

第一章 認識 Java

本章內容

- 1-1 Java 簡介
- 1-2 Java 的版本與執行環境
- 1-3 設定 Java 的執行環境
- 1-4 測試 Java 的執行環境
- 1-5 體驗 Java 的程式
- 1-6 Java 程式的架構
- 1-6 Java 程式的架構

1-1 Java 簡介

- Java 是美國昇陽公司 (Sun Microsystems) 所開發的程式語言。在 1991 年時，昇陽公司成立了一個稱為 (Green Project) 的研究計劃，主要的目的是開發消費性電子產品的控制軟體，而由於當時所使用的「C++」程式語言過於複雜且缺乏安全性，所以，當時的計劃主持人 James Gosling 便以「C++」為基礎，重新開發一套新的程式語言，名稱為「Oak」，它便是 Java 的前身。

1-1 Java 簡介

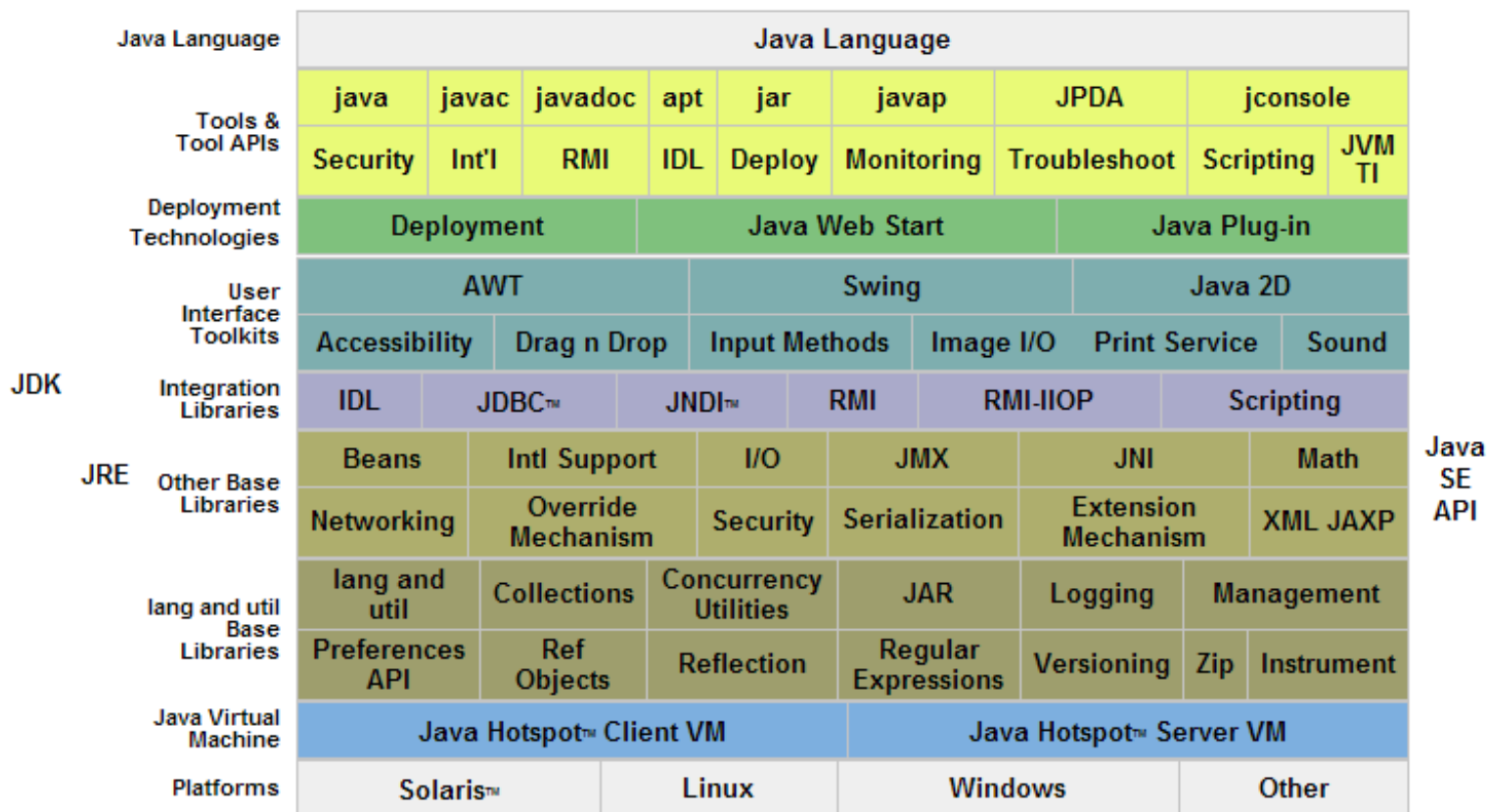
- Java 語言和 C++ 有關，而 C++ 又來自於 C 語言。其實，Java 語言許多的特性是繼承自這兩種程式語言，依照昇陽官方網站 (<http://java.sun.com/docs/white/>) 的白皮書中，Java 語言的特色有以下幾點：
 - 簡單 (Simple) 、物件導向 (Object-Oriented) 、網路功能 (Network-Savvy) 、強韌 (Robust) 、安全性 (Secure) 、跨平台 (Architecture Neutral) 、程式直譯 (Interpreted) 、高效率 (high-performance) 、多執行緒 (MultiThread)

1-2 Java 的版本與執行環境

- Java 自從發表後，不同的版本都能增加更多實用的功能。最值得注意是 1999 年發表的 Java 1.2 版，這也是較多人所熟悉的「Java 2」，此時，Java 區分為四個不同的版本：
 - **J2EE (企業版, Enterprise Edition)**：主要提供企業伺服器端應用程式的開發需求。
 - **J2SE (標準版, Standard Edition)**：主要提供桌上型電腦應用程式的開發需求。
 - **J2ME (精簡版, Micro Edition)**：主要是提供資訊家電應用程式的開發需求。
 - **Java Card**：主要是提供 smart card 市場的需求。
- 2004 年所發表的 J2SE 1.5 版（正式版時改編號為 J2SE 5.0，代號 Tiger）也是值得注意的版本，除了內含功能的增多之外，語法也同時更新。
- 目前，Java 最新的版本為 1.6 版，版本仍在持續的更新中。在名稱上，也同時做了變更。原本的 J2SE、J2ME、J2EE 等名詞將成為過去式，在新的版本中，將以「Java SE(標準版)」、「Java ME(精簡版)」及「Java EE(企業版)」來

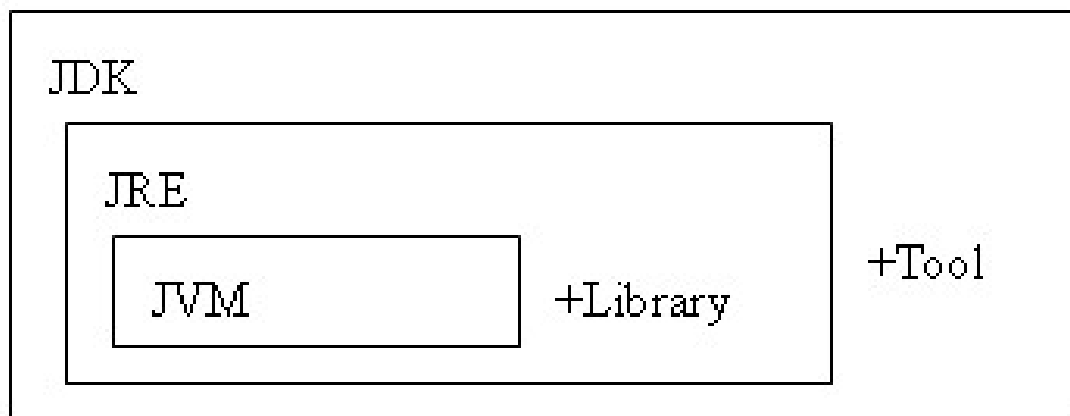
1-2-1 Java SE 的平台技術

- 我們先了解較新的 Java SE Platform 的主要架構。下圖是來自 Java 官方網站 (<http://java.sun.com/javase/6/docs/index.html>) 所公開的 Java SE 的架構圖：



