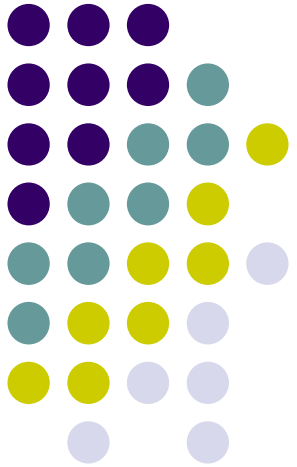
第二十二章
網路程式設計

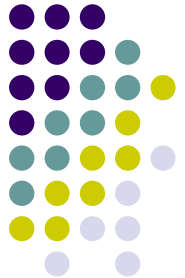


認識網路

學習如何取得文件的內容資訊

學習如何建立socket連線

學習建立TCP伺服程式與客戶程式



IP位址

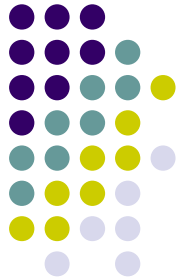
- IP位址
  - 以4個8 bits的數值，以10進位表示，用來區分網路上的電腦
  - 例如，「PChome Online網路家庭」放置Web主機的IP位址為

    210.59.230.60

    只要在瀏覽器內建入

    http://210.59.230.60

    即可進到「PChome Online網路家庭」的首頁



host name

- 網路是以IP位址來識別不同的電腦，它是唯一的
- hose name（主機名稱）
  - 習慣上會把電腦取一個簡單易記名稱，稱為host name
  - hose name可以沒有，也可以有好幾個
  - host name雖然好記，但電腦只認得IP位址，於是有DNS（Domain name service）伺服器的產生
- DNS伺服器可以將host name轉成相對應的IP位址
  - 要連上「PChome Online網路家庭」的首頁，只要鍵入：

```
http://pchome.com.tw
```

InetAddress類別

- InetAddress類別處理有關host name與IP位址的取得
- InetAddress類別並沒有提供建構元
- 下表列出InetAddress類別常用的method：

表 22.1.1 java.net.InetAddress 常用的 method

method	主要功能
static InetAddress[] getAllByName(String host)	給予電腦的 host name，取得該主機下所有提供服務的 IP 位址
static InetAddress getByName(String host)	給予電腦的 host name，取得該主機的 IP 位址
String getHostAddress()	取得電腦的 IP 位址
String getHostName()	取得電腦的 host name
static InetAddress getLocalHost()	取得本地端電腦的 host name 與 IP 位址

取得host name與IP位址 (1/2)

- 下面的程式碼是取得本機的名稱與IP位址之範例：

```
01  // app22_1, 取得本機的名稱與 IP 位址
02  import java.net.*;
03
04  public class app22_1
05  {
06     public static void main(String args[])
07     {
08        try
09        {
10           InetAddress adr=InetAddress.getLocalHost();
11           System.out.println(adr.getHostAddress());
12           System.out.println(adr.getHostName());
13           System.out.println(adr);
14        }
15        catch(UnknownHostException e) // 捕捉由 InetAddress()拋出的例外
16        {
17           System.out.println("無法取得 IP 位址");
18        }
19     }
20  }
```

```
/* app22_1 OUTPUT--------
169.254.180.217
buffalo
buffalo/169.254.180.217
------------------------*/
```

getLocalHost會拋出UnknownHostException例外

取得host name與IP位址 (2/2)

- getByName() method也可取得相對應的IP位址，如：

```
01  // app22_2, 取得本機的名稱與 IP 位址
02  import java.net.*;
03
04  public class app22_2
05  {
06     public static void main(String args[])
07     {
08        try
09        {
10           InetAddress adr;      // 宣告 InetAddress 類別型態的變數 adr
11           adr=InetAddress.getByName("udn.com");    // 取得 IP 位址
12           System.out.println(adr);
13        }
14        catch(UnknownHostException e)
15        {
16           System.out.println("無法取得 IP 位址");
17        }
18     }
19  }
```

```
/* app22_2 OUTPUT------
udn.com/210.244.31.140
----------------------*/
```

URL的介紹

- URL是universal resource locator，表示網路資源的位址
  - 下面的網址便是一個URL的範例：

```
http://udn.com:80/NEWS/main.html
```

  - 「http」代表取得資源的方式是利用http的通訊協定（protocol）
    - 通訊協定，可以想像成是電腦與電腦之間溝通的語言
    - 例如，TCP/IP就是電腦連上網路的通訊協定
  - 「udn.com」是伺服器的名稱，「80」是埠號

