第十四章
檔案處理

本章學習目標
- 認識串流(Stream)
- 學習檔案的開啟與關閉
- 學習如何處理文字檔
- 學習如何處理二進位檔

文字檔(text file)　二進位檔(binary file)

- Text file:
  - A **text file** stores data in the form of alphabets, digits and other special symbols by storing their **ASCII** values
  - and are in a **human readable format**.
  - For example, any file with a **.txt**, .c, etc extension.
- Binary file:
  - A **binary file** contains a sequence or a collection of **bytes**
  - which are **not in a human readable format**.
  - For example, files with **.exe**, **.mp**3, etc extension.

本章重點

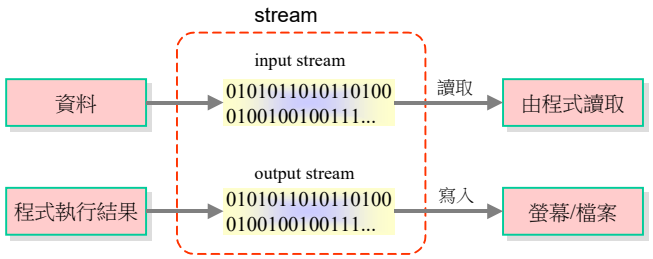
- 檔案處理：處理檔案read, write, save等動作
- 文字檔：可以當成文字來閱讀的檔案，例如：*.java, html的程式碼
- 二進位檔：binary file, 例如：音樂檔，圖形檔，編譯後的類別檔
- Method
  - (1) 純文字檔的存取：file writer直接寫入磁碟; BufferedWriter寫入Buffer
  - (2) 二進位檔(或文字檔)的存取：File input stream

14.1 關於串流

14-2

串流可分為「輸入串流」（**input stream**）與「輸出串流」（**output stream**）兩種。

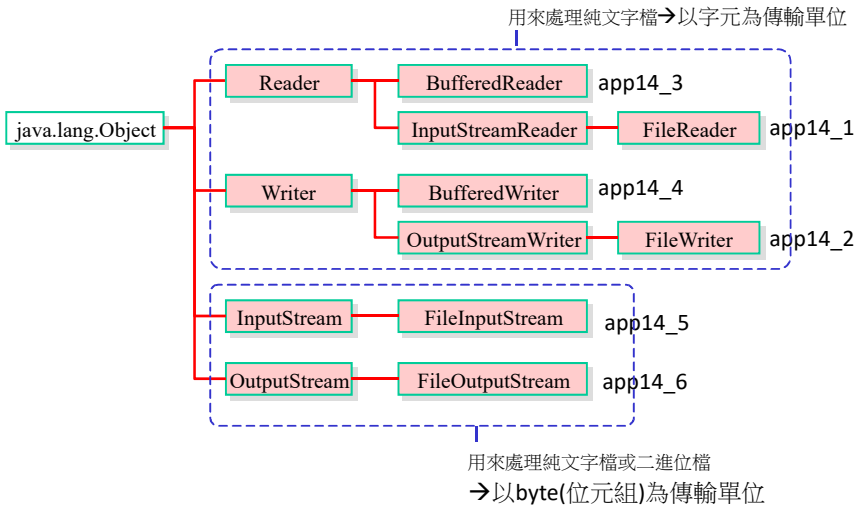
下圖說明了串流如何做為檔案處理的橋樑：



Stream(串流) 類似切片段的多工處理，像鏈條一樣排列變成一個管道，亦即每個 item 讀完後再讀下一個 item。而使用並行去處理時，資料會被分成多個段，每段都在不同的執行緒中處理，然後將結果一起輸出。