1-2-2 了解 Java 的執行環境

- 在下載 Java 相關檔案時，您要先知道一些相關的名詞：
 - **JVM (Java Virtual Machine)**：執行 Java 程式。JVM 是以軟體模擬的方式，在真實的機器上虛構出來的執行環境。
 - **JRE (Java Runtime Environment)**：Java 程式的執行環境，內含類別函式庫。
 - **JDK (Java Software Development Kit)**：也可稱為 SDK，提供開發 Java 應用程式的一些工具。
- 如果以圖示來表示，三者的關係可以是：



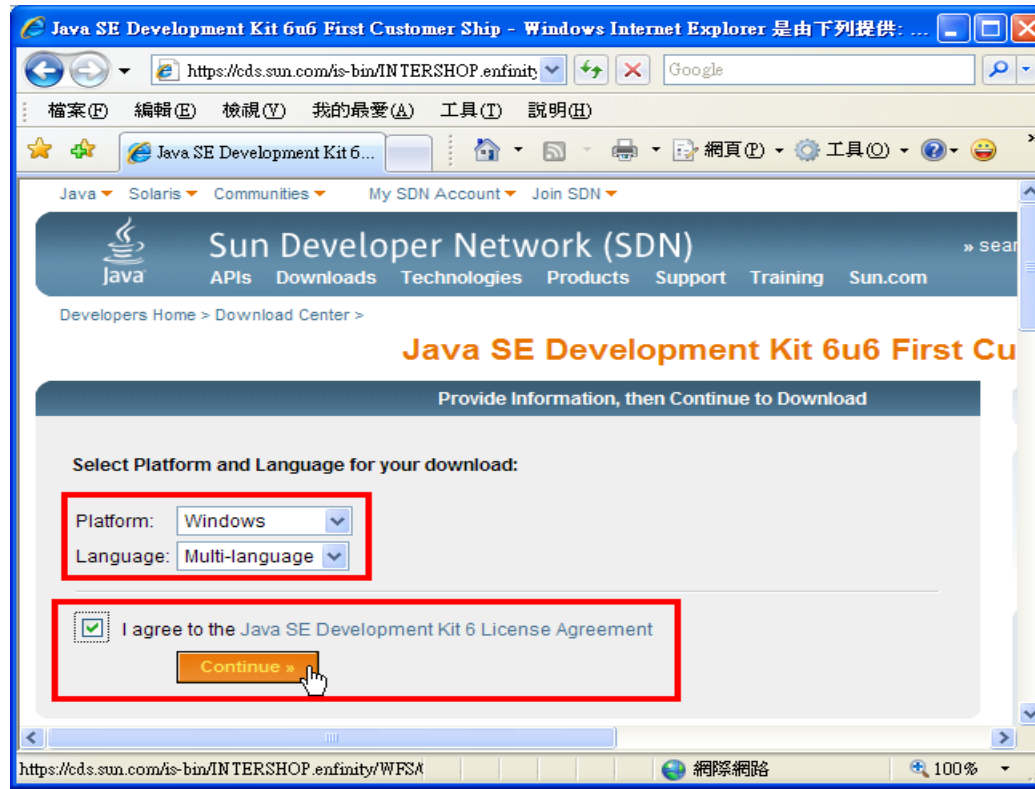
1-2-3 下載 Java 執行環境所需檔案

- 要開發 Java 程式前，我們要先下載 Java SE 6，並安裝相關的檔案。
- 我們可以由 Java 的官方網站免費取得 Java SE。請連上「<http://java.sun.com/>」網站，並在該頁面中找到相關的下載連結。



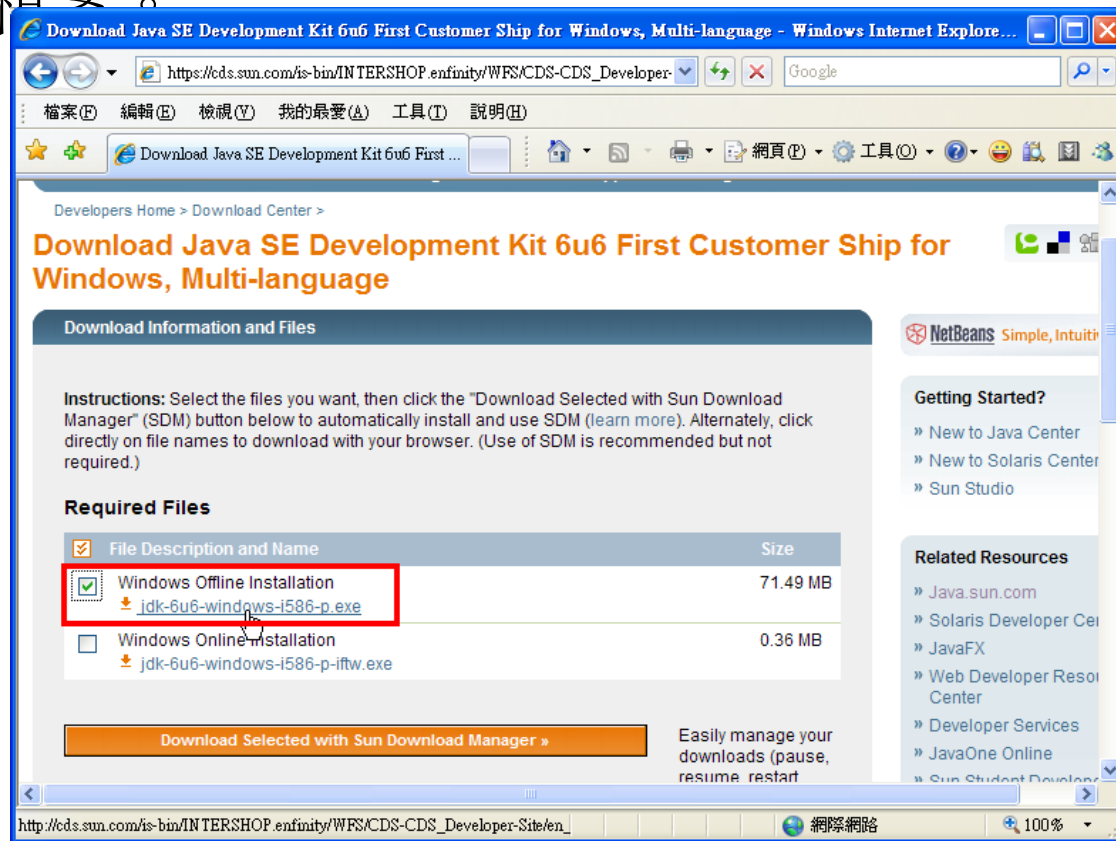
1-2-3 下載 Java 執行環境所需檔案

- 請按「**Java SE**」這個連結，更換頁面後，再按「**JDK 6 Update 6**」旁的「**Download**」連結按鈕。
- 選擇合適的作業平台，勾選「**I agree to the Java SE Development Kit 6 License Agreement**」旁的核取方塊，再按「**Continue>>**」按鈕。



