

# CHAPTER 2

- 從JDK到IDE

## 學習目標

- 瞭解與設定PATH
- 瞭解與指定CLASSPATH
- 瞭解與指定SOURCEPATH
- 使用package與import管理類別
- 初步認識JDK與IDE的對應

# 撰寫Java原始碼



# 撰寫Java原始碼

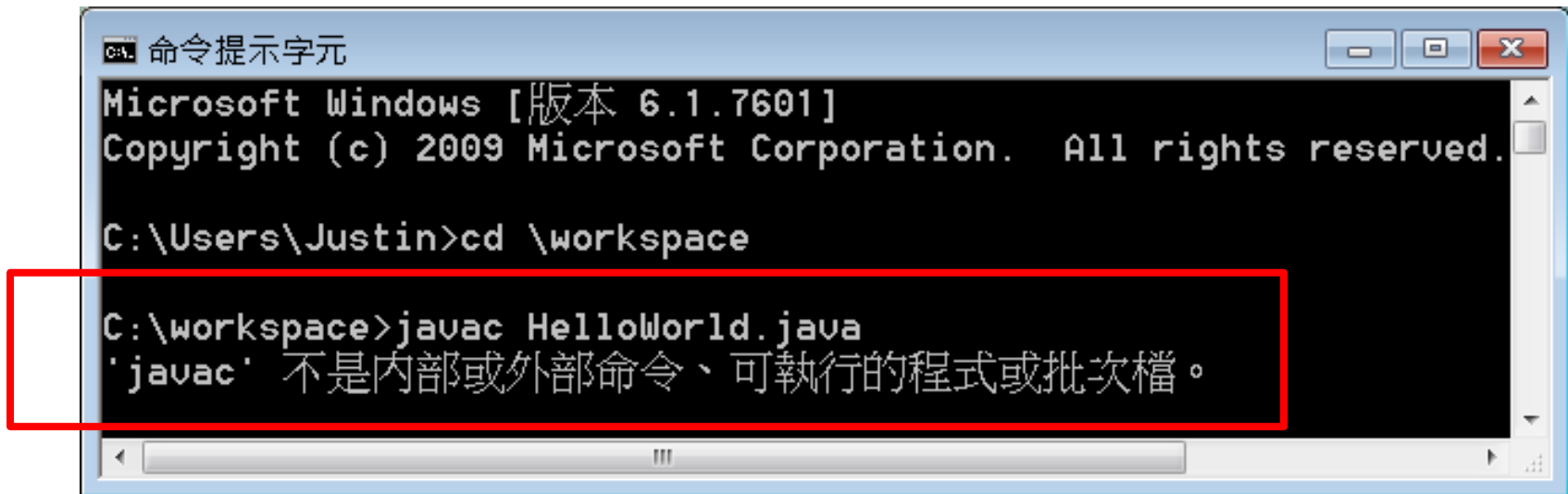
- 副檔名是 .java
- 主檔名與類別名稱必須相同
- 注意每個字母大小寫
- 空白只能是半型空白字元或是Tab字元
- **程式碼不可使用全形文字包括全形空白**

# Console command

- 指令英文字母大小寫皆OK

指令	功能
CD\	切換至根目錄
CD□目錄名	切換至下一層目錄
CD..	切換至上一層目錄
DIR	列出目錄內容
CLS	清除畫面
HELP	查詢所有指令功能

# javac編譯 與 PATH



A screenshot of a Windows command prompt window titled "命令提示字元". The window shows the following text: "Microsoft Windows [版本 6.1.7601] Copyright (c) 2009 Microsoft Corporation. All rights reserved. C:\Users\Justin>cd \workspace C:\workspace>javac HelloWorld.java 'javac' 不是內部或外部命令、可執行的程式或批次檔。". The last two lines are highlighted with a red rectangular box.

```
命令提示字元
Microsoft Windows [版本 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Justin>cd \workspace

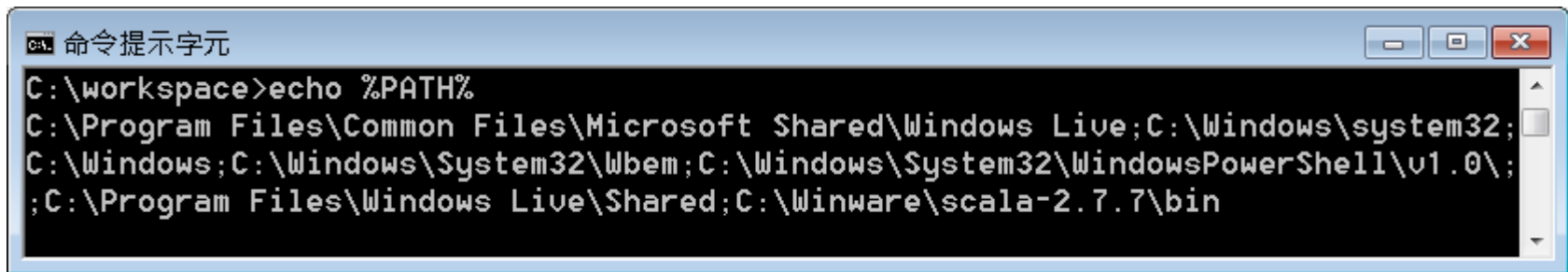
C:\workspace>javac HelloWorld.java
'javac' 不是內部或外部命令、可執行的程式或批次檔。
```