14-3

下圖列出了與檔案相關類別的繼承圖。



14-4

14.2 檔案的基本處理

下面列出了Reader與Writer類別所提供的method。

表14.2.1 Reader類別的method

method	主要功能
void close()	關閉串流
int read()	讀取串流中的一個字元
int read(char[] cbuf)	**從串流讀取資料，放到字元陣列cbuf中，並傳回所讀取字元的總數**
int read(char[] cbuf, int off, int len)	**從串流讀取資料，並放到陣列cbuf的某個範圍（off表示陣列索引值，len表示讀取字元數）**
long skip(long n)	跳過n個字元不讀取

14-5

表14.2.2 Writer類別的method

method	主要功能
void close()	關閉串流
abstract void **flush()**	將**緩衝區的資料寫到檔案裡**。 注意這是抽象method，其明確的定義是撰寫在Writer的子類別裡
void write(char[] cbuf)	將字元陣列輸出到串流
void write(char[] cbuf, int off, int len)	將字元陣列依指定的格式輸出到串流中（off表示陣列索引值，len表示寫入字元數）
void write(int c)	將單一字元c輸出到串流中
void write(String str)	將字串str輸出到串流中
void write(String str, int off, int len)	將字串str輸出到串流（off表示陣列索引值，len表示寫入字元數）

14-6

14.2.1 讀取檔案的內容--使用FileReader類別

FileReader類別可用來讀取文字檔。

FileReader() 建構元的格式可參考下表：

表14.2.3 FileReader建構元

建構元	主要功能
FileReader(String name)	依檔案名稱建立一個可供讀取字元的輸入串流物件

下面的範例說明了如何讀取文字檔train.txt：

14-7

```
01  // app14_1, 使用FileReader類別讀取檔案
02  import java.io.*;       // 載入java.io類別庫裡的所有類別
03  public class app14_1
04  {
05    public static void main(String args[]) throws IOException
06    {
07      char data[]=new char[128];  // 建立可容納128個字元的陣列
08      FileReader fr=new FileReader("c:\\Java\\train.txt"); // 建立物件fr
09
10      int num=fr.read(data); // 將資料讀入字元串列data內,並傳回讀入的字數
11      String str=new String(data,0,num); // 將字元串列轉換成字串,從索引0開始印出num個字元數
12      System.out.println("Characters read= "+num);
13      System.out.println(str);
14      fr.close();
16    }
17  }
```

```
/* app14_1 OUTPUT---
Characters read= 29
火車快飛，火車快飛
越過高山，飛過小溪
不知走了幾百里
-------------------------------*/
```

fr.close() 是關檔的動作，若已經關檔，接著再次讀取已經關的檔案，就會拋出例外[如下頁範例]。

```
// app14_1, 使用FileReader類別讀取檔案
  import java.io.*;          // 載入java.io類別庫裡的所有類別
  public class ex01
  {
     public static void main(String args[]) throws IOException
     {
        char data[]=new char[128];// 建立可容納128個字元的陣列
        FileReader fr=new FileReader("c:\\Java\\train.txt"); // 建立物件fr

      int num=fr.read(data); // 將資料讀入字元串列data內,並傳回讀入的字數
      String str=new String(data,0,num);  // 將字元串列轉換成字串,從索引0開始印出num個字元數
      System.out.println("Characters read= "+num);
      System.out.println(str);

      fr.close();

      int nm=fr.read(data);
     }
  }
```

```
java.io.IOException: Stream closed
at sun.nio.cs.StreamDecoder.ensureOpen(Unknown
Source)
at sun.nio.cs.StreamDecoder.read(Unknown Source)
at java.io.InputStreamReader.read(Unknown Source)
at java.io.Reader.read(Unknown Source)
at ex01.main(ex01.java:17)
```

中文字在eclipse亂碼的問題

- .txt預設是UTF-8 所以可以eclipse更改成UTF-8
- 或者.txt另存成ANSI格式，即可搭配eclipse正確顯示中文

- **Eclipse 環境設定**
  - 選擇「Window」>「Preferences」進行設定
  - 在「General」>「Workspace」裡面設定「Text file encoding」將Default (MS950) **改成**「Other:」選擇「**UTF-8**」之後按下「Apply」並按「OK」結束設定

