



J01-04

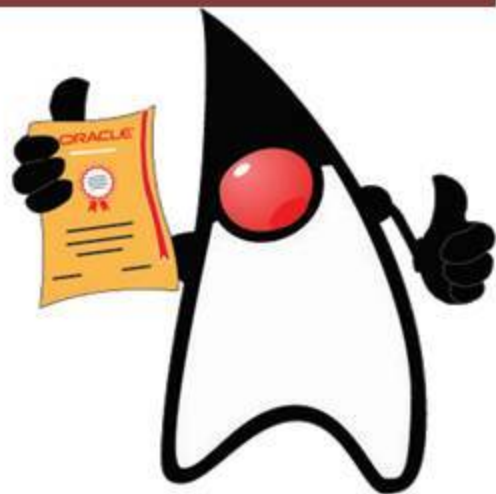
## 認識Java語法與建立類別

曾瑞君 (Jim\_Tzeng)

# 學習目標

1. 定義class，並了解它的成員。
2. 認識「main」method
3. 認識「keywords」
4. 使用Java指令編譯並執行程式
5. 使用Eclipse匯入範例專案並執行程式





1/5

定義class，並了解它的成員



# 建立 class

- 宣告 class
- 宣告 **fields** (欄位，optional)
  - 來自物件的屬性
  - 屬性一定有 value / state，宣告時必須告知「型態」
  - 宣告時可以一併給值
- 宣告 **methods** (方法，optional)
  - 來自物件的行為，需描述行為的內容
  - 物件行為執行後，可分成
    - 有結果，必須宣告其結果的「型態」
    - 沒結果，必須宣告其型態為「void」
- 使用 comments (註解，optional)
  - 註解程式內容



# 建構 class

- 宣告 class，語法：

```
[modifiers] class class_identifier {  
    class_code_block  
}
```

- 開啟文字編輯程式如Notepad++撰寫程式：

```
class Shirt {  
  
}
```

# 建構 class – fields (欄位)

- 宣告 **fields** (optional)
  - 來自物件的屬性
  - 屬性有 value / state，宣告時必須告知「**型態**」
  - 宣告時可以一併給 value / state

```
class Shirt {  
    int size;  
    double price = 100.5;  
}
```

# 建構 class – methods (方法)

- 宣告 **methods** (optional) , 語法為 :

```
[modifiers] return_type method_identifier ( [arguments] ) {  
    method_code_block  
}
```



# 建構 class - methods

- 宣告 **methods** (optional)
  - 來自物件的**行為**，需描述行為的內容
  - 物件**行為**執行後，分成

有結果，  
必須宣告其結果的「**型態**」

沒結果，  
必須宣告其型態為「**void**」

```
class Shirt {  
  
    int size;  
    double price = 100.5;  
  
    double getPrice() {  
        return price;  
    }  
  
    void display() {  
        System.out.println("size = " + size);  
        System.out.println("price = " + price);  
    }  
}
```



# 建構 class – comments (註解)

- 使用 comments (optional)

- 註解程式內容

```
/* this is a class
 * to show you what a class is.
 */
class Shirt {

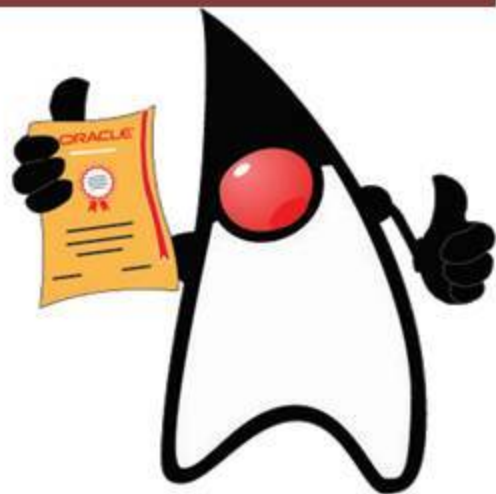
    // this is a field
    int size;
    double price = 100.5;

    //this is a method
    double getPrice() {
        return price;
    }
}
```