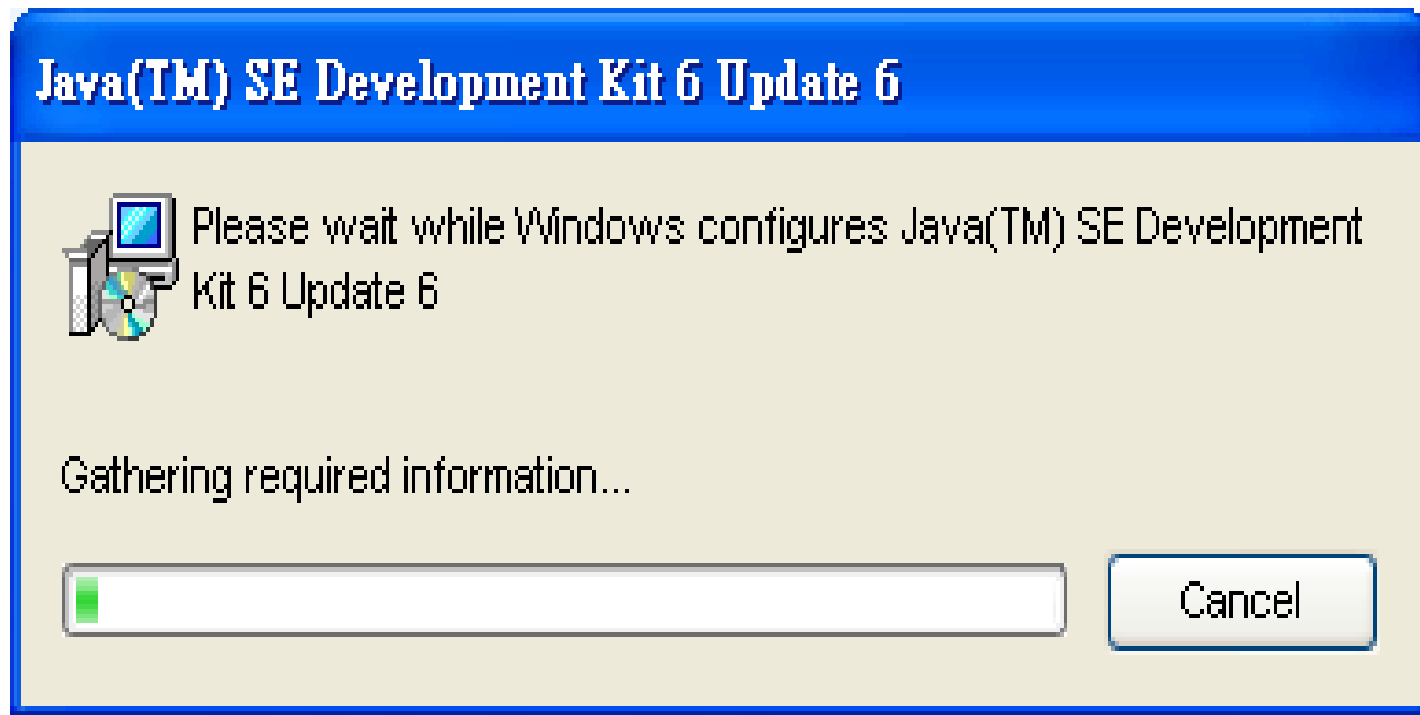
1-2-3 下載 Java 執行環境所需檔案

- 選擇想下載的選項，再按檔案名稱的連結下載，例如：
「**jdk-6u6-windows-i586-p.exe**」。您可能
會因為連上該網站的時間的不同，而取得不同的版本
編號的檔案。



1-2-4 安裝 Java SE

- 請執行下載的「**jdk-6u6-windows-i586-p.exe**」檔案，進入安裝程序。

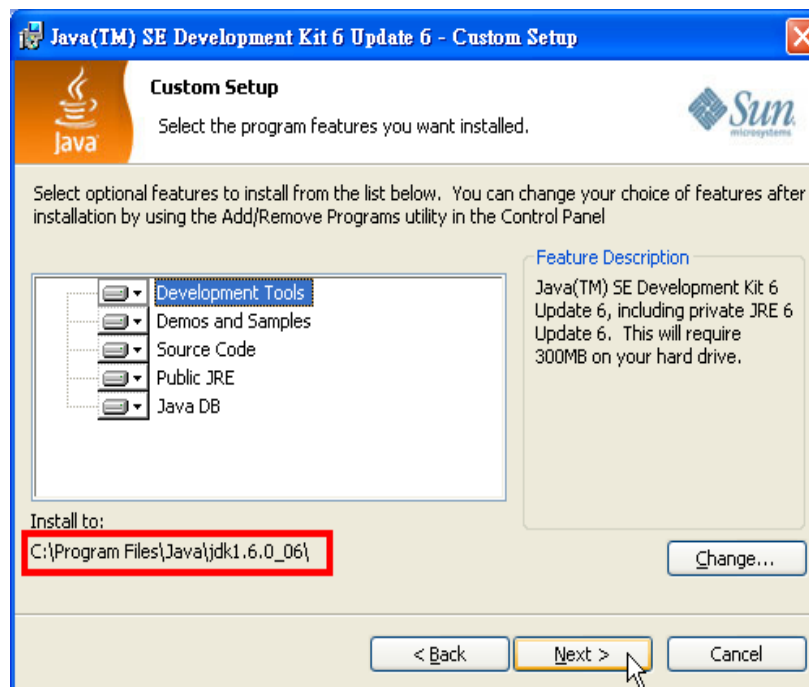


1-2-4 安裝 Java SE

- 進入安裝選項的畫面後，共有五個選項。
 - **Development Tools** 是最主要的選項，包含了 JRE 以及相關的開發工具，您必需安裝這個選項。
 - **Demos and Samples** 包含了一些範例程式及相關的原始碼，您可以在日後的學習過程中，當作參考範本。
 - 「**Source Code**」是 API 類別庫中的原始程式碼，您可以從各類別的原始碼中，學習到如何撰寫漂亮的 Java 程式。
 - 「**Public JRE**」也包含了執行環境，並幫您在瀏覽器中安裝 Java 的 Plug-In。
 - 「**Java DB**」包含了可供使用的資料庫。

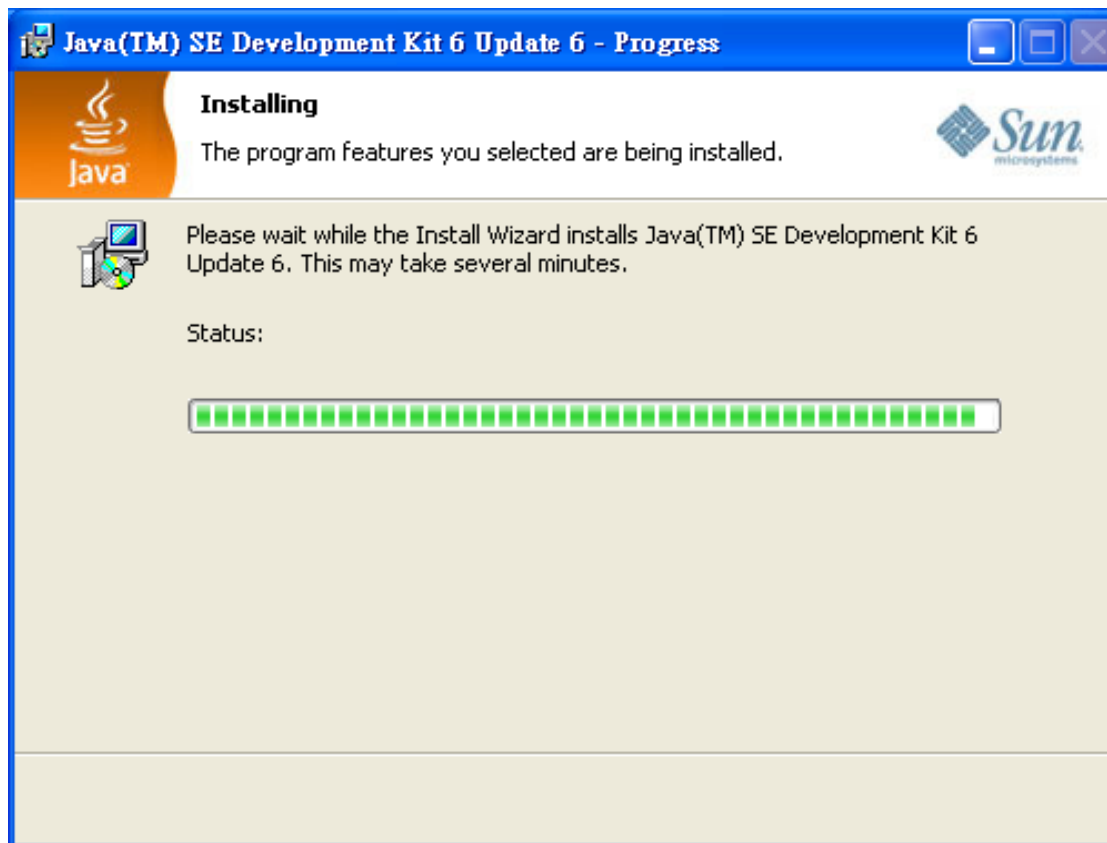
1-2-4 安裝 Java SE

- 請特別注意到這些項目的安裝路徑，預設的路徑是在「**C:\Program Files\Java**」之下，之後的資料夾名稱則會隨版本的不同而有所變更。記得這個路徑，往後我們要設定 Java 執行環境時，會使用到這個路徑的完整名稱。或者，您也可以按「**Change**」按鈕，改變安裝的路徑。
- 設定好您所需要安裝的項目後，按下「**Next>**」按鈕，繼續安裝 ...