編譯(compile)程式碼：**javac** 檔名.java

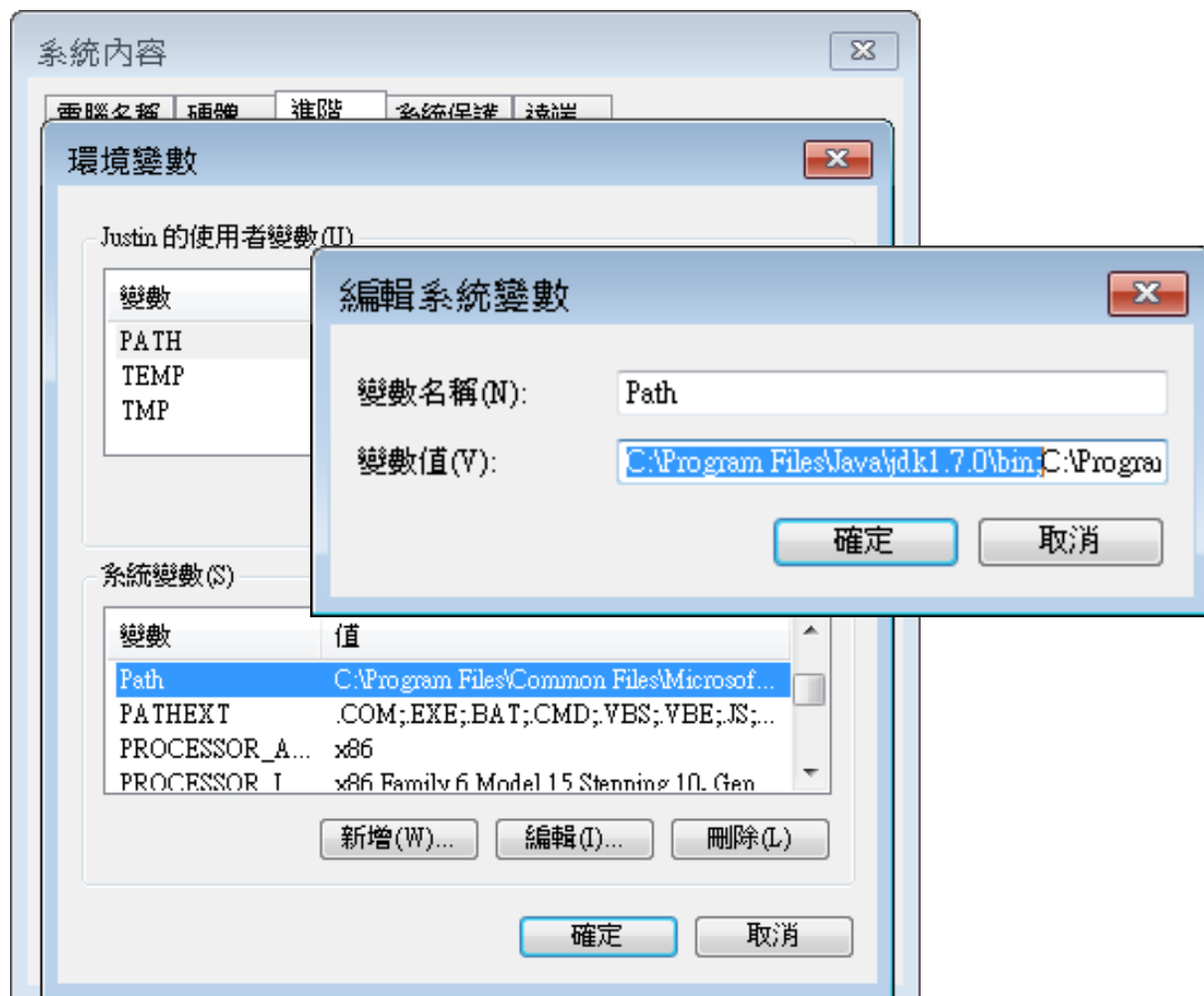
注意檔名大小寫有區分

# PATH是什麼？

## 在Console 下輸入 path 可知道系統設定的Path



```
命令提示字元
C:\workspace>echo %PATH%
C:\Program Files\Common Files\Microsoft Shared\Windows Live;C:\Windows\system32;
C:\Windows;C:\Windows\System32\Wbem;C:\Windows\System32\WindowsPowerShell\v1.0\;
;C:\Program Files\Windows Live\Shared;C:\Winware\scala-2.7.7\bin
```



# 設定PATH

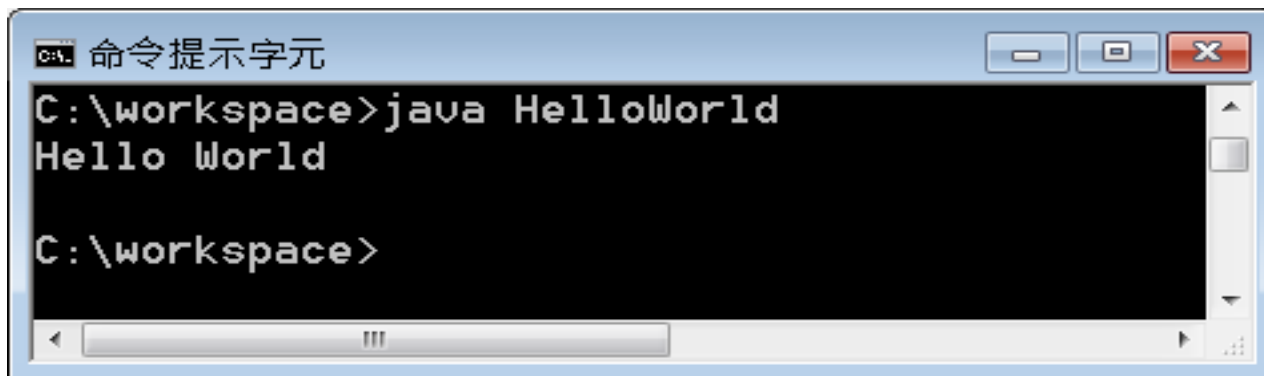
- 建議將JDK的bin路徑放在Path變數的最前方
  - 因為系統搜尋Path路徑時，會從最前方開始，如果路徑下找到指定的工具程式就會直接執行
- 若系統中安裝兩個以上JDK時，Path路徑中設定的順序，將決定執行哪個JDK下的工具程式
- 在安裝了多個JDK或JRE的電腦中，確定執行了哪個版本的JDK或JRE非常重要，確定PATH資訊是一定要作的動作
- **但非必要只安裝一個版本的JDK或JRE**