# 建構 class

- 使用的特殊符號

| 符號  | 名稱    | 用途             |
|-----|-------|----------------|
| { } | 大括號   | 一段程式碼          |
| ( ) | 小括號   | 方法名稱後，可放參數     |
| ;   | 分號    | 一個statement的結束 |
| ,   | 逗號    | 分離變數和值         |
| ' ' | 單引號   | 用於字元           |
| " " | 雙引號   | 用於字串           |
| //  | 雙斜線符號 | 用於註解           |



2/5

認識「keywords」



# Java 關鍵字 (保留字)

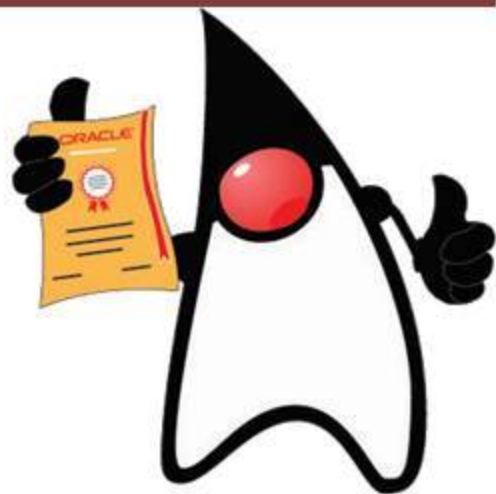
- 在程式中已被 compiler 賦予特殊意義，不可用在 class、field、method 的命名：

|          |          |            |           |              |
|----------|----------|------------|-----------|--------------|
| abstract | continue | for        | new       | switch       |
| assert   | default  | goto       | package   | synchronized |
| boolean  | do       | if         | private   | this         |
| break    | double   | implements | protected | throw        |
| byte     | else     | import     | public    | throws       |
| case     | enum     | instanceof | return    | transient    |
| catch    | extends  | int        | short     | try          |
| char     | final    | interface  | static    | void         |
| class    | finally  | long       | strictfp  | volatile     |
| const    | float    | native     | super     | while        |



- 詳細內容可以參考  
[https://docs.oracle.com/javase/tutorial/java/nutsandbolts/\\_keywords.html](https://docs.oracle.com/javase/tutorial/java/nutsandbolts/_keywords.html)。其中的const與goto在新版的Java已經不再使用。
- 關鍵字都是以小寫開頭，但不用特別去記憶。隨著對Java程式語言的了解自然就會認識。





3/5

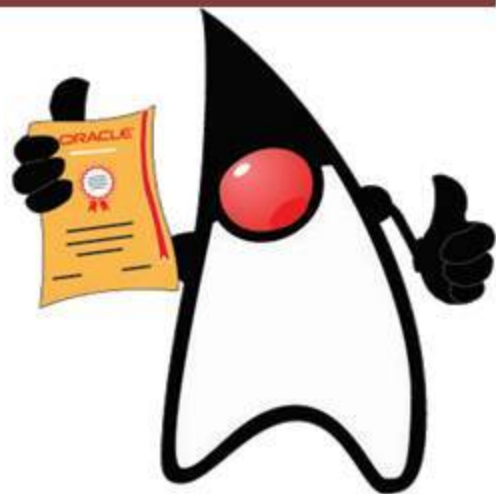
## 認識「main」method

# 【main】方法

- Java 內特殊的method。不屬於物件的行為，而是用來發動程式。
- 名稱必須是【main】，區分大小寫。語法：

```
class Shirt {  
  
    public static void main (String[] args) {  
        /*  
         * what you want to launch  
         */  
    }  
  
}
```





4/5

## 使用**JAVA**指令編譯並執行程式

# 建立測試類別，並建立main方法

- 建立一個 ShirtTest 類別，並建立方法 main()
- 執行內容為：
  - 由 Shirt 類別產生 Shirt 物件/實例 (instance)
  - 執行 Shirt 物件的 display() 方法

```
class ShirtTest {  
  
    public static void main (String[] args) {  
  
        new Shirt();  
  
        new Shirt().display();  
    }  
}
```

# 編譯 並 執行

- Java 的程式碼檔案附檔名須為“\*.java”
- 編譯
  - 語法：`javac java程式檔`
  - 編譯後產生 \*.class 檔案
  - 如 `javac ShirtTest.java`，將產出 `ShirtTest.class` 檔案
- 執行
  - 語法：`java java程式編譯檔` (不能加class的副檔名)
  - 執行的程式中，必須有main方法作為程式進入點
  - 如 `java ShirtTest` (無附檔名)