14-28

Java把一個中文字看成是一個字元，在Windows裡，Java把換行字元「\r\n」看成是兩個字元，如下圖所示：

\r return，把游標移到當前行的最左邊
\n newline，換行
\r\n = enter的動作

ex14_2_1.java

請在記事本裡建好**donkey.txt**，並完成下列問題：

(a) 請利用**FileReader** 讀取**donkey.txt**，將檔案內容列印出來，並計算讀取的字元數。

(b) 在**donkey.txt** 裡共有中文字**26** 個，與程式中所計算的字元數一樣嗎？為什麼？

若讀到亂碼，可更改txt的編碼格式(ANSI)
或者更改eclipse的存檔類型(Preference中更改)

目的是只要兩方讀檔格式一致即可

```
/* output----------
Characters read= 30
我有一隻小毛驢
我從來也不騎
有一天我心血來潮騎著去趕集
----------*/
```

14-9

14.2.2 將資料寫入檔案--使用FileWriter類別

FileWriter類別可用來將字元型態的資料寫入檔案。

FileWriter() 建構元的格式可參考下表：

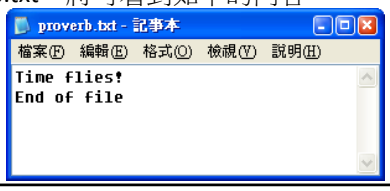
表14.2.4 FileWriter建構元

建構元	主要功能
FileWriter(String filename)	依檔案名稱建立一個可供寫入字元資料的串流物件，原先的檔案會被覆蓋
FileWriter(String filename, Boolean a)	同上，但如果a設為true，則會將資料附加在原先的資料後面

下面的範例說明如何以FileWriter類別將字元陣列與字串寫到檔案裡 14-10

```
01  // app14_2, 使用FileWriter類別將資料寫入檔案內
02  import java.io.*;
03  public class app14_2
04  {
05    public static void main(String args[]) throws IOException
06    {
07      FileWriter fw=new FileWriter("c:\\Java\\proverb.txt");
08      char data[]={'T','i','m','e',' ','f','l','i','e','s','!','\r','\n'};
09      String str="End of file";
10      fw.write(data);   // 將字元陣列寫到檔案裡
11      fw.write(str);    // 將字串寫到檔案裡
12      fw.close();
13    }
14  }
```

用記事本開啟proverb.txt，將可看到如下的內容：



ex14_2_2.java

請利用**FileWriter** 類別，將字元陣列**hi** 寫入檔案 **hello.txt** 中。

```
char hi[]={'H','e','l','l','o',',','J','a','v','a','!','\r','\n'};
```

14.2 回家作業
檔案的基本處理

hw14_2_1.java

試改寫ex14_2_1，將 "我有一隻小毛驢" 一行忽略不讀取。

```
/* output----------
Characters read= 21
我從來也不騎
有一天我心血來潮騎著去趕集
----------*/
```

long skip(long n)	跳過n個字元不讀取

17

14.2 回家作業
檔案的基本處理

hw14_2_2.java

接續ex14_2_2 ， 先開啟文字檔hello.txt ， 在原先檔案內容的後面再寫入字串"Welcome!"，然後印出整個檔案內容（字串 "Welcome!" 請撰寫在新的一行）。

```
//-------------------------------------------------
....
FileWriter fw = new FileWriter("gg.txt");//覆蓋原有
.....
//-------------------------------------------------
......
FileWriter fw = new FileWriter("gg.txt", true); //加在尾端
.......
```

```
/* output----------
Hello Java!
Welcome!
----------*/
```

18

```
//-----------------------------------------------------
import java.io.*;
public class A{
public static void main(String [] args) throws Exception{
FileWriter fw = new FileWriter("gg.txt");//覆蓋原有
fw.write("你好!!");
fw.close();
}
}

//-----------------------------------------------------
import java.io.*;
public class A{
public static void main(String [] args) throws Exception{
FileWriter fw = new FileWriter("gg.txt", true); //加在尾端
fw.write("你好!!");
fw.close();
}
}
```