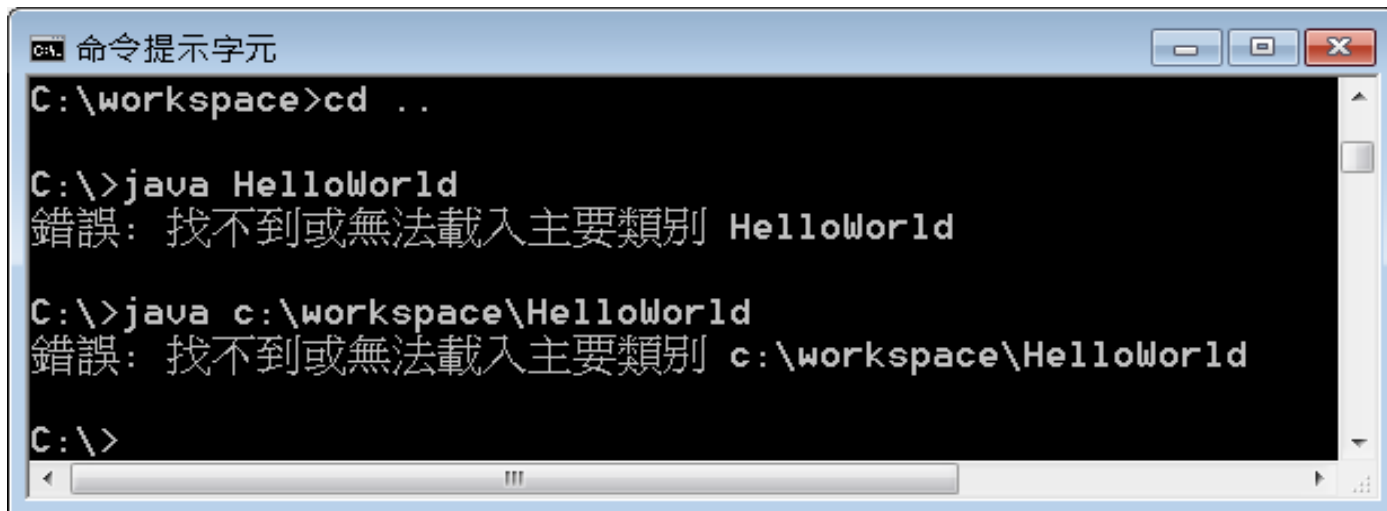
# java 執行工具程式

編譯 (Compile) 程式碼：**javac** 檔名.java

執行 (Run) 位元碼：**java** 檔名



```
命令提示字元
C:\workspace>java HelloWorld
Hello World
C:\workspace>
```



```
命令提示字元
C:\workspace>cd ..
C:\>java HelloWorld
錯誤: 找不到或無法載入主要類別 HelloWorld
C:\>java c:\workspace\HelloWorld
錯誤: 找不到或無法載入主要類別 c:\workspace\HelloWorld
C:\>
```