# 編譯Java程式

本機 > OSDisk (C:) > java11 > code > ch04

名稱

Shirt.class

Shirt.java

ShirtTest.class

ShirtTest.java

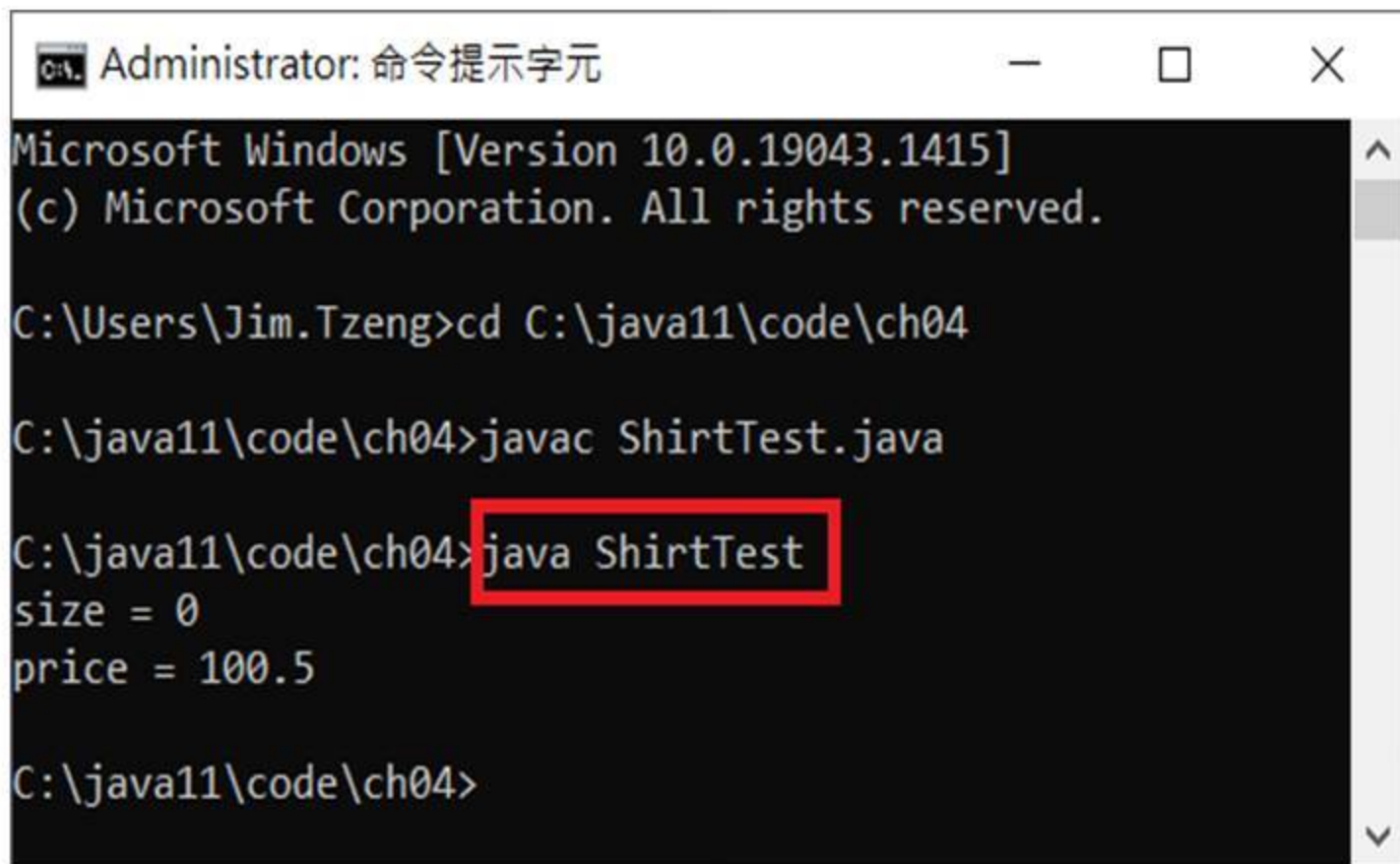
```
Administrator: 命令提示字元
Microsoft Windows [Version 10.0.19043.1415]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Jim.Tzeng>cd C:\java11\code\ch04

C:\java11\code\ch04> javac ShirtTest.java

C:\java11\code\ch04>
```