19

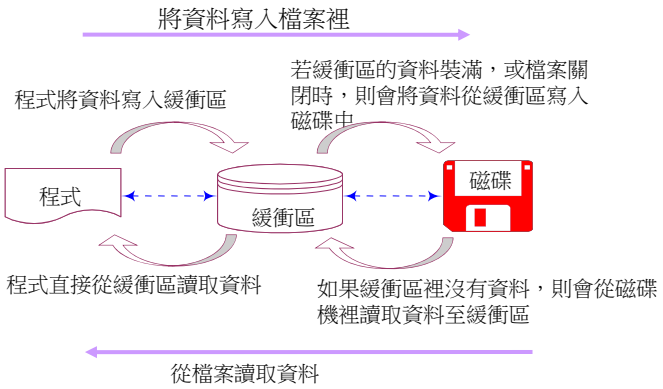
14.3 利用緩衝區來讀寫資料

14-11

★具有緩衝區的檔案處理方式是在存取時，會先將資料放置到緩衝區，而不會直接存取磁碟。

★優點在於不需要不斷地做磁碟讀取，可以提升執行效率

★ 缺點是緩衝區會佔用一塊記憶體空間，如果沒有關閉檔案或系統當機，則緩衝區的資料會因尚未寫入磁碟而造成資料流失。



14-12

14.3.1 從緩衝區讀取資料--使用BufferedReader類別

下表列出了BufferedReader類別常用的建構元與method：

表14.3.1 BufferedReader的建構元

建構元	主要功能
BufferedReader(Reader in)	建立緩衝區字元讀取串流
BufferedReader(Reader in, int size)	建立緩衝區字元讀取串流，並設定緩衝區大小

表14.3.2 BufferedReader的method

method	主要功能
void close()	關閉串流
int read()	讀取單一字元
int read(char[] cbuf, int off, int len)	讀取字元陣列（off表示陣列索引值，len表示讀取位元數）
long skip(long n)	跳過n個字元不讀取
String readLine()	**讀取一行字串**

14-13

下面是以BufferedReader類別讀取number.txt的範例：

```
01  // app14_3, 從緩衝區裡讀入資料
02  import java.io.*;
03  public class app14_3
04  {
05    public static void main(String args[]) throws IOException
06    {
07      String str;
08      int count=0;
09      FileReader fr=new FileReader("c:\\Java\\number.txt");
10      BufferedReader bfr=new BufferedReader(fr);
11
12      while((str=bfr.readLine())!=null)  // 每次讀取一本文，直到檔案結束
13      {
14        count++;    // 計算讀取的行數
15        System.out.println(str);
16      }
17      System.out.println(count+" lines read");
18
19      fr.close();   // 關閉檔案
20    }
21  }
```

```
/* app14_3 OUTPUT---
12
34
63
14
16
56
6 lines read
----------*/
```

ex14_3_1.java

請將**aaa.txt**、**bbb.txt** 的內容分別列印出來。

Note:
aaa.txt檔案中的文字 **Look before you leap.**
bbb.txt檔案中的文字 **Make hay while the sun shines.**

```
/* output----------------------
Look before you leap.
Make hay while the sun shines.
-------------------------------*/
```


14.3.2 將資料寫入緩衝區--使用BufferedWriter類別

下表列出了BufferedWriter類別常用的建構元與method：

表14.3.3 BufferedWriter的建構元

建構元	主要功能
BufferedWriter(Writer out)	建立緩衝區字元寫入串流
BufferedWriter(Writer out, int size)	建立緩衝區字元寫入串流，並設定緩衝區的大小