URL類別的建構元與method

- 下表列出URL類別常見的建構元與method：

表 22.2.1　java.net.URL 常用的建構元與 method

建構元	主要功能
URL(String spec)	由字串 spec 建立 URL 物件
URL(String protocol, String host, String file)	以通訊協定、host name 與檔案路徑的字串建立 URL 物件
URL(URL context, String spec)	以一個絕對路徑的 URL 物件，以及相對路徑的檔案名稱建立 URL 物件

method	主要功能
Object getContent()	取得 URL 檔案的內容
String getFile()	取得 URL 的檔案名稱
String getHost()	取得 URL 的 host name
String getPath()	取得 URL 的路徑
int getPort()	取得 URL 的埠號
String getProtocol()	取得 URL 的通訊協定名稱
URLConnection openConnection()	建立一個 URL 連線，並傳回 URLConnection 物件

建立URL物件

- 下面的範例建立一個URL物件，並取得物件裡的資訊：

```
01  // app22_3, 使用 URL 類別
02  import java.net.*;
03
04  public class app22_3
05  {
06     public static void main(String args[])
07     {
08        try
09        {
10           URL u=new URL("http://udn.com/NEWS/main.html");
11
12           System.out.println("通訊協定名稱為 "+u.getProtocol());
13           System.out.println("host name 為 "+u.getHost());
14           System.out.println("埠號為 "+u.getPort());
15           System.out.println("檔名為 "+u.getFile());
16        }
17        catch(MalformedURLException e)
18        {
19           System.out.println("發生了" +e+ "例外");
20        }
21     }
22  }
```

```
/* app22_3 OUTPUT-------
通訊協定名稱為 http
host name 為 udn.com
埠號為 -1
檔名為 /NEWS/main.html
----------------------*/
```

載入URL的檔案內容

本例用getContent() method
取得URL內的資源

```
01  // app22_4, 載入 URL 的檔案內容
02  import java.net.*;
03  import java.io.*;
04
05  public class app22_4
06  {
07     public static void main(String args[])
08     {
09        String str;
10
11        try
12        {
13           URL u=new URL("file:/c:\\java\\poem.txt");
14
15           Object obj=u.getContent();                    // 取得 URL 的內容
16           InputStreamReader isr=new InputStreamReader((InputStream) obj);
17           BufferedReader br=new BufferedReader(isr);
18
19           while((str=br.readLine())!=null)
20              System.out.println(str);
21           br.close();
22        }
23        catch(IOException e)
24        {
25           System.out.println("發生了"+e+"例外");
26        }
27     }
28  }
```

如果URL路徑是設在網路上，在執行本程式前一定要先連上網路，否則會發生UnknownHostException例外

```
/* app22_4 OUTPUT---

床前明月光，
疑是地上霜。
舉頭望明月，
低頭思故鄉。
------------------*/
```

URLConnection類別的method

- 透過URLConnection類別可以取得
  - 檔案的大小
  - 類型
- 下表列出URLConnection類別提供的method：

表 22.2.2　java.net.URLConnection 常用 method

method	主要功能
int getContentLength()	取得資料所佔的位元數
int getContentType()	取得資料的型態

使用URLConnection類別

本例用getContentLength()取得檔案的大小

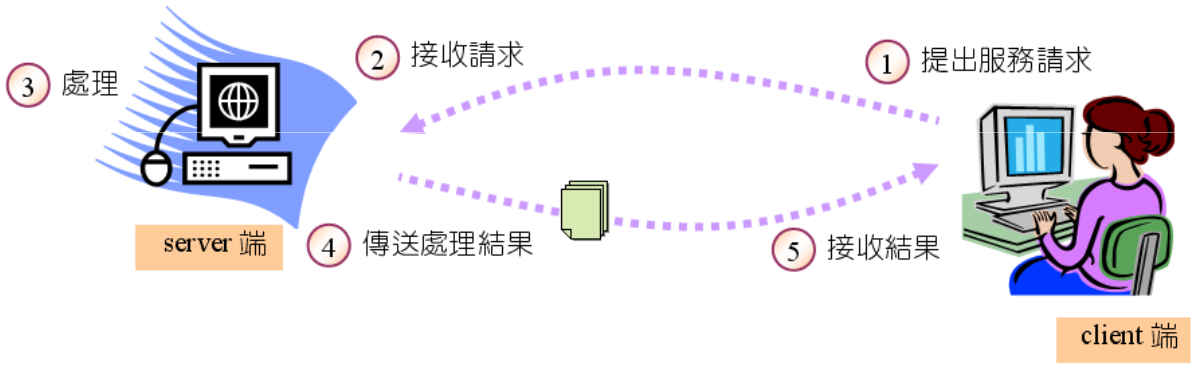
```
01  // app22_5, 使用 URLConnection 類別
02  import java.net.*;
03  import java.io.*;
04
05  public class app22_5
06  {
07     public static void main(String args[])
08     {
09        try
10        {
11           URL u1=new URL("http://udn.com");
12           URL u2=new URL("file:/c:\\java\\poem.txt");
13           URL u3=new URL("file:/c:\\java\\pic0.jpg");
14
15           URLConnection uc1=u1.openConnection();
16           URLConnection uc2=u2.openConnection();
17           URLConnection uc3=u3.openConnection();
18
19           System.out.print("主網頁的大小為 " + uc1.getContentLength());
20           System.out.println(", 類型為 " + uc1.getContentType());
21           System.out.print("poem.txt的大小為 "+uc2.getContentLength());
22           System.out.println(", 類型為 " + uc2.getContentType());
23           System.out.print("pic0.jpg的大小為 " + uc3.getContentLength());
24           System.out.println(", 類型為 " + uc3.getContentType());
25        }
26        catch(IOException e)
27        {
28           System.out.println("發生了"+e+"例外");
29        }
30     }
31  }
```

```
/* app22_5 OUTPUT-----------------------

主網頁的大小為 50406,  類型為 text/html
poem.txt 的大小為 54,  類型為 text/plain
pic0.jpg 的大小為 6280,  類型為 image/jpeg
---------------------------------------*/
```