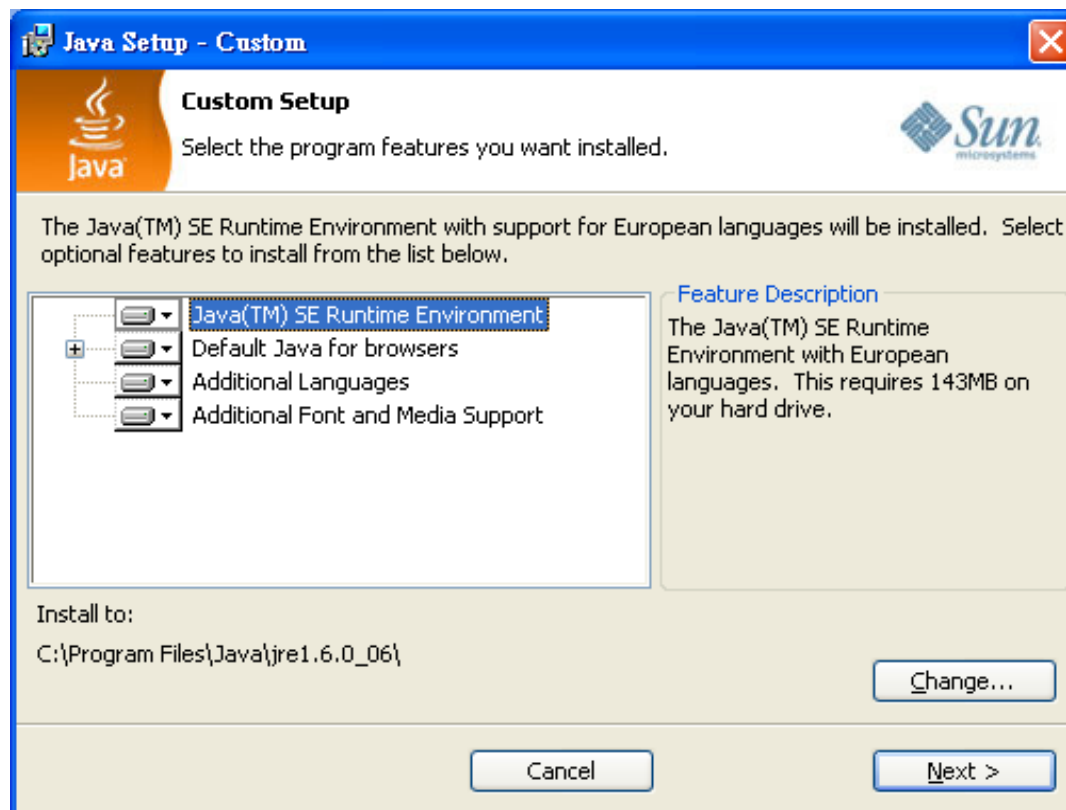
1-2-4 安裝 Java SE

- 進行安裝動作的畫面 ...



1-2-4 安裝 Java SE

- 安裝完主要的選項後，安裝程式會再出現另一個選項畫面，讓您可以選擇並安裝相關的 JRE 套件。請依照步驟三的方式，選取相關的套件，或是直接按下「**Next>**」按鈕，繼續安裝 ...



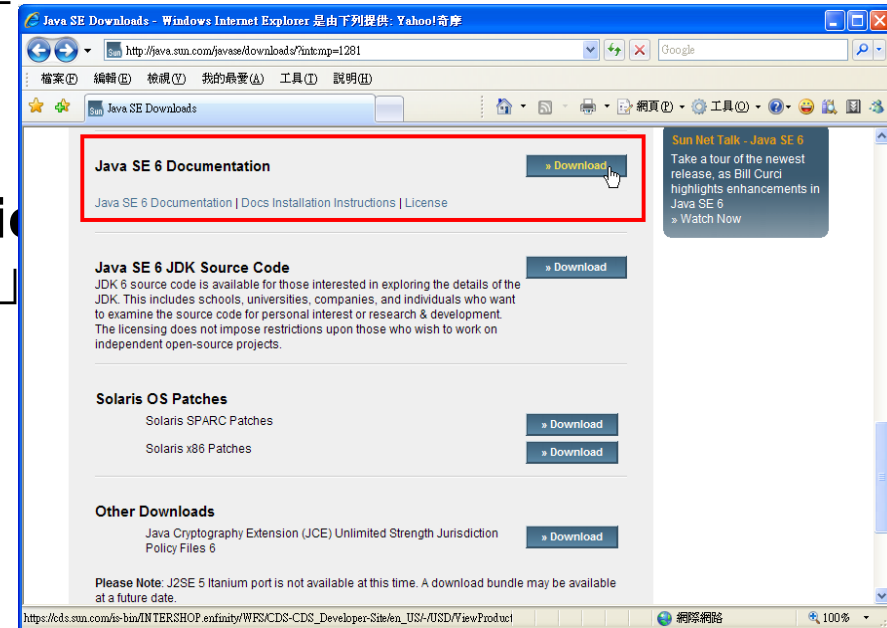
1-2-4 安裝 Java SE

- 當您安裝好所有的項目後，請按下「 **Finish** 」按鈕，完成安裝。



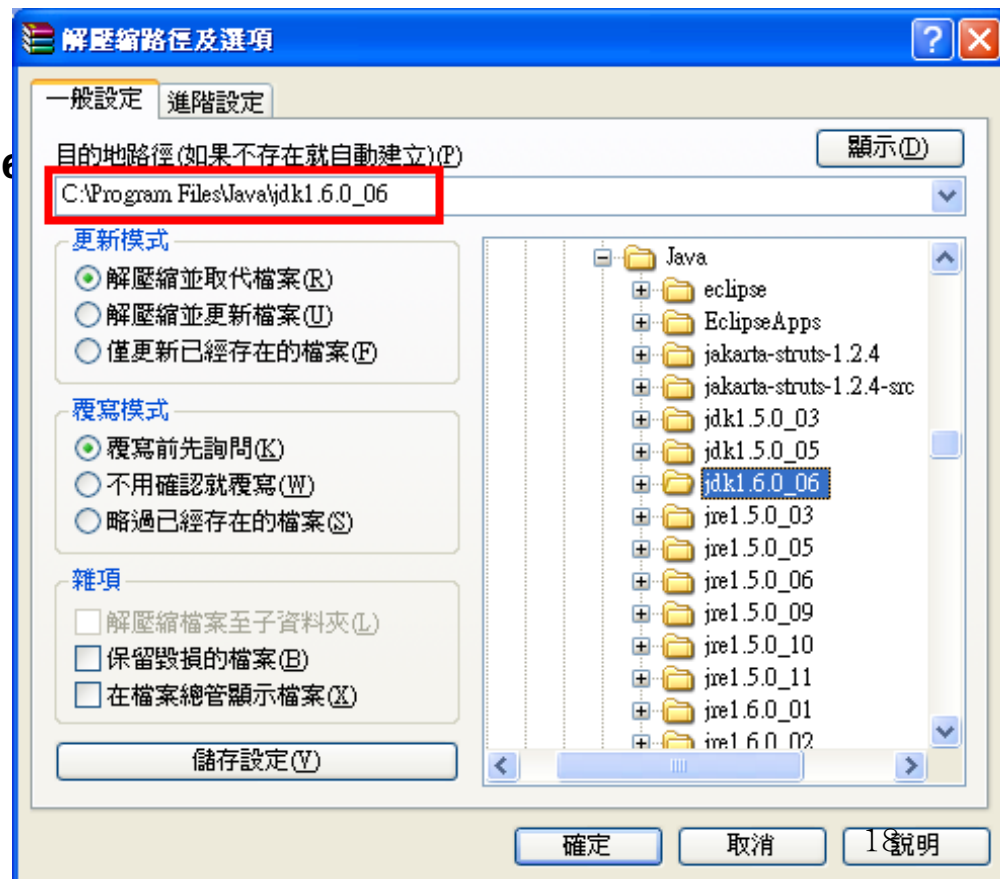
1-2-5 安裝說明文件

- 我們下載的 Java SE 6 中並未包含 Java 的說明文件。該文件包含了最完整的 Java 的類別庫，及其相關的屬性、方法的說明，在我們寫作 Java 程式時，一定會不斷的參考該說明文件，因為，沒有任何一本書的內容會比得上該文件完整。
- 安裝說明文件前，我們還是要由 Java 的網路中下載說明文件檔。在下載 Java SE 6 檔案的「**步驟二**」畫面中，我們可以找到相關的連結。找到「**Java SE 6 Documentation**」選項，並按「**>>Download**」連結後，參考下載 Java SE 6 的步驟，下載檔案。



