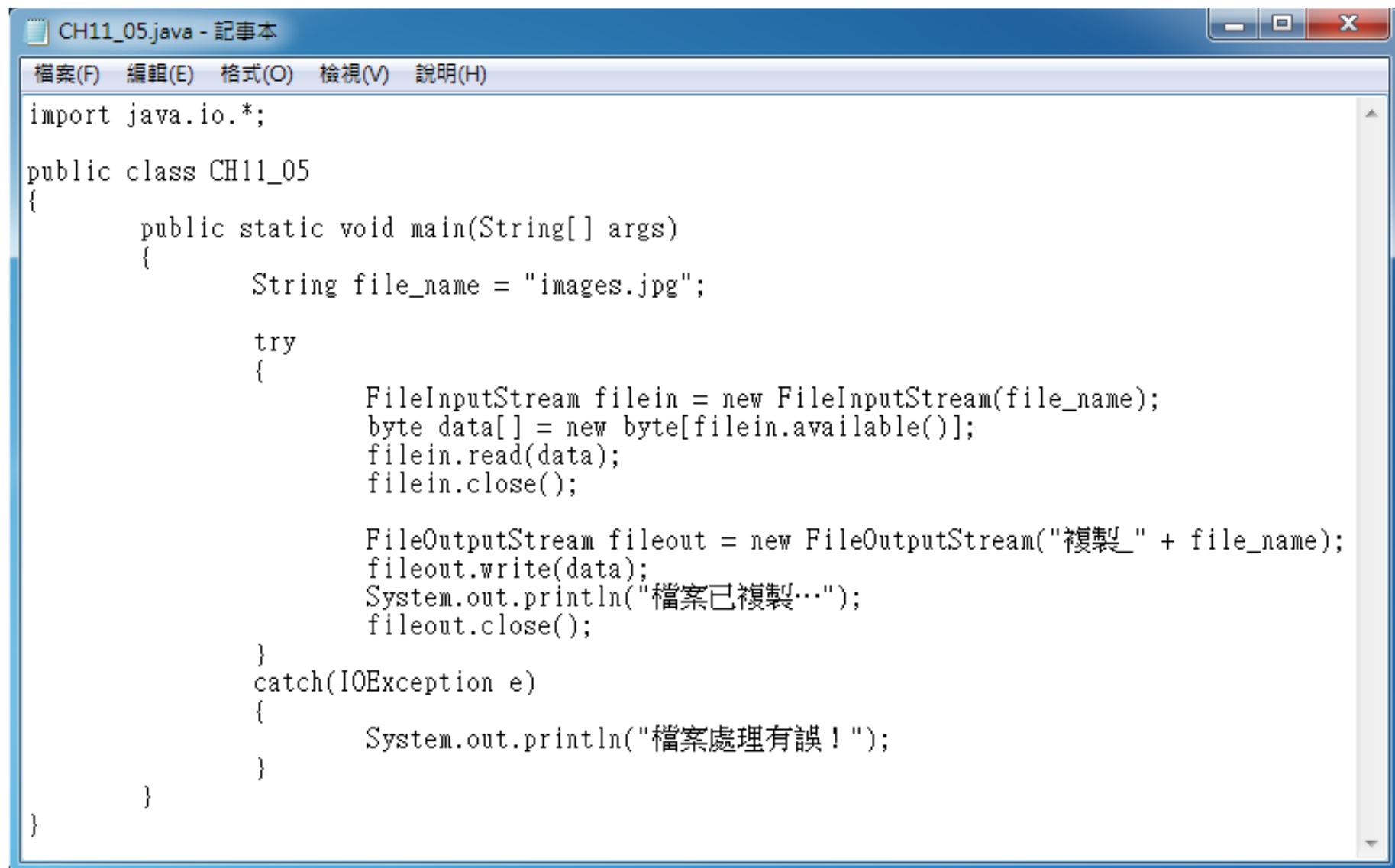
```
            fileout.write(data);
```

```
            System.out.println("檔案已複製...");
```