# 執行Java程式



```
Administrator: 命令提示字元
Microsoft Windows [Version 10.0.19043.1415]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Jim.Tzeng>cd C:\java11\code\ch04

C:\java11\code\ch04>javac ShirtTest.java

C:\java11\code\ch04>java ShirtTest
size = 0
price = 100.5

C:\java11\code\ch04>
```





# 建立Java程式檔

---

Java的程式檔案名稱和檔內定義的class有若干關係：

- 程式碼檔案附檔名必須是「.java」
- 程式碼檔案內可以定義多個非 public 的 class，檔名不一定要和這些 class 名稱相同。編譯後，檔案內宣告的每個 class 都會各自產生編譯檔。
- 程式碼檔案內若有 public 的 class，則檔名必須和該 public 的 class 的名稱相同。而且一個程式碼檔案內只允許一個宣告為 public 的 class。

# 使用jar指令打包程式後再執行

- 以先前的 ShirtTest.java 與 Shirt.java 為例，2個類別具有關連性，因此編譯時必須一起存在，執行時也必須一起存在。當專案逐漸龐大，程式裡還有其他設定檔、圖檔、資料夾等時，相當不易維護！
- 解決方式是把相關檔案打包一起存成JAR 檔案。JAR是「Java Archive」的縮寫，它是一個ZIP 壓縮檔，可以使用檔案壓縮工具如 7-Zip 瀏覽內容。事實上也可以使用 7-Zip 製作 JAR 檔案，但不比 Java 的內建指令「jar.exe」有效率。
- 指令在安裝目錄的bin資料夾內，分成5段：

**jar** -cfe JAR檔案 具備main()方法的類別 程式編譯檔





## jar -cfe JAR檔案 具備main()方法的類別 程式編譯檔

1. 第一部分是 jar 指令。
2. 第二部分是指令選項，jar 指令有很多選項可以使用，這裡使用 cfe 分別代表：
  - 指令選項 c 指 create，表示指令要建立新 JAR 檔案。
  - 指令選項 f 指 file，選項後要提供 JAR 檔案名稱。
  - 指令選項 e 指 entry point，表示程式進入點，要指定具備 main() 方法的類別名稱。
3. 第三部分是具備 main() 方法的類別名稱。
4. 第四部分是要產生的 JAR 檔案名稱。
5. 第五部分是程式編譯檔，就是 \*.class。所以在使用 jar 指令前，要先完成編譯。





- 假如要把 ShirtTest.class 和 Shirt.class 打包成 shirt.jar 檔案，可以使用以下指令：

```
jar -cfe shirt.jar ShirtTest *.class
```

- 或是清楚指定 \*.class 包含哪些檔案：

```
jar -cfe shirt.jar ShirtTest Shirt.class ShirtTest.class
```

- 打包完成後，可以使用以下指令執行 shirt.jar 檔案：

```
java -jar shirt.jar
```



> 本機 > OSDisk (C:) > java11 > code > ch04

名稱

Shirt.class

shirt.jar

Shirt.java

ShirtTest.class

ShirtTest.java

命令提示字元

Microsoft Windows [Version 10.0.19043.1415]  
(c) Microsoft Corporation. All rights reserved.