表14.3.4 BufferedWriter的method

method	主要功能
void close()	關閉串流
void flush()	寫入緩衝區內的字元到檔案裡
void newLine()	寫入換行字元
void writer(int c)	寫入單一字元
void writer(char[] cbuf, int off, int len)	寫入字元陣列（off表示陣列索引值，len表示讀取位元數）
void writer(String s, int off, int len)	寫入字串（off與len代表的意義同上）

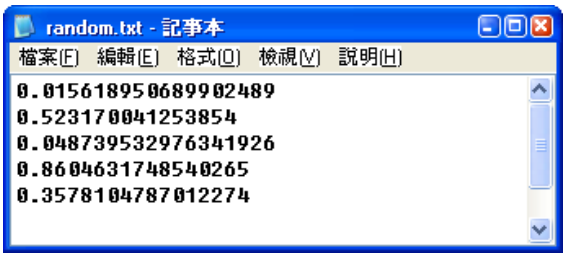
14-15

下面的範例說明了如何使用**BufferedWrite**：


```
01  // app14_4, 將資料寫到緩衝區內
02  import java.io.*;
03  public class app14_4
04  {
05    public static void main(String args[]) throws IOException
06    {
07      FileWriter fw=new FileWriter("c:\\Java\\random.txt");
08      BufferedWriter bfw=new BufferedWriter(fw);
09
10      for(int i=1;i<=5;i++)
11      {
12        bfw.write(Double.toString(Math.random())); // 寫入亂數到緩衝區
13        bfw.newLine();    // 寫入換行符號
14      }
15      bfw.flush();          // 將緩衝區內的資料寫到檔案裡
16      fw.close();           // 關閉檔案
17    }
18  }
```

14-16

如果打開random.txt，將可看到如下的類似畫面：



ex14_3_2.java

試撰寫一程式，可讀取文字檔**aaa.txt** 與 **bbb.txt**，將其內容合併後，存成檔案**ccc.txt**。

Note:
aaa.txt檔案中的文字 **Look before you leap.**

bbb.txt檔案中的文字 **Make hay while the sun shines.**

ccc.txt合併檔案後的文字
Look before you leap.
Make hay while the sun shines.

14.3 回家作業
利用緩衝區來讀寫資料

hw14_3_1.java

利用BufferedReader在文字模式下取得使用者輸入（可包括空白字元輸入），並重新顯示在主控台中。

(提示:可使用readLine() method)

```
/* output----------
輸入一列文字: This is a test!
您輸入的文字: This is a test!
----------*/
```

28

14.3 回家作業
利用緩衝區來讀寫資料

hw14_3_2.java

試依下列的步驟完成程式設計：

試產生1000 個1~99999 之間的整數亂數，再利用BufferedWriter 類別將它寫入"rand.txt"檔案內。

撰寫一程式讀取純文字檔rand.txt 的內容，並找出這1000 個數值的平均值、最大值與最小值。

撰寫一程式讀取rand.txt 的內容，並對這1000 個數值由小排到大，並將結果寫到rand2.txt。

```
/* output--------------
Maximum=99952,Minimum=58
Average=50584
---------------------*/
```

結果因亂數結果不同

29

14.4 使用InputStream與OutputStream類別

14-17

InputStream與OutputSteram類別可以
處理純文字檔及二進位檔(binary file)的資料。

14.4.1 讀取檔案的內容--使用FileInputStream類別

下表列出了FileInputStream類別的建構元與常用的method：

表14.4.1 FileInputStream的建構元

建構元	主要功能
FileInputStream (String name)	根據所給予的字串建立FileInputStream類別的物件

14-18

表14.4.2 FileInputStream類別的method

method	主要功能
int available()	**取得所讀取資料所佔的位元組數（bytes）**
void close()	關閉位元串流
long skip(long n)	在位元串流裡略過n個位元的資料
int read()	從輸出串流讀取一個位元組
int read(byte[] b)	從輸出串流讀取位元資料，並它存放到陣列b 中
int read(byte[] b, int off, int len)	從輸出串流讀取位元資料，並存放到指定的陣列中（off表示陣列索引值，len表示讀取位元數）