位元組資料的存取

```
        fileout.close();
    }
    catch(IOException e)
    {
        System.out.println("檔案處理有誤！");
    }
}
}
```

位元組資料的存取



```
CH11_05.java - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

import java.io.*;

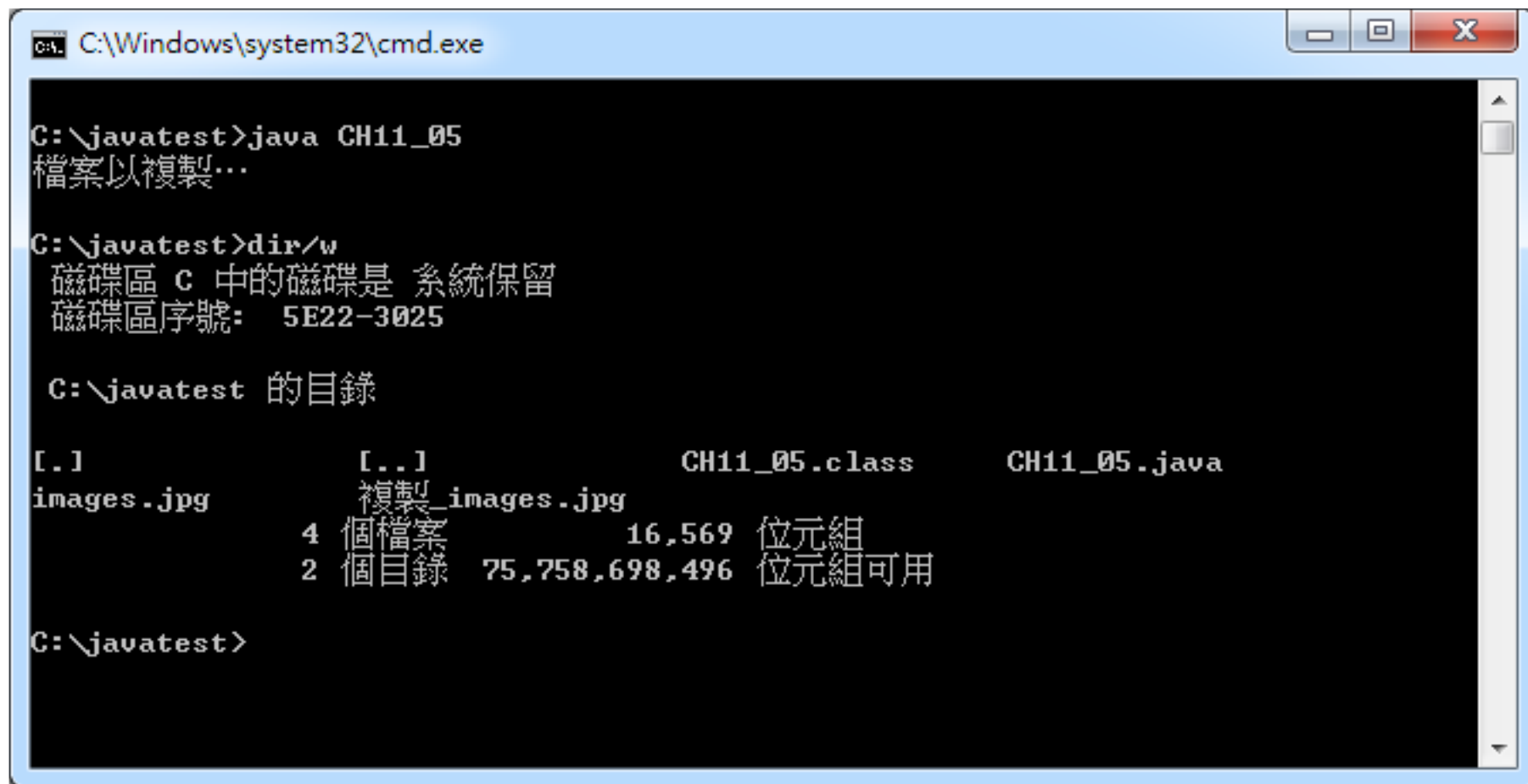
public class CH11_05
{
    public static void main(String[] args)
    {
        String file_name = "images.jpg";

        try
        {
            FileInputStream filein = new FileInputStream(file_name);
            byte data[] = new byte[filein.available()];
            filein.read(data);
            filein.close();

            FileOutputStream fileout = new FileOutputStream("複製_" + file_name);
            fileout.write(data);
            System.out.println("檔案已複製...");
            fileout.close();
        }
        catch(IOException e)
        {
            System.out.println("檔案處理有誤!");
        }
    }
}
```

