Java Steps

2011, 董少桓、林彥宏

January 09, 2012

CONTENTS

CHAPTER

ONE

簡介

1.1 Java 程式語言的特色

追溯至 1990 年 12 月，Sun (昇陽電腦) 公司成立 Green Team 團隊，主要成員有 Patrick Naughton、Mike Sheridan 及 James Gosling。他們開始著手一項新專案「Green Project」，目標是發展一種系統架構，使程式能夠在電腦以外的消費性電子產品平台運作（例如手機、資訊家電等)。[1]

Green Team 在 1992 年 9 月，發表一款名為 Star Seven 的機器，和後來市面上的 PDA 裝置類似，Star 7 在當時已具有先進的無線通訊功能。Java 程式語言的前身 Oak 在此時誕生，用來開發 Star 7 的軟體程式；當時 James Gosling 因為看見窗外的「橡樹 (oak)」，決定將新程式語言命名為 Oak 。

當時 Oak 要註冊商標時，發現這個名字已經被別家公司先用。由於工程師們喜歡在討論時喝杯咖啡，就將程式語言名稱改為 Java（一種咖啡的名稱)，這個名稱就一直沿用到現在。

Java 有別於許多傳統程式語言:

- 傳統的程式語言在編譯後會產生 machince code（機器碼)，然後直接在硬體上執行;
- Java 在編譯後則會產生 Byte Code，並間接的在 Java Virtual Machine （JVM) 上執行。這個 JVM （Java 虛擬機器）其實是一個軟體，其功用是解譯並執行 Byte Code，而 JVM 仍然是在硬體上執行。

因為 JVM 是軟體，所以 Java 程式具有跨平台的特性：只要為不同的處理器或作業系統設計其專屬的 JVM，Java 程式便可以不需改寫，就能在這些不同的處理器或作業系統上執行。這便是「Write once, runs everywhere （一次編譯、到處執行)」的由來。

[1] http://en.wikibooks.org/wiki/Java_Programming/History

Java 也支援物件導向程式設計（Object-Oriented Programming），所謂「物件」，簡單的說有「屬性」也有「方法」，例如冷氣機的「屬性」可以包括:「開關」及「溫度」;而「方法」則可以包括:「開機」、「關機」及「設定溫度」等。為了讓程式設計師，可以比較容易的使用物件撰寫模擬、控制與應用電腦本身（如滑鼠與鍵盤等也是物件）和我們生活周遭的物件的程式，因此便有研究人員發明了支援物件導向程式設計的語言。

Java 的設計搭上了全球資訊網的順風車，原因是 Java 的設計團隊可以寫一個能夠在瀏覽器中執行的 JVM，而讓 Java Applet 程式可以透過網路下載至瀏覽器中執行。這個「網路」＋「物件導向」的特性讓 Java 瞬間爆紅。

除了跨平台、物件導向、可透過網路動態的載入及執行程式等功能之外，Java 還支援多執行緒、例外狀態處理與自動記憶體回收的功能:

- 多執行緒讓一個程式可以執行數個工作;
- 例外狀態處理讓處理例外的程式碼也能夠物件化;
- 自動記憶體回收則讓程式設計師免除了使用低階的指標（pointers）來設計資料結構及管理記憶體的負擔。這個特色成了 C 與 C++ 語言程式設計師的福音，因為它可以為程式設計師減少許多不容易 debug 的錯誤。

1.2 安裝 JDK

在編譯與執行 Java 程式前，你的電腦必須先安裝 JDK（Java Development Kit）:

- 下載新版的 JDK http://java.sun.com/javase/downloads/index.jsp

在安裝完成後，也需要完成 PATH 及 CLASSPATH 的設定:

- `PATH=C:\Program Files\Java\jdk1.6.0\bin;...`
- `CLASSPATH=.;C:\Program Files\Java\jdk1.6.0\lib;...`

請注意: 以上路徑中的 jdk1.6.0 會因版本的不同而異。此外，在設定 classpath 時要特別注意在＝號的右邊要輸入這個 「.」 。這個點的意義是目前的目錄（current directory），也是執行 Java 程式時用來搜尋執行檔的目錄。