# JVM ( java ) 與CLASSPATH

- 實體作業系統下執行某個指令時，會依PATH中的路徑資訊，試圖找到可執行檔案
- JVM是Java程式唯一認得的作業系統，對JVM來說，可執行檔就是副檔名為.class的檔案 or .jar的檔案

編譯 (Compile) 程式碼：**javac** 檔名.java

執行 (Run) 位元碼：**java** 檔名

**javac Hello.java** (enter)

**java Hello** (enter)

# Java JAR 檔

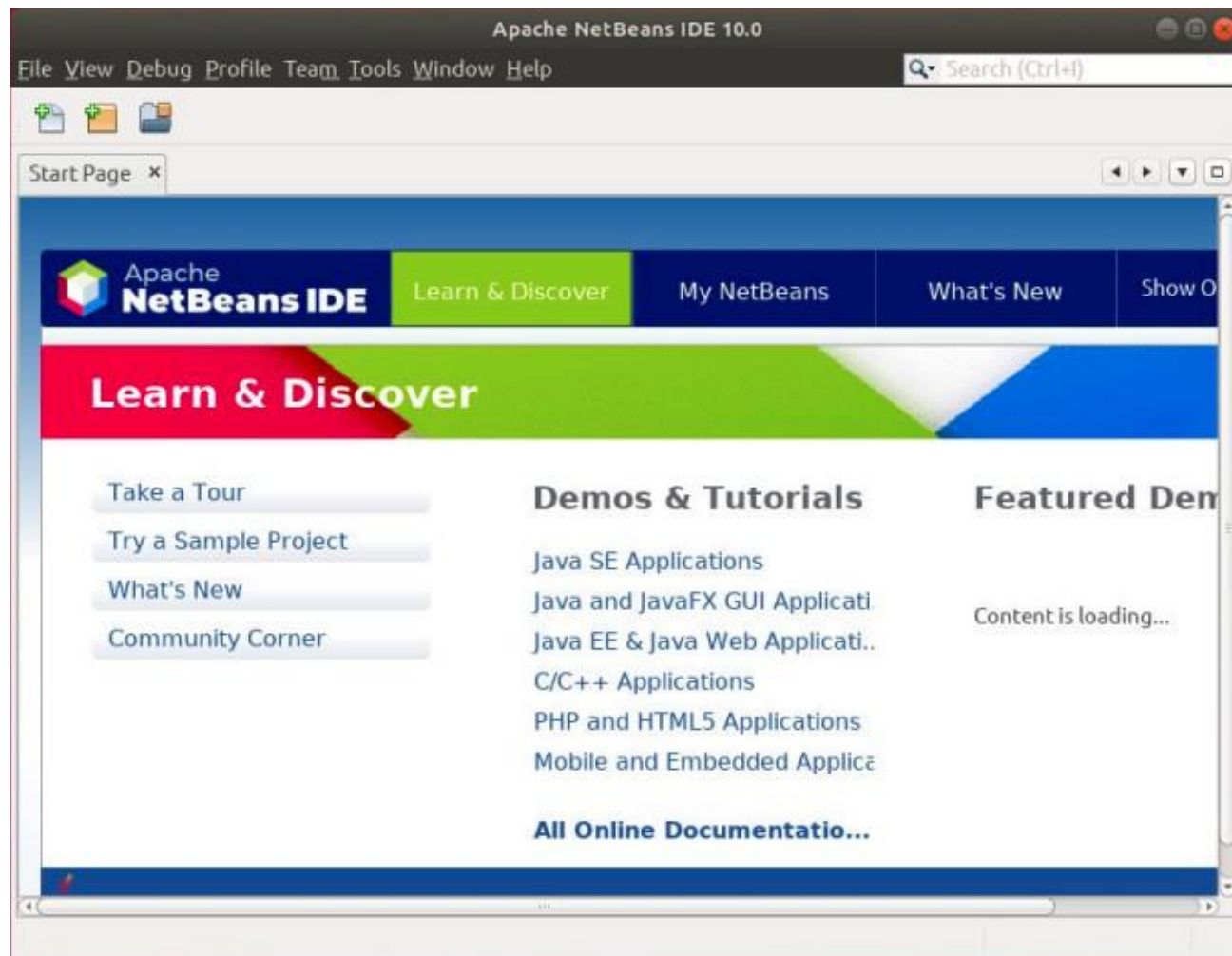
- 程式庫中的類別檔案，會封裝為**JAR ( Java Archive )** 檔案，也就是副檔名為**.jar**的檔案
  - 使用ZIP格式壓縮，當中包含一堆.class檔案
- 一個Java專案完成也會封裝成一個JAR檔作為執行檔