主從架構的認識

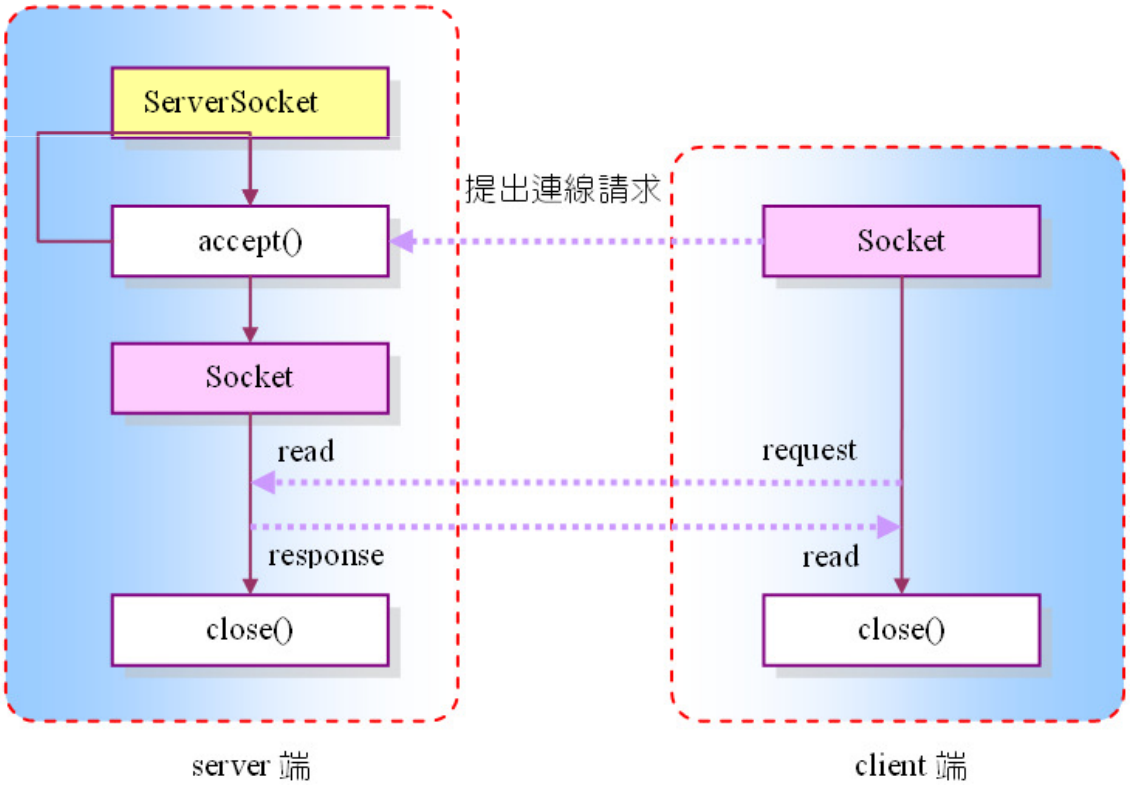
- 主從架構裡，
  - 不同的伺服程式會使用不同的埠號，同時在伺服器裡執行
  - 當client端（客戶端）的請求送來時，server端（伺服端）的伺服程式會做出回應：