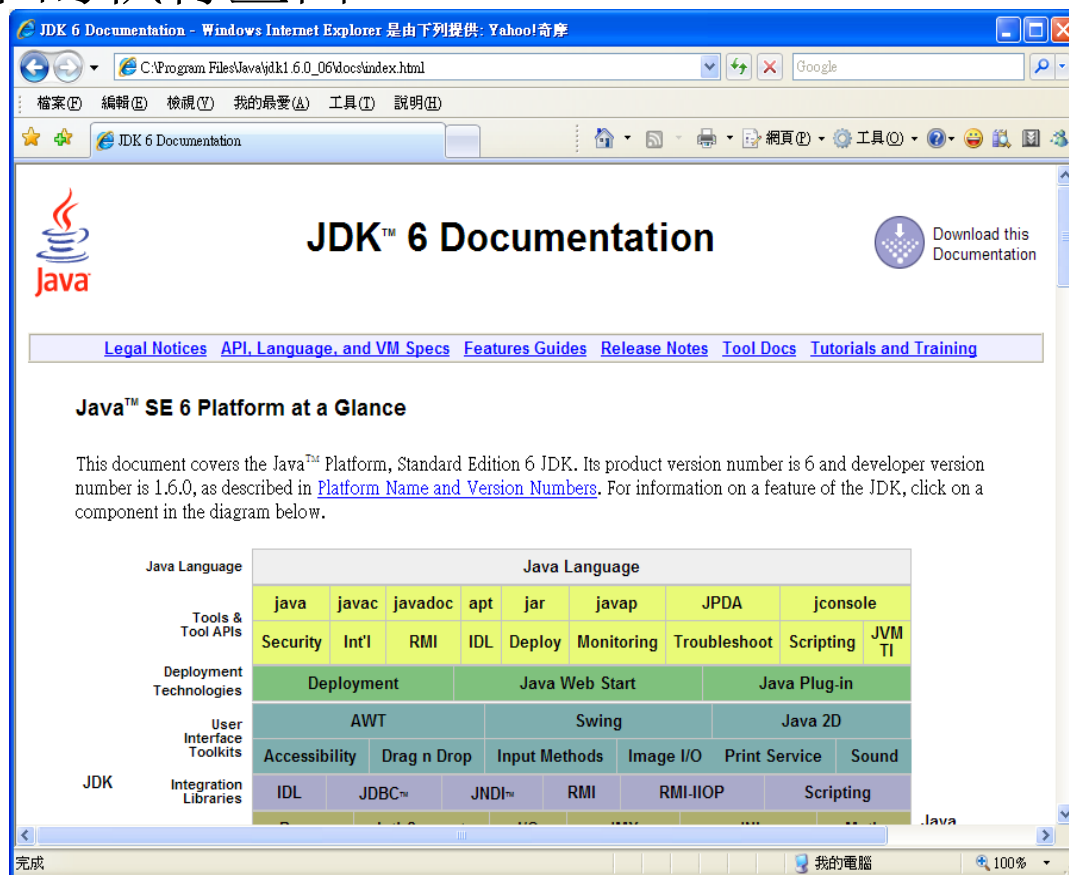
1-2-5 安裝說明文件

- 下載後的檔名應該類似「**jdk-6-doc.zip**」，這是一個「**ZIP**」壓縮檔，並不需要安裝，直接解開後就可以使用了。
- 建議您將該程式解壓縮至 Java 的相關路徑下，例如：
「**C:\Program Files\Java\jdk1.6.0_06**」
這個 JDK 的安裝位置。



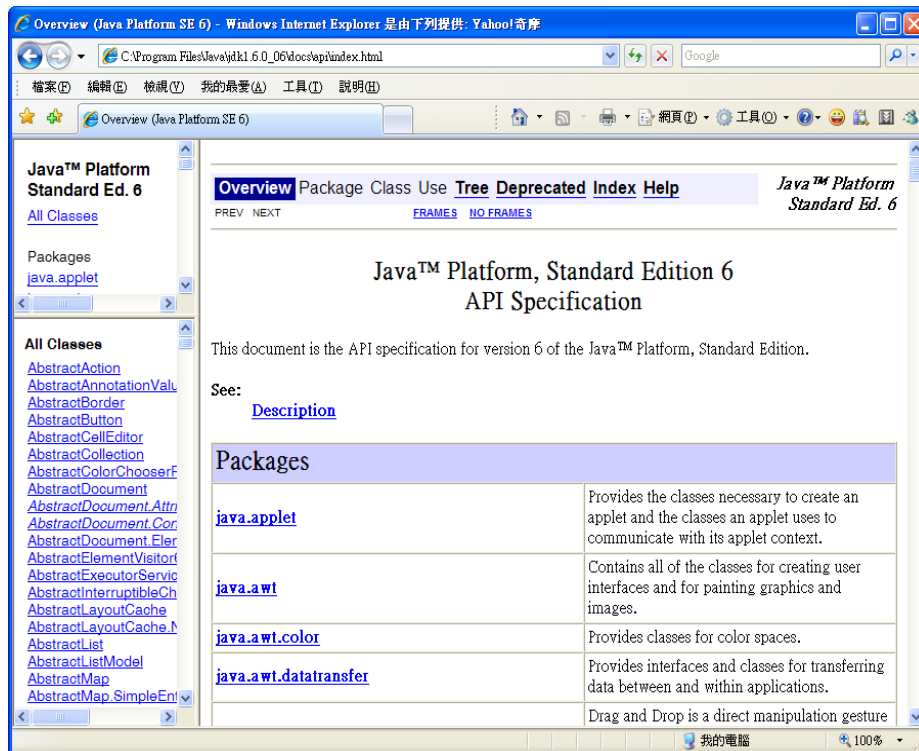
1-2-5 安裝說明文件

- 解壓縮後，會產生一個「**docs**」資料夾，在該資料夾中有個「**index.html**」檔案，打開該檔，您可以看到以下的執行畫面。



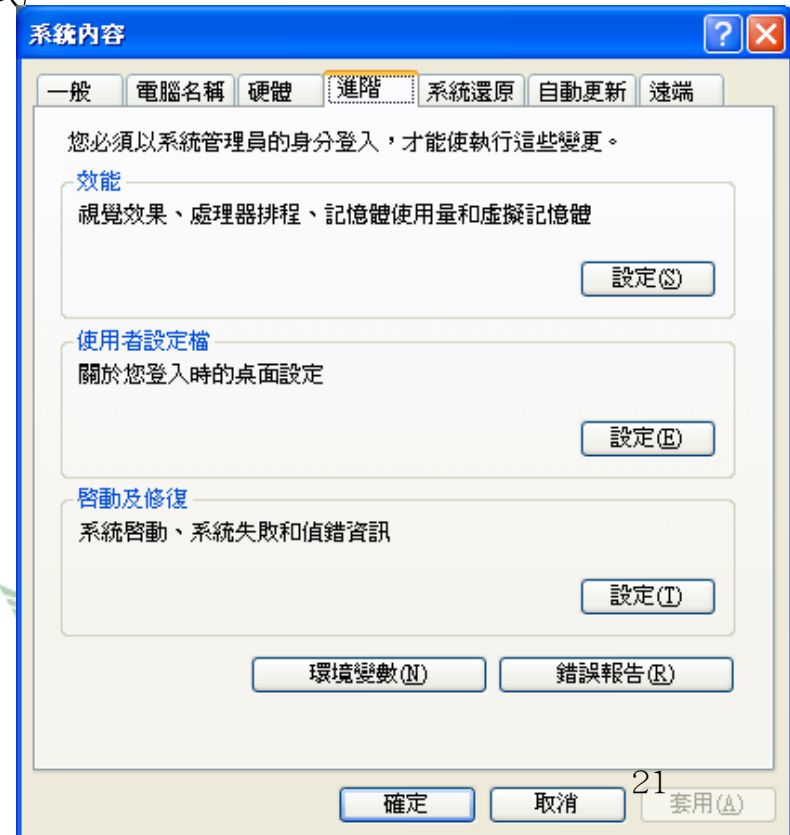
1-2-5 安裝說明文件

- 在該網頁中，往下找到「**API, Language, and Virtual Machine Documentation**」分類，再點選「**Java Platform API Specification**」連結，您可以找到 API Specification，建議您將這個網頁加入「我的最愛」中，以方便日後參考類別函式庫的內容。