執行專案封裝 **jar : java -jar 檔名.jar**

# Netbeans IDE 整合設計開發軟體



Apache  
**NetBeans IDE**

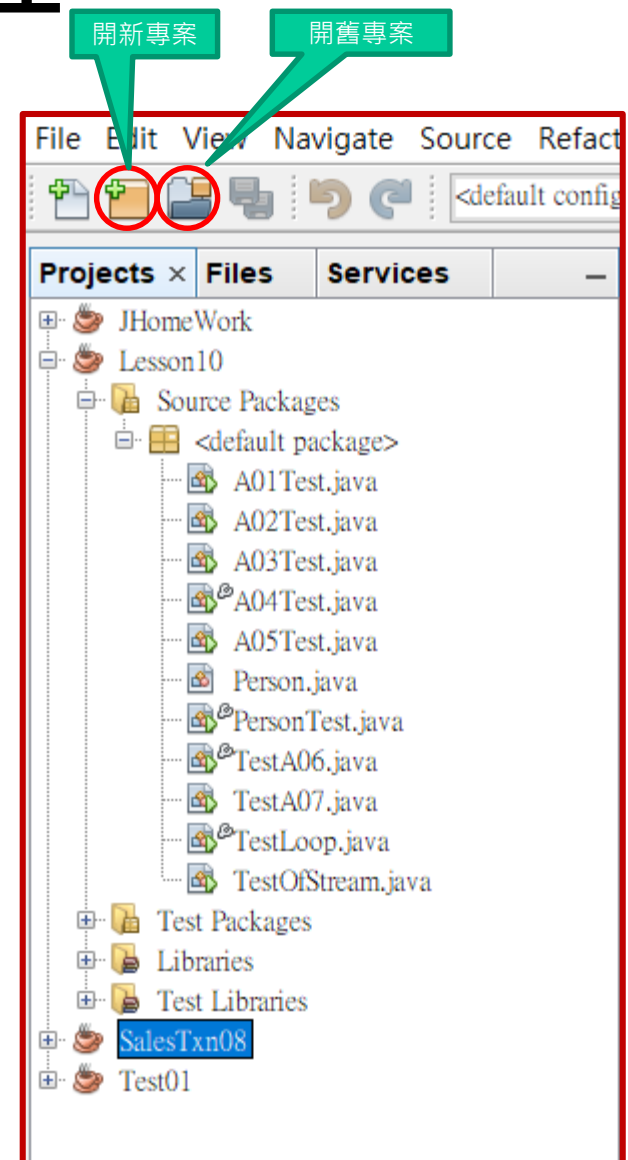
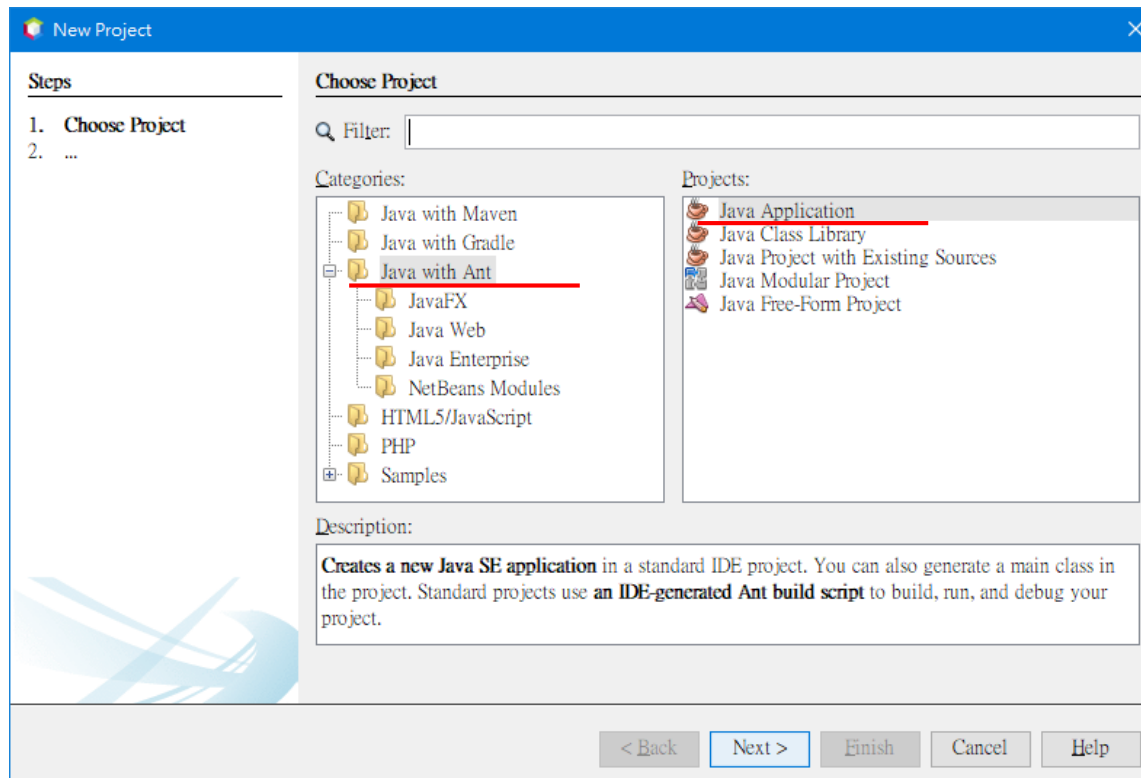