C:\Users\Jim.Tzeng>cd C:\java11\code\ch04

C:\java11\code\ch04>jar -cfe shirt.jar ShirtTest \*.class

C:\java11\code\ch04>jar -cfe shirt.jar ShirtTest Shirt.class ShirtTest.class

C:\java11\code\ch04>java -jar shirt.jar

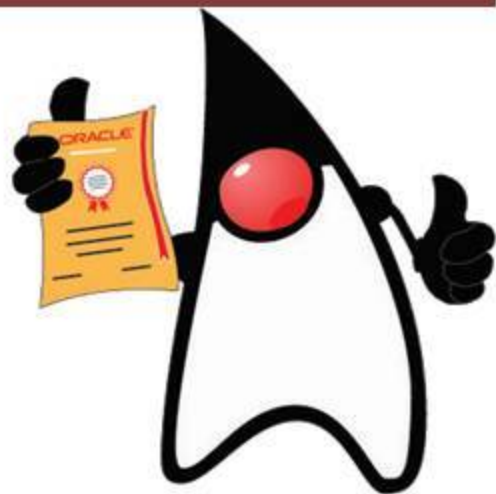
size = 0  
price = 100.5

C:\java11\code\ch04>



- JAR檔用來執行Java SE程式時，因為可執行(executable)的特性，被稱為「Executable JAR」。
- 還有一種不是用來執行，只是單純做為函式庫的，就是一般JAR檔。Java是開源(open source)的程式語言，擁有很多社群(community)釋出函式庫，使用它們可以節省許多開發時間。





5/5


# 使用 **ECLIPSE** 匯入範例專案並執行程式



- 我們在第二章示範使用Eclipse建立Java程式並執行，也說明了如何匯入Eclipse專案。接下來就匯入本書上冊的完整範例專案「java11-ocp-1」，並執行前述的ShirtTest類別。