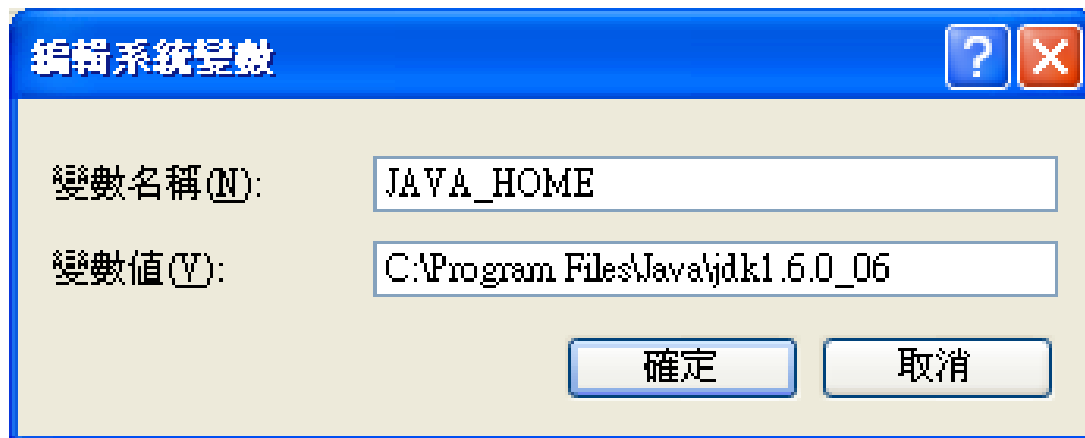
1-3 設定 Java 的執行環境

- 當我們下載並安裝 Java SE 6 後，我們還需要正確的設定相關的環境變數，完成後，我們才可以在我們的機器上編譯並執行 Java 程式。Windows 2000 以後的作業系統都可以採用相同的設定方式。
- 在 Windows 桌面上的「我的電腦」圖示上按右鍵，並選擇「內容」。再按「進階」標籤，再按「環境變數 (N)」按鈕。



1-3 設定 Java 的執行環境

- 進入「新增系統變數」設定視窗後，請在「變數名稱」選項中輸入「**JAVA_HOME**」；「變數值」設定為「**C:\Program Files\Java\jdk1.6.0_06**」，這是先前安裝 Java SE 的資料夾。設定時，字母的大小寫必需相同。設定完成後，按「確定」按鍵，回「環境變數」的設定視窗。



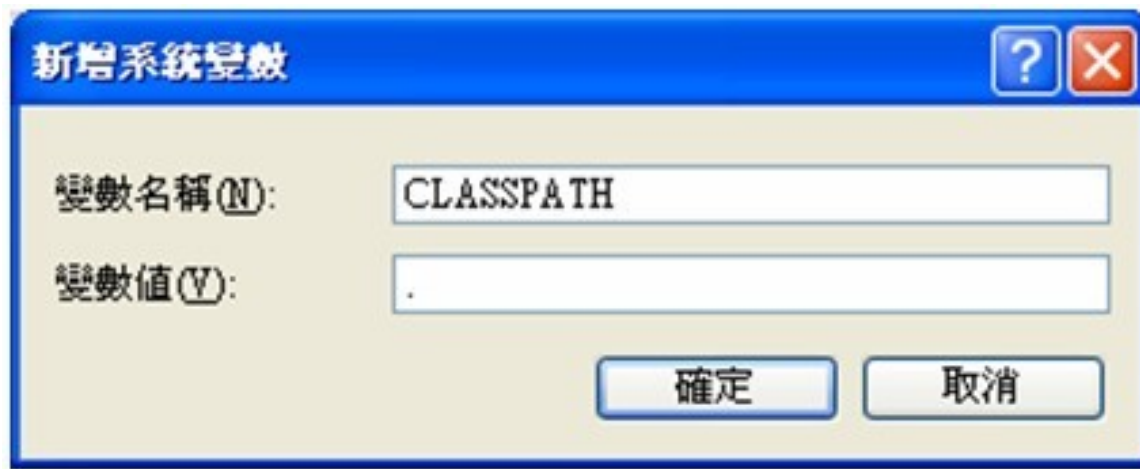
1-3 設定 Java 的執行環境

- ✂ 接下來，我們要設定的是路徑變數。
在「環境變數」視窗中的「系統變數」選項中，按「**Path**」變數，再按「**編輯**」按鈕。
- ✂ 在出現的「編輯系統變數」視窗中：
請在「**變數值**」的最前方加上「**%JAVA_HOME%\bin;**」等文字，同樣的，小心的輸入這些文字，但不要修改其他的內容。設定完成後，按「**確定**」按鍵，再回「環境變數」的設定視窗。



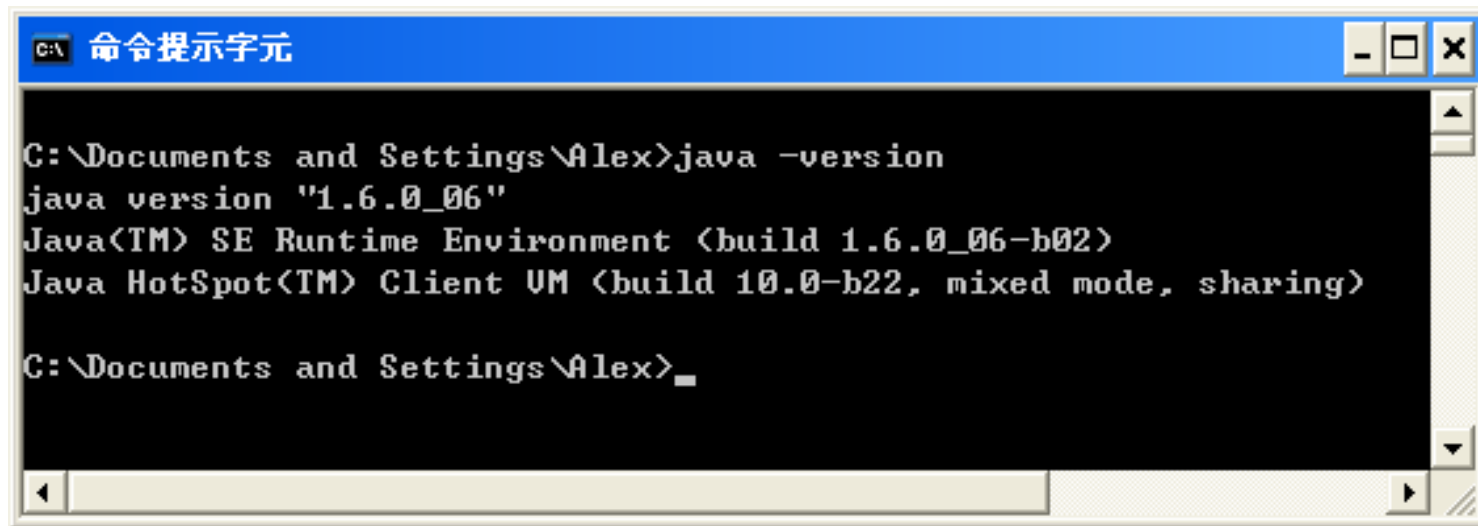
1-3 設定 Java 的執行環境

- 最後，我們還要設定「**CLASSPATH**」變數：
請在「系統變數」選項中，按「新增 (N)」按鈕，在出現的「新增系統變數」設定視窗中，請在「變數名稱」選項中輸入「**CLASSPATH**」；
「變數值」設定為「**.**」。設定完成後，依次按「確定」按鈕，關閉設定視窗。



1-4 測試 Java 的執行環境

- 請執行「開始\程式集\附屬應用程式\命令提示字元」，啟動「命令提示字元」視窗。
- 輸入「**java -version**」指令，再按「**Enter**」鍵，您的視窗中應該會有類似下圖的輸出畫面：