14-19

下面的範例示範如何使用**FileInputStream類別**：

```
01 // app14_5, 利用FileInputStream讀取現有的檔案
02 import java.io.*;
03 public class app14_5
04 {
05   public static void main(String args[]) throws IOException
06   {
07     FileInputStream fi=new FileInputStream("c:\\Java\\train.txt");
08     System.out.println("file size="+fi.available());//取得所讀取資料所佔的位元組數
09     byte ba[]=new byte[fi.available()];    // 建立byte陣列
10
11     fi.read(ba);  // 將讀取的內容寫到陣列ba裡
12     System.out.println(new String(ba));  // 印出陣列ba的內容
13     fi.close();
14   }
15 }
```

```
/* app14_5 OUTPUT---
file size=54
火車快飛，火車快飛
越過高山，飛過小溪
不知走了幾百里
----------*/
```

14.4.2 將資料寫入檔案--使用FileOutputStraem類別

14-20

下表列出了FileOutputStream類別的建構元與其常用的method：

建構元	主要功能
FileOutputStream(String filename)	依檔案名稱建立一個可供寫入資料的輸出串流物件，原先的檔案會被覆蓋
FileOutputStream(String name, Boolean a)	同上，但如果a設為true，則會將資料附加在原先的資料後面

method	主要功能
void close()	關閉位元串流
void write(byte[] b)	寫入位元陣列b到串流裡
void write(byte[] b, int off, int len)	寫入位元陣列b到串流裡（off表示陣列索引值，len表示寫入位元數）

app14_6示範了如何讀入一個gif圖檔，並將它另存新檔

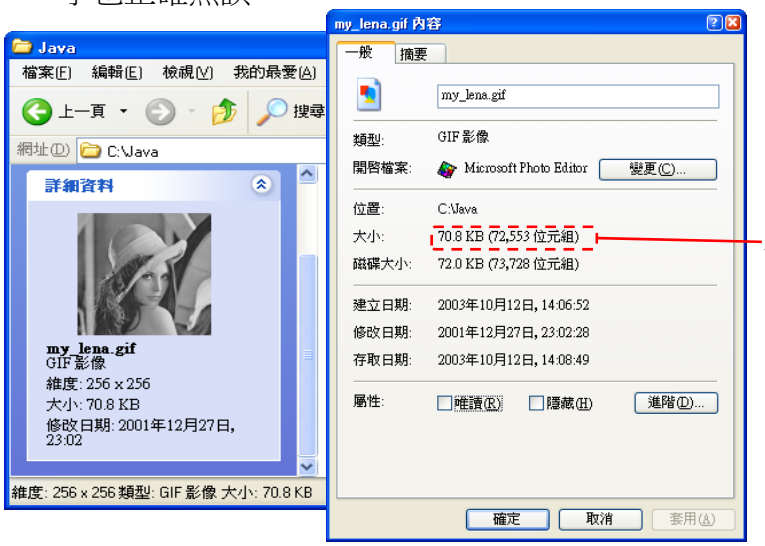
14-21

```
01  // app14_6, 讀入與寫入二進位檔案
02  import java.io.*;
03  public class app14_6
04  {
05    public static void main(String args[]) throws IOException
06    {
07      FileInputStream fi=new FileInputStream("c:\\Java\\lena.gif");//現有的檔案要讀出
08      FileOutputStream fo=new FileOutputStream("c:\\Java\\my_lena.gif");//新建檔案要寫入
09
10      System.out.println("file size="+fi.available()); // 印出檔案大小
11      byte data[]=new byte[fi.available()];  // 建立byte型態的陣列data
12
13      fi.read(data);              // 將圖檔讀入data陣列
14      fo.write(data);             // 將data陣列裡的資料寫入新檔my_lena.gif
15      System.out.println("file copied and renamed");
16      fi.close();
17      fo.close();
18    }
19  }
```

```
/* app14_6 OUTPUT-----
file size=72553
file copied and renamed
----------------------*/
```