# 使用套件(package)

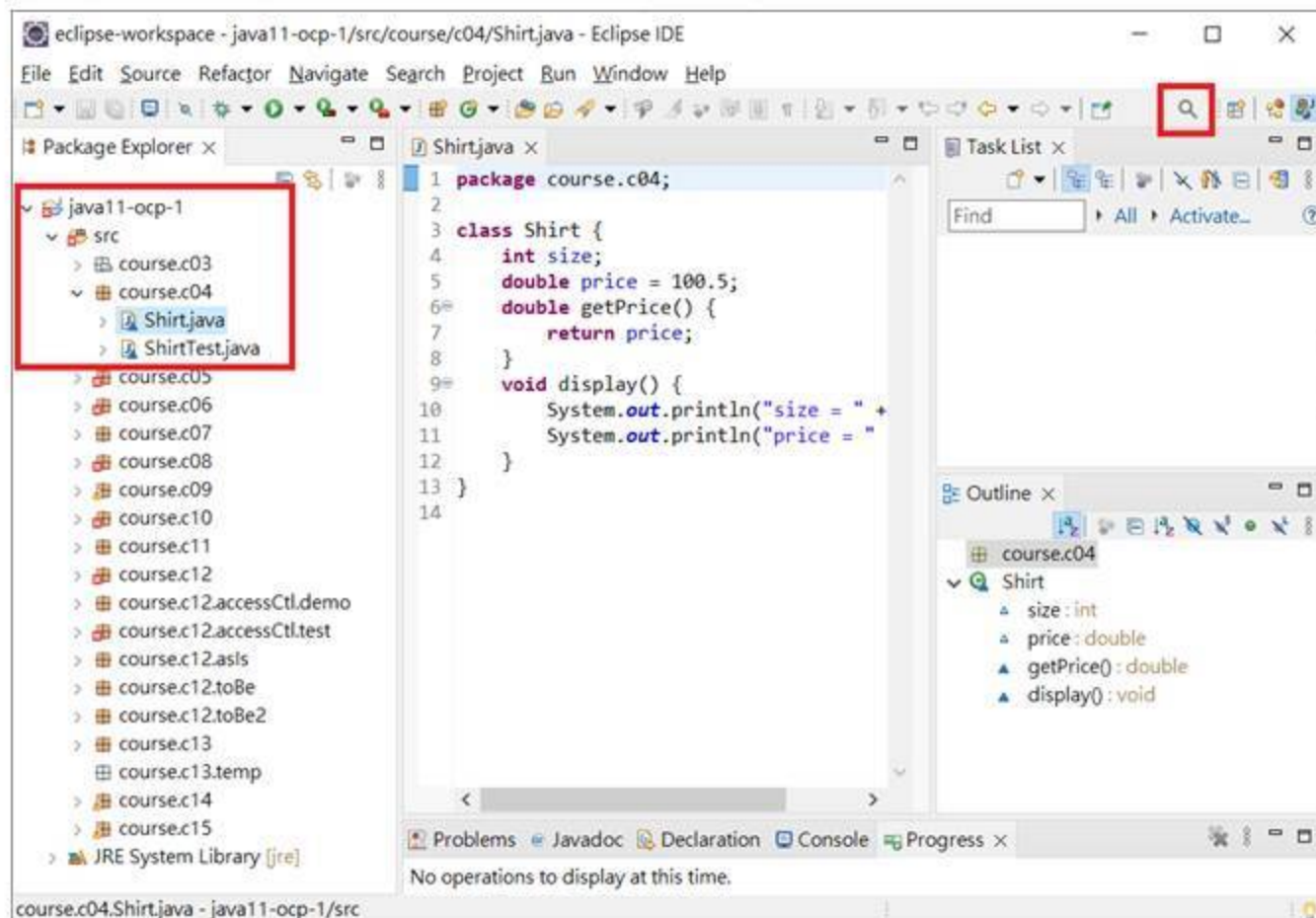
---

- 關鍵字「package」代表的意義是對類別進行「分類」，會和類別所在的檔案目錄有關。本書的範例類別眾多，需要依照不同章節分類，所以在簡單的範例中就使用了package宣告。參考後續步驟了解package的使用目的。