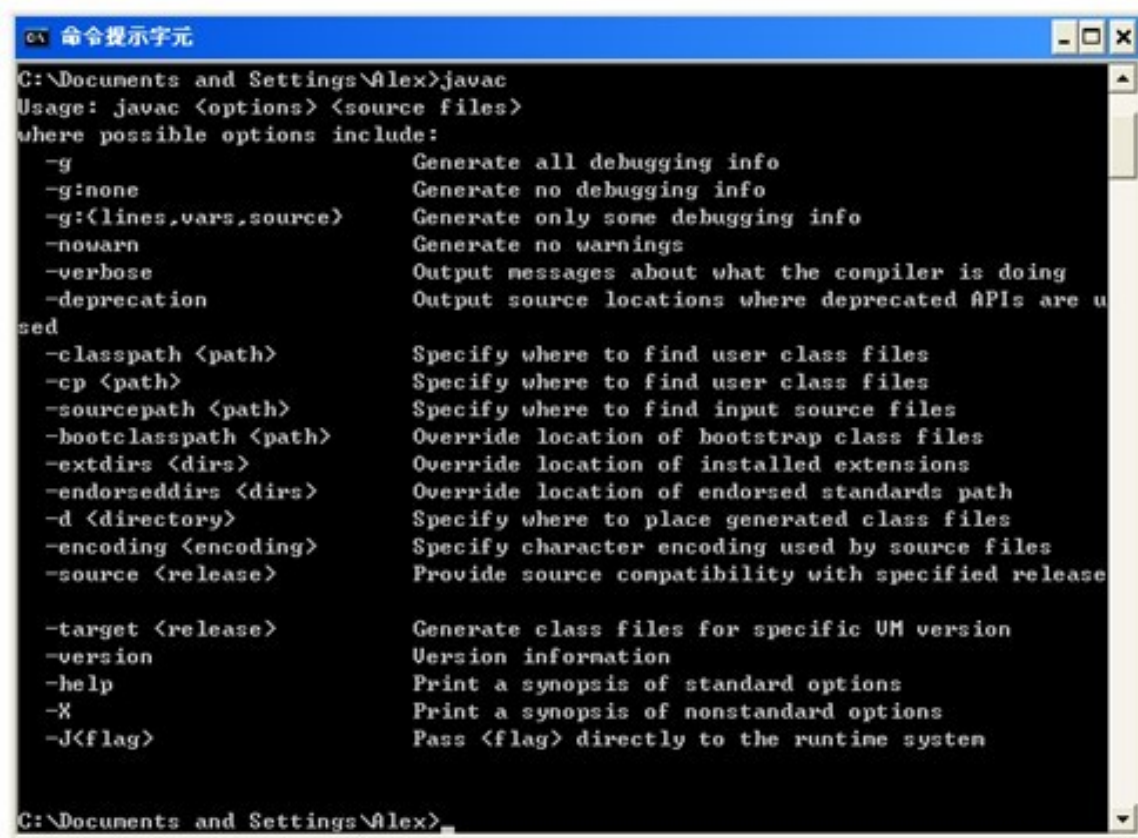


```
C:\Documents and Settings\Alex>java -version
java version "1.6.0_06"
Java(TM) SE Runtime Environment (build 1.6.0_06-b02)
Java HotSpot(TM) Client VM (build 10.0-b22, mixed mode, sharing)

C:\Documents and Settings\Alex>_
```

1-4 測試 Java 的執行環境

- 請再於「命令提示字元」視窗中輸入「**javac**」指令，再按「**Enter**」鍵，您的視窗中應該會有類似下圖的輸出畫面：



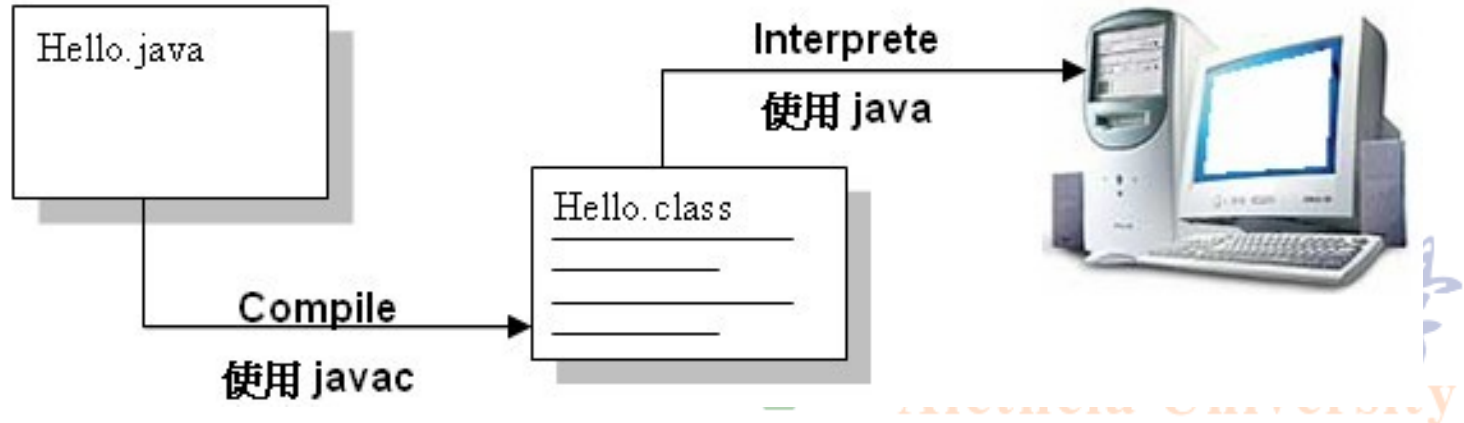
```
GA 命令提示字元
C:\Documents and Settings\Alex>javac
Usage: javac <options> <source files>
where possible options include:
    -g                      Generate all debugging info
    -g:none                 Generate no debugging info
    -g:<lines,vars,source>  Generate only some debugging info
    -nowarn                 Generate no warnings
    -verbose                Output messages about what the compiler is doing
    -deprecation            Output source locations where deprecated APIs are used
    -classpath <path>       Specify where to find user class files
    -cp <path>               Specify where to find user class files
    -sourcepath <path>       Specify where to find input source files
    -bootclasspath <path>   Override location of bootstrap class files
    -extdirs <dirs>          Override location of installed extensions
    -endorseddirs <dirs>    Override location of endorsed standards path
    -d <directory>          Specify where to place generated class files
    -encoding <encoding>    Specify character encoding used by source files
    -source <release>        Provide source compatibility with specified release

    -target <release>        Generate class files for specific VM version
    -version                 Version information
    -help                    Print a synopsis of standard options
    -X                       Print a synopsis of nonstandard options
    -J<flag>                 Pass <flag> directly to the runtime system

C:\Documents and Settings\Alex>
```

1-5 體驗 Java 的程式

- Java 的程式原始檔是一個副檔名為「**.java**」的文字檔，執行 Java 程式前，必需將這個原始檔利用 Java 的「**編譯器 (Compiler)**」編譯成一個副檔名為「**.class**」的 Java bytecodes。我們可以利用「**javac 檔名 .java**」這個指令來完成這項工作。如果原始程式未曾變更，編譯的過程只需要執行一次就好。
- 如果要實際執行該「**.class**」檔，再由 Java 的「**直譯器**」來剖析並執行 Java bytecodes 指令。我們可以用「**java 檔名**」指令來執行程式。過程如下圖



1-5-1 相關的編輯軟體

- **Java** 程式的原始檔只是一個很單純的文字檔，我們可以使用一般的文字編輯軟體，例如：「記事本」、「UltraEdit」、「jEdit」…來撰寫原始碼。
- 在初學時最好不要使用如「JBuilder」…等專業軟體，過多的支援，會讓您失去很多學習的機會，更可能會妨礙您日後的學習成就。
- 建議您使用「記事本」、「UltraEdit」或是「Jedit」來撰寫原始程式。「記事本」程式已內建在 Windows 系統之中，您可以立即使用。
- 如果需要使用「UltraEdit」，您可以至「<http://www.ultraedit.com/>」網站下載試用版，試用版會有 30~45 天的使用期限。或者，您也可以至「<http://www.jedit.com/>」網站下載「Jedit」軟體使用，它是免費的。

1-5-2 以記事本撰寫第一個程式

- 程式 1-1 : Chap1\First.java

```
1.  /**
2.   這是我的第一個 Java 程式
3.   */
4.   class First
5.   {
6.       public static void main(String[] args)
7.       {
8.           // 將需要顯示的內容輸出
9.           System.out.println("Hello World!");
10.      }
11. }
```

1-5-2 以記事本撰寫第一個程式

- Java 的原始碼撰寫完成後，還必需將它編譯成副檔名為「.class」的 Byte code 檔，所使用的工具是利用 JDK 中的「javac.exe」程式來做編譯的工作，指令如下

`javac First.java`

- 如果編譯後，您顯示的畫面如下圖，就代表本程式已經編譯成功了。



1-5-2 以記事本撰寫第一個程式

- 試著執行「**dir**」指令，您應該會看到編譯後所產生的「**First.class**」檔案。



```
C:\> 命令提示字元
C:\Work\Chap1>dir
磁碟區 C 中的磁碟沒有標籤。
磁碟區序號: D023-6311

C:\Work\Chap1 的目錄

2005/09/28 下午 01:27    <DIR>          .
2005/09/28 下午 01:27    <DIR>          ..
2005/09/28 下午 01:26             416 First.class
2005/09/25 上午 09:17             167 First.java
                2 個檔案             583 位元組
                2 個目錄    140,197,425,152 位元組可用

C:\Work\Chap1>
```