1.3 編譯及執行 Java 程式

有兩種方式可以編譯及執行一個 Java 程式。第一種是使用程式開發環境（program development environment），例如: Eclipse ; 另一種則是使用一般的程式編輯器。以

下是使用「記事本」寫 Java 程式時所需要進行的三個步驟:

1. 使用「記事本」輸入以下的程式並將檔案命名為 EnglishExam.java （注意: 附檔名必須是.java 而不是.txt， **而這個檔案的主檔名必須與 public class 後面的 EnglishExam 相同**):

```
public class EnglishExam {
  public static void main(String argv[]) {
    System.out.println(”Your score is 97.”);
  }
}
```

2. 執行「命令提示字元」並將目錄切換至儲存 EnglishExam.java 的目錄，然後執行:

   ```
   javac EnglishExam.java
   ```

   執行這個 javac 指令就是執行 Java 的編譯器（compiler)，其結果是在同樣的目錄產生一個 EnglishExam.class 的 byte code 檔。

3. 上面的步驟如果有編譯錯誤則繼續修改程式。如果沒有編譯錯誤則可以執行:

   ```
   java EnglishExam
   ```

   執行後 Your score is 97。便會顯示在螢幕上。

在開發 Java 程式的過程中，有可能發生編譯錯誤（compile-time error)。這時便需要再次的使用編輯器修改錯誤，直到沒有任何的編譯錯誤為止。編譯完畢之後，在程式執行時也有可能發生 run-time error。同樣的，這時也需要使用編輯器修改、編譯、執行、除錯，直到沒有錯誤為止。

一般的 Java 程式都是由一或多個類別（class）所組成， **其中的一個類別至少要有一個命名為 public static void main 的方法（method)，而這個程式就是由 main 開始執行** 。（透過網路瀏覽器執行的 Java applet 不適用此規則。)

上例中的 public class EnglishExam 是指定 EnglishExam 這個類別是 public 是公用的，也就是可以被程式中其他的類別引用。而 public static void main(String argv[]) 的意義是:

1. public: 指定 main 為一個可以被其他類別使用的 public method;
2. static: 指定 main 為一個類別方法（static method)，一個類別方法隸屬於一個 class;
3. void: 代表 main 執行完畢後回傳的型態，因為 main 沒有回傳任何數值，因此它的回傳型態是 void;
4. String argv[]: 指這個方法的輸入參數是 argv[] 而 String 則是它的型態。main 的輸入參數 String argv[] 可以在執行一個 Java 程式時將字串（String）資料輸入這

個程式。例如在編譯以下的程式之後:

```
public class HelloJava {
        public static void main(String argv[]) {
                System.out.println("Hello " + argv[0] + argv[1]);
        }
}
```

以「命令提示字元」執行:

```
java HelloJava Basic C++
```

便會呼叫 System.out.println 並輸出:

```
Hello Basic C++
```

這個程式的 argv[] 代表 argv 這個變數是一個陣列，而 argv[0]、argv[1] 則取用 argv 內第 0、1 個儲存格的內容。

Java 程式中 **用大刮號 { } 標示的 Block（區塊）** 是用來組織程式層次關係的語法。

例如上例的程式就有兩個區塊，一組用來標示 class 的區塊，另一組則用來標示 main 的區域。區塊中可以包含其他的區塊，在撰寫程式時也應注意要把區塊的內容往右縮排。一組用來標示類別的區塊內，可以有數個變數與方法。而一組用來標示方法的區塊內可以有一或多句以「;」結束的程式碼。這些程式碼共同構成了這個方法的 body。

為 Java 程式中使用的名字命名，有一個不成文的規定： **類別名稱的第一個字母要用大寫** 。 **方法或變數的第一個字母則是小寫** ，若有數個字合併時則 **後續的字的第一個字母也習慣用大寫** 。

動手練習：修正程式碼的錯誤