# 建立新專案

- 新增一個空白的專案
- 新增各種 Java 程式檔案，例如 main class 等等
- **NetBean , Eclipse , IntelliJ , Oracle Jdeveloper**  
專業的IDE(整合式開發工具)  
皆有即時偵查編譯和程式編碼輔助功能
- 若語法錯誤也就是 **compilation error** 該行有紅底線

# IDE專案管理

- 正確的專案是一個咖啡圖示
- 一個專案可儲存多個程式檔



# IDE提供許多編碼輔助功能

```
1 package cc.openhome;
2
3 public class Xyz {
4     publ
5     class Xyz is public, should be declared in a file named Xyz.java
6     ----
7     (Alt-Enter shows hints)
8 }
```

```
1 package cc.openhome;
2
3 public class Main {
4     cannot find symbol
5     symbol: class Scanner
6     location: class cc.openhome.Main
7     Scanner scanner;
8
9     Add import for java.util.Scanner
10    Add import for com.sun.java.cup.internal.runtime.Scanner
11    Create class "Scanner" in package cc.openhome
12    Create class "Scanner" in cc.openhome.Main
```

# 管理原始碼與位元碼檔案

- 來觀察一下目前你的  
原始碼（.java）檔案與位元碼檔案（.class）  
都放在一起
- 想像一下，如果程式規模稍大，一堆.java  
與.class檔案還放在一起，會有多麼混亂
- **你需要有效率地管理原始碼與位元碼檔案**



# 使用package管理類別

- 一個專案中 \*.java放在**src**資料夾中，編譯出來的 \*.class放置在**build\classes**資料夾下
- 就如同你會分不同資料夾來放置不同作用的檔案，所以專案中的程式相關檔案也應該分門別類加以放置，
- 專案中的檔案分類管理，使用的項目稱之為 **package (套件)**

# IDE專案管理