14-22

從下面的畫面中，可以確定檔案已正確的寫入，且檔案大小也正確無誤：



ex14_4_1.java

- 將original.txt另存成final.txt，並將original檔案內容輸出到螢幕上。

```
/* ex14_4_1 OUTPUT-----
original.txt
A B C
----------------------------*/
```

14.4 回家作業
使用InputStream 與OutputStream 類別

hw14_4_1.java

請下載一圖檔，存至自己建立的資料夾中，命名為picture.bmp 圖檔，再請複製它成為picture_1.bmp，並且只複製前60k 位元組，然後使用ACDSee 看圖軟體觀察兩圖的差別。

```
/* output--------------
圖檔前61440位元組複製完畢
---------------------*/
```

37

14.4 回家作業
使用InputStream 與OutputStream 類別

hw14_4_2.java

將study.txt另存成study_new.txt，並將study.txt檔案內容輸出到螢幕上。

```
/* output--------------
我的姓名是王小明(請寫您的姓名)
一年來，我的學習心得是....(請直接於
study.txt中撰寫您的學習心得)
---------------------*/
```

38

檔案處理_總整理

- 以字元(character)為傳輸單位(讀text file)
  - FileReader fr=new FileReader("c:\\Java\\train.txt");//讀取已經存在的檔案
  - FileWriter fw=new FileWriter("c:\\Java\\proverb.txt");//建立一個新檔案
- 以位元組(bytes)為傳輸單位(讀Binary file)
  - FileInputStream fi=new FileInputStream("c:\\Java\\lena.gif");//現有的檔案要讀出
  - FileOutputStream fo=new FileOutputStream("c:\\Java\\my_lena.gif");//新建檔案要寫入
- Buffered(以緩衝區處理資料流的讀取和寫入增加處理效能)
  - FileReader fr=new FileReader ("c:\\Java\\random.txt"); //讀取檔案
  - BufferedReader bfr=new BufferedReader (fr);
  - FileInputStream fin = new FileInputStream(filename); //讀取檔案
  - BufferedReader reader = new BufferedReader(new InputStreamReader(fin));

InputStream(輸入流）用來讀取資料
OutputStream（輸出流）用來寫出資料

39

Java中使用BufferedReader讀取文件的疑惑

- IO的緩衝區的存在就是為了提高效率,把要操作的資料放進緩衝區,然後一次性把緩衝區的內容寫到目的地,而不是寫一次就往目的地寫一次
- 舉例來說，讓你去超市買餅乾，你覺得是一次讓你去買十袋效率高呢，還是讓你去十次、每次買一袋效率高？

如何比較不同存取方法的效率

```
package work;

import java.io.*;
public class buffer
{
   public static void main(String args[]) throws IOException
   {
    bufferread();
    fileread();
   }

   public static void bufferread() throws IOException
   {
   int len=-1;
      BufferedReader br=new BufferedReader(new FileReader("c:\\Java\\number.txt"));
      long start=System.currentTimeMillis();
      while((len=br.read())!=-1)  {
      }
      long end=System.currentTimeMillis();
      System.out.println("BufferRead時間="+(end-start)+"毫秒");
   }

   public static void fileread() throws IOException
   {
      int len=-1;
      FileReader br=new FileReader("c:\\Java\\number.txt");
      long start=System.currentTimeMillis();
      while((len=br.read())!=-1)  {
      }
      long end=System.currentTimeMillis();
      System.out.println("FileRead時間="+(end-start)+"毫秒");
   }
}
```

```
BufferRead時間=173毫秒
FileRead時間=262毫秒
```

ex14_performance.java

- 試撰寫一程式，可以比較不同讀取方法的存取績效。
- 並將不同的讀取方法分別撰寫在不同的class中，以建立物件的方式存取。
- 註解每一行程式碼的意義

```
BufferRead時間=…毫秒
FileRead時間=…毫秒
```