1-5-2 以記事本撰寫第一個程式

- 在成功的編譯完程式後，我們就可以執行「First.class」這個 Byte code 檔了。該檔必需在 Java 的 JVM 中執行，程式執行時，必需先執行 JVM，再由 JVM 來執行這個程式。指令如下

java First

- 執行的結果如下圖



```
命令提示字元

C:\Work\Chap1>javac First.java

C:\Work\Chap1>java First
Hello World!

C:\Work\Chap1>_
```


1-6 Java 程式的基本架構

- 程式註解

```
/**
```

這是我的第一個 Java 程式

```
*/
```

- 而 Java 程式中的註解部份可以用三種方式表示：
 - **文件註解：**以「`/**`」開頭，並以「`*/`」結尾。文件註解可以是在同一行或是跨越多行，但請注意，「`/**`」和「`*/`」必需成對的出現。如果您的程式中的註解內容是使用「文件註解」的方式，您還可以利用 JDK 的文件產生工具（`bin\javadoc.exe`）來直接的產生程式的說明文件。
 - **可跨行註解：**以「`/*`」開頭，並以「`*/`」結尾，註解的寫作可以是在同一行或是跨越多行，但請注意，「`/*`」和「`*/`」必需成對的出現，否則，編譯時會產生錯誤。
 - **行內註解：**在程式中以「`//`」符號表示。



1-6 Java 程式的基本架構

- 程式的類別 (**Class**)

```
class First  
{  
    // 相關的程式碼  
}
```

- 程式中的「**class**」是類別宣告的保留字，而「**First**」則是類別的名稱。
- 類別區段的本體是以「**{**」開始（範例程式中的第 5 行），並以「**}**」結束。
- 一個 **Java** 程式中至少需包含一組類別的宣告，我們目前的範例程中只有一個類別，而該類別也包含程式執行的進入點 (**main**)，所以，這個類別的名稱必需和 **Java** 檔案的名稱 (**First.java**) 相同。

1-6 Java 程式的基本架構

- 類別之中的方法 (**Method**)

```
public static void main(String[] args)
{
    // 相關的程式碼
}
```

- 方法是指在類別中宣告的動作或是稱為功能。「方法」的區段的本體也是以「**{**」開始，並以「**}**」結束。
- 程式中的「**main**」是方法的名稱，而在「**main()**」中的「**String**」是指需要傳入此方法的引數（或稱為「參數」）型別，而「**args**」則是在此方法中，能夠使用的引數的名稱。
- 上述的「**main**」方法宣告時，用到的關鍵字如下：
 - **public**：代表這是可以讓其他類別使用的方法。
 - **static**：代表這個方法不需要產生類別的實體即可使用。
 - **void**：代表這個方法在執行後，不回傳值。

1-6 Java 程式的基本架構

- 方法的内容

`System.out.println("Hello World!");`

- 上述的程式碼是一行敘述 (Statement)，敘述要以「**;**」結尾。
- 一個方法之中可以只有單行或是有多行敘述，代表執行某些特定的工作。
- 範例程式中的「**main**」方法中只有一行敘述。
- 在整段程式的執行是在程式啟動時，找到「**First**」這個類別，並由該類別中找到「**main**」這個方法，並執行「**main**」中的程式敘述。

1-6 Java 程式的基本架構

- 程式的內縮處理

- 如果您參考範例程式，你會發現，並不是每一行程式都是由第一個字開始輸入的，而是經過適當的內縮處理，讓我們閱讀程式時會較方便。
- 一般的原則是：包含在某個區段本體的內容會有一次的縮排，例如：第 6 行的「**main**」的宣告就比第 4 行的類別宣告多了一次的內縮，而第 8、9 行的內容又比「**main**」的宣告多了一次的內縮。
- 其實，**Java** 程式在編譯時，是會忽略敘述中的一些特殊字元，例如：空白、內縮處理、換行記號。這代表如果您在撰寫程式時，各行的程式都是齊頭排列，甚至是將所有的敘述寫成同一行，程式依然可以成功的執行，但這只會造成您或是他人閱讀上的不便。

1-7 取得使用者的輸入

- 程式 1-2 : Chap1\Second.java

2. /*

3. 這是我的第二個 Java 程式

4. */

5. class Second

6. {

7. public static void main(String[] args)

8. {

9. System.out.println("Hello " + args[0]);

10. System.out.println(" 您在學 .. " + args[1]);

11. }

12. }



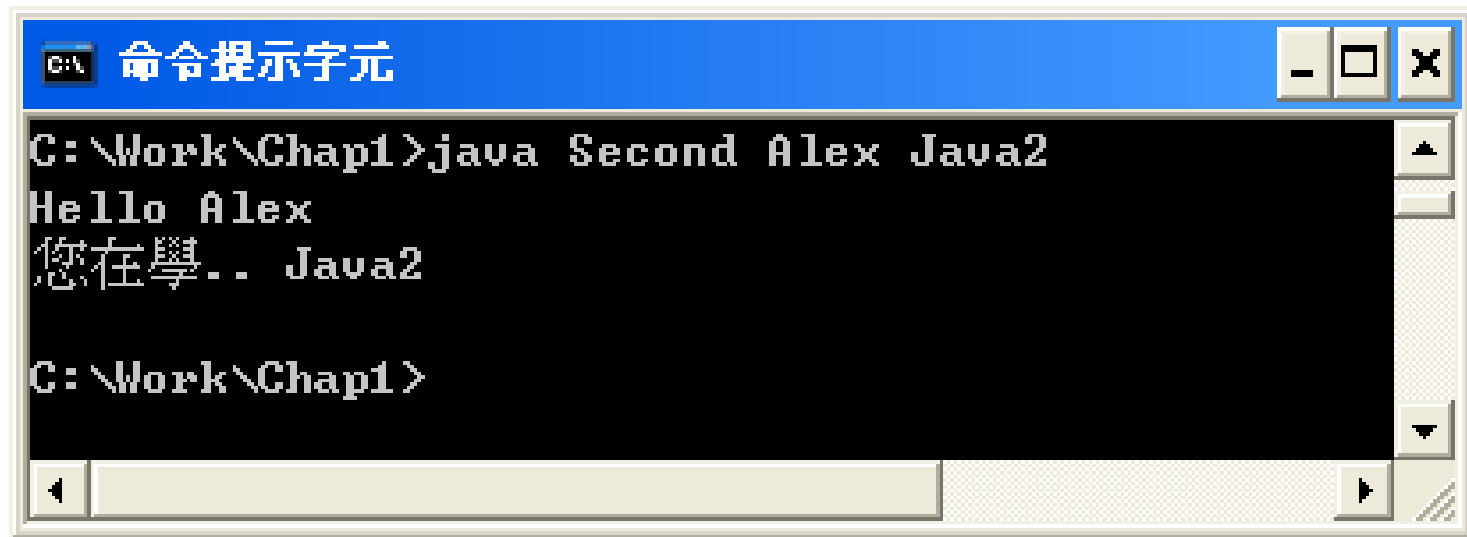
真理大學
Aletheia University

1-7 取得使用者的輸入

- 這次，我們執行程式的指令為：

`java Second Alex Java2`

- 範例程式執行的結果為：



The screenshot shows a Windows Command Prompt window titled "命令提示字元" (Command Prompt). The window has a blue title bar with standard minimize, maximize, and close buttons. The command prompt shows the following text:

```
C:\Work\Chap1>java Second Alex Java2
Hello Alex
您在學.. Java2
C:\Work\Chap1>
```

The output consists of two lines: "Hello Alex" and "您在學.. Java2". The prompt "C:\Work\Chap1>" is shown at the beginning and end of the command sequence.

1-7 取得使用者的輸入

- 指令中的「**Alex**」和「**Java2**」是傳給程式的參數，共有兩個。在「**main**」的方法中，傳入的參數的值會儲存在「**args**」中。所以，第一個參數值「**Alex**」會存在「**args[0]**」之中，而第二個參數值會存在「**args[1]**」之中。程式第 8 行的「**+**」是字串相連的符號，它是用來將兩個字串「連」起來。所以，您會看到執行的結果中會顯示出您輸入的內容。