位元組資料的存取

◆ 執行結果：



```
C:\Windows\system32\cmd.exe

C:\javatest>java CH11_05
檔案以複製...

C:\javatest>dir/w
磁碟區 C 中的磁碟是 系統保留
磁碟區序號: 5E22-3025

C:\javatest 的目錄

[.]                [...]                CH11_05.class      CH11_05.java
images.jpg         複製 images.jpg
                  4 個檔案                16,569 位元組
                  2 個目錄              75,758,698,496 位元組可用

C:\javatest>
```

位元組資料的存取

◆ 說明：

- 行01：
 - ◆ 載入「java.io.*」套件。
- 行07：
 - ◆ 宣告字串變數，並指定初值。
- 行11：
 - ◆ 利用 `FileInputStream(String filename)` 建構子，建立 `filein` 物件。
 - ◆ 即取得輸入串流。
- 行12：
 - ◆ 取得輸入串流所佔的位元組大小，並建立一個同位元組元素數量的 `data` 陣列。

位元組資料的存取

- 行13：
 - ◆ 將輸入串流存放到data陣列。
- 行14：
 - ◆ 關閉檔案。
- 行16：
 - ◆ 利用FileOutputStream(String filename)建構子，建立fileout物件。
 - ◆ 即取得輸出串流。
 - ◆ 建立一個可供寫入位元資料的串流物件，預定要將串流輸出的檔案名稱為「複製- images.jpg」。
 - ◆ 若有同名稱的檔案會被覆蓋掉。
- 行17：
 - ◆ 將存放在位元組陣列data內的資料寫入到輸出串流物件。

位元組資料的存取

- 行18：
 - ◆ 顯示訊息。
- 行19：
 - ◆ 關閉檔案。
- 行23：
 - ◆ 若有發生錯誤或無法讀出此檔案，會出現「檔案處理有誤！」訊息。

資料來源

- 蔡文龍、何嘉益、張志成、張力元，JAVA SE 10基礎必修課，台北市，碁峰資訊股份有限公司，2018年7月，出版。
- 吳燦銘、胡昭民，圖解資料結構-使用Java(第三版)，新北市，博碩文化股份有限公司，2018年5月，出版。
- Ivor Horton，Java 8 教學手冊，台北市，碁峰資訊股份有限公司，2016年9月，出版。
- 李春雄，程式邏輯訓練入門與運用---使用JAVA SE 8，台北市，上奇科技股份有限公司，2016年6月，初版。
- 位元文化，Java 8視窗程式設計，台北市，松崗資產管理股份有限公司，2015年12月，出版。
- Benjamin J Evans、David Flanagan，Java 技術手冊 第六版，台北市，碁峰資訊股份有限公司，2015年7月，出版。
- 蔡文龍、張志成，JAVA SE 8 基礎必修課，台北市，碁峰資訊股份有限公司，2014年11月，出版。
- 陳德來，Java SE 8程式設計實例，台北市，上奇科技股份有限公司，2014年11月，初版。
- 林信良，Java SE 8 技術手冊，台北市，碁峰資訊股份有限公司，2014年6月，出版。
- 何嘉益、黃世陽、李篤易、張世杰、黃鳳梅，徐政棠譯，JAVA2 程式設計從零開始--適用JDK7，台北市，上奇資訊股份有限公司，2012年5月，出版。