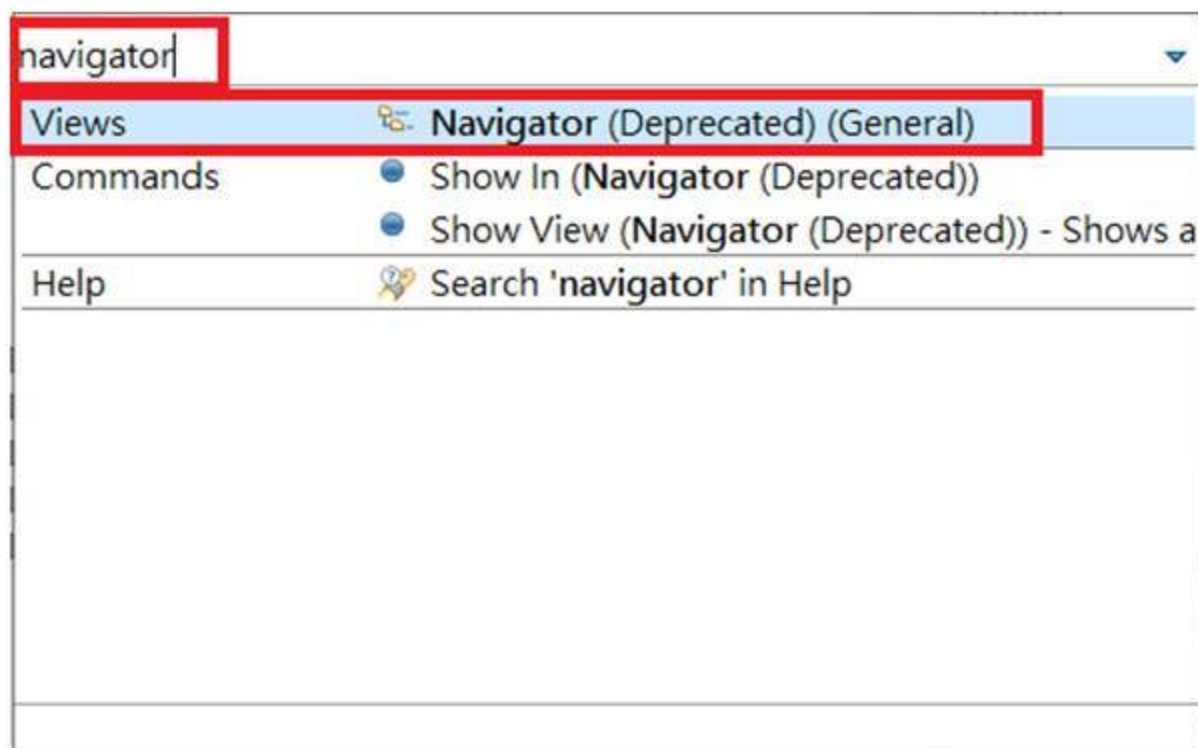
# 【STEP01】

- 參照章節二匯入本書範例專案java11-ocp-1後，依下圖展開Shirt.java類別，並點擊右上角的放大鏡圖示：



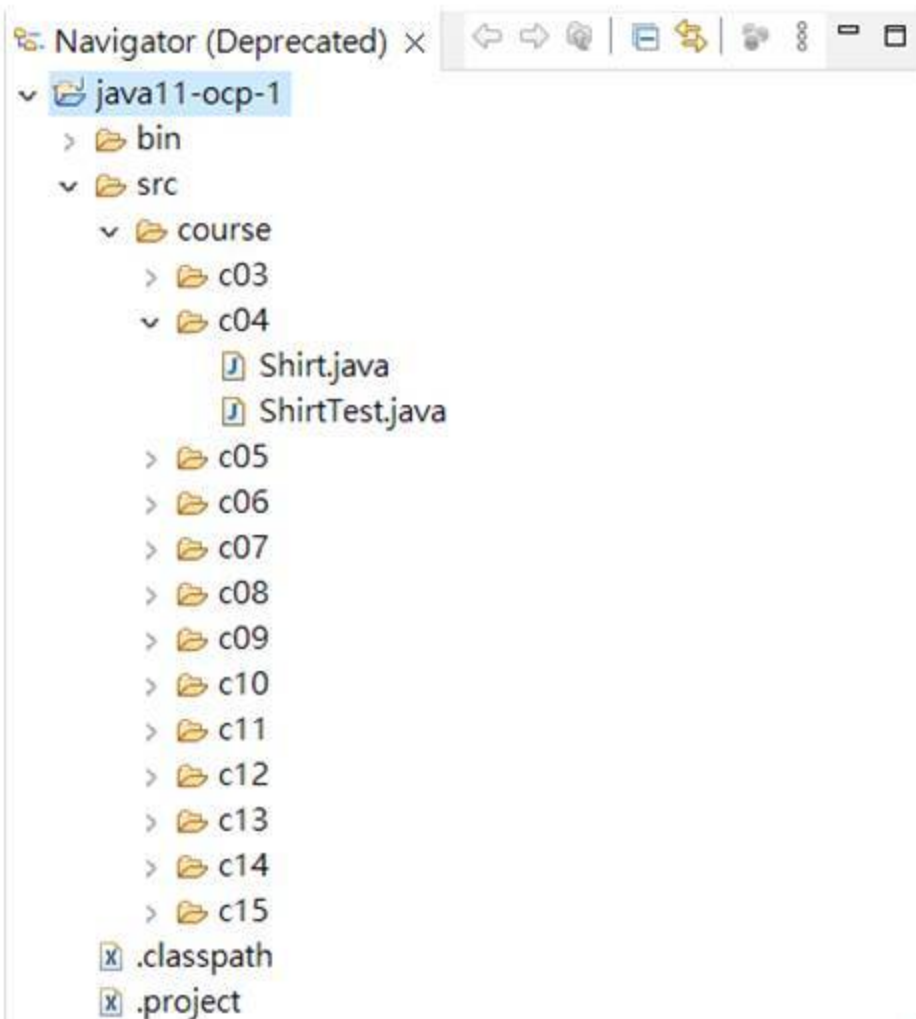
## 【STEP02】

- 在彈出視窗中輸入「navigator」，點擊搜尋結果：



# 【STEP03】

- 彈出 Navigator 視窗：



# 【STEP04】

- 關注預設的  
Package Explorer視  
窗：





# Navigator與Package Explorer視窗的差異

---

- Navigator視窗以檔案系統的「樹狀結構」呈現，類似檔案總管；每個類別就是\*.java的檔案，位在特定的目錄中。檔案目錄「src」是專案的根目錄，所有Java類別檔都必須放在該目錄下才能編譯。如Shirt.java位於「src/cource/c04」的目錄中。
- Package Explorer視窗是以Java的套件(package)呈現專案，沒有樹狀結構。以Shirt.java為例，其位於「cource.c04」套件下。