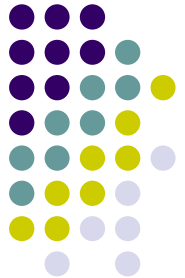
  - 收到client端的請求後，server端的伺服程式會建立新的socket物件，透過這個物件與客戶端連線
  - socket是程式與網路之間的一種介面
  - ServerSocket與Socket類別，可將客戶端的電腦連上伺服器，並從伺服器傳遞資料給客戶端

主從架構運作的流程

- client-server的運作的流程如下圖所示：



ServerSocket類別的建構元與method



- 下面列出ServerSocket類別提供的建構元與method：

表 22.3.1　java.net.ServerSocket 常用的建構元與 method

建構元	主要功能
ServerSocket(int port)	以埠號 port 建立 server socket 連線

method	主要功能
Socket accept()	監控客戶端的請求。當客戶端有請求時，便建立 socket 物件與客戶端連繫
void close()	關閉 socket

Socket類別的建構元與method

- 下面列出Socket類別提供的建構元與method：

表 22.3.2 java.net.Socket 常用的建構元與 method

建構元	主要功能
Socket()	建立 socket 物件
Socket(InetAddress address, int port)	根據 IP 位址與埠號建立 socket 物件

method	主要功能
void close()	關閉 socket 連線
InetAddress getInetAddress()	取得 socket 所連線的 IP 位址
InetAddress getLocalAddress()	取得 socket 連線的本地端 IP 位址
InputStream getInputStream()	取得 socket 連線的輸入串流
OutputStream getOutputStream()	取得 socket 連線的輸出串流
int getLocalPort()	取得 socket 連線的本地端埠號
int getPort()	取得 socket 連線時遠端的埠號

建立Server端的伺服程式

本範例將建立一個server端的伺服程式

```
01  // app22_6, 建立 Server 端的伺服程式
02  import java.net.*;
03  import java.io.*;
04
05  public class app22_6
06  {
07     public static void main(String args[])
08     {
09        try
10        {
11           ServerSocket svs=new ServerSocket(2525);
12
13           System.out.println("等候客戶端的請求中...");
14           Socket s=svs.accept();   // 等候客戶端的請求
15           System.out.println("客戶端已和本機連上線...");
16
17           OutputStream out=s.getOutputStream();   // 取得輸出串流
18           String str="Honor shows the man.";
19           System.out.println("資料正在傳送中...");
20           out.write(str.getBytes()); // 將字串轉成 Byte 陣列，再寫入串流中
21           out.close();               // 關閉輸出串流
22           s.close();                 // 關閉 socket
23           System.out.println("資料傳送完畢...");
24        }
25        catch(Exception e)
26        {
27           System.out.println("發生了"+e+"例外");
28        }
29     }
30  }
```

由於尚未撰寫客戶端的程式碼，因此本範例的執行結果只會在螢幕中出現 "等候客戶端的請求中..." 字串

埠號共有0~65535可供使用，其中埠號0~1023為系統所保留，可讓使用者自行設定的埠號只有1024~65535

建立Client端的伺服程式

```
01  // app22_7, 建立 Client 端的伺服程式
02  import java.net.*;
03  import java.io.*;
04
05  public class app22_7
06  {
07     public static void main(String args[])
08     {
09        byte buff[]=new byte[1024];              // 建立 byte 型態的陣列
10        try
11        {
12           System.out.println("正在與伺服器建立連線...");
13           Socket s=new Socket("127.0.0.1",2525);   // 建立 socket 物件
14           System.out.println("已經與伺服器取得連線...");
15           InputStream in=s.getInputStream();       // 建立輸入串流
16           int n=in.read(buff);                     // 從串流讀入資料
17           System.out.print("從伺服端收到: ");
18           System.out.println (new String(buff,0,n));// 印出讀入的內容
19           in.close();
20           s.close();
21        }
22        catch(Exception e)
23        {
24           System.out.println("發生了"+e+"例外");
25        }
26     }
27  }
```

客戶端的程式，可對伺服器端的程式送出請求

必須先執行伺服器端的程式，然後再執行客戶端的程式

主從架構範例的執行結果

- app22_6與app22_7的執行結果如下所示：

伺服器端的程式

```
/* app22_6 OUTPUT ----
等候客戶端的請求中...
客戶端已和本機連上線...
資料正在傳送中...
資料傳送完畢...
----------------------*/
```

客戶端的程式

```
/* app22_7 OUTPUT----
正在與伺服器建立連線...
已經與伺服器取得連線...
從伺服端收到: Honor shows the man.
-------------------------------*/
```

必須先執行伺服器端的程式，然後再執行客戶端的程式

-The End-