# Navigator與Package Explorer視窗的差異

---

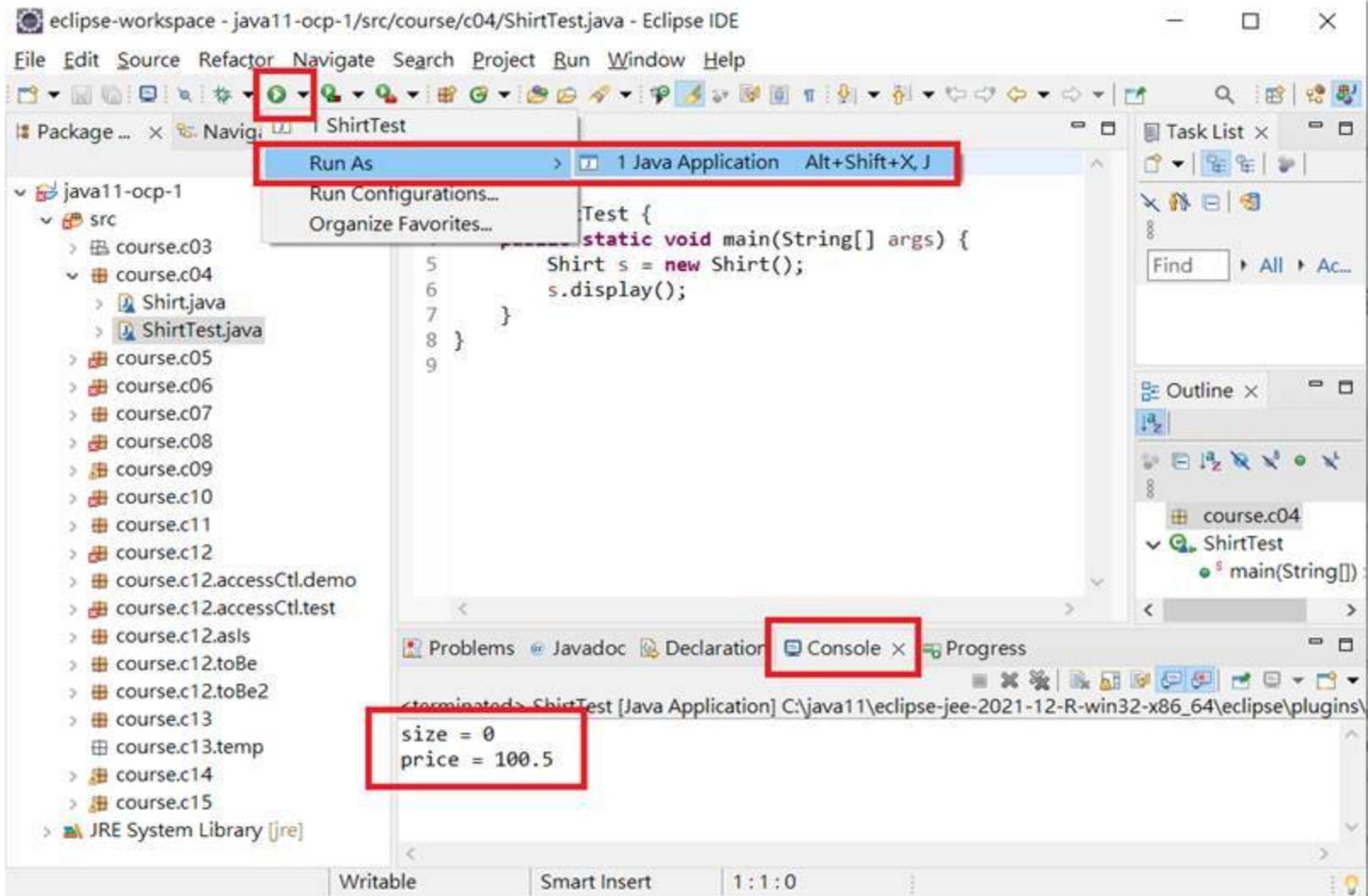
- Package Explorer視窗可以顯示編譯失敗的類別。  
Navigator視窗因為是以檔案總管的角度呈現類別目錄，將不顯示編譯結果。Eclipse匯入專案後預設會自動編譯，將發現部分類別編譯失敗，這是課程設計的需要，屬正常現象。



- 因為加入套件的關係，類別 Shirt 與 ShirtTest 的行 1 都多了「`package course.c04;`」敘述。
- 選擇具備 `main()` 方法的類別 ShirtTest，點選工具列的程式執行鍵，選擇「Run As」，再選擇「Java Application」，就可以執行程式。結束後在下方 Console 視窗可以看到執行結果：









---

**END** ~~

Thank you!!

