Object-Oriented Concept & Java Language Structure

物件導向基礎

Java Fundamental

Content

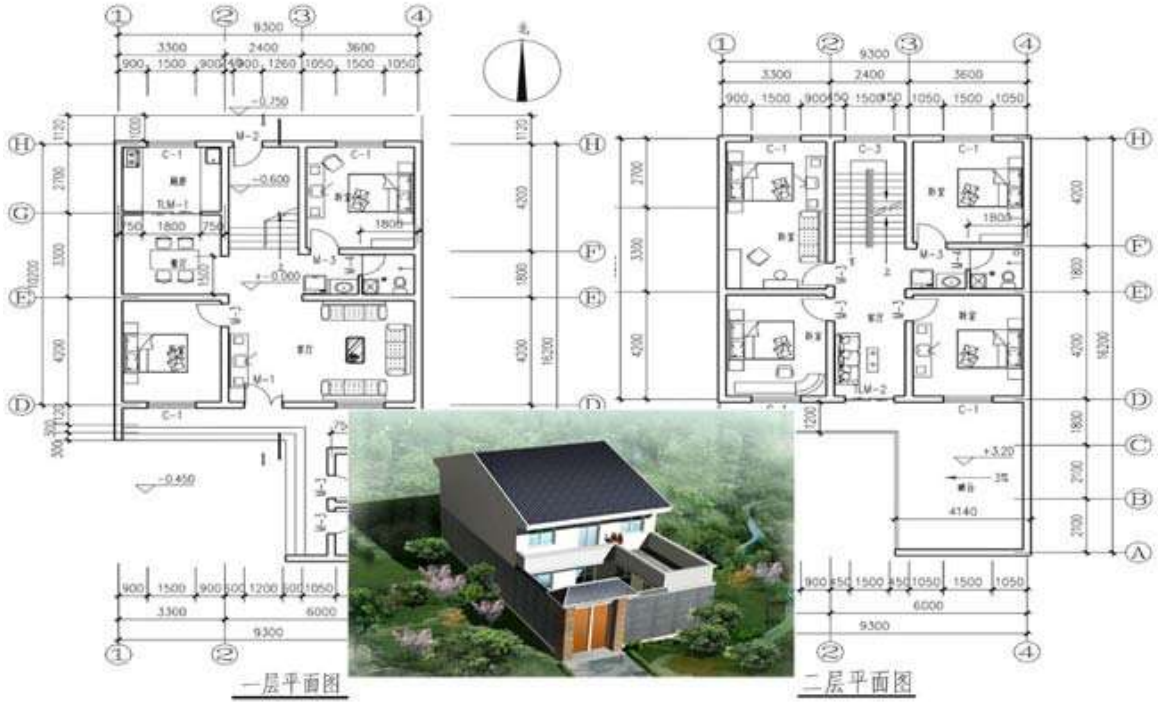
- 類別與物件
- 物件導向開發
- Java程式結構

Content

- 類別與物件
- 物件導向開發
- Java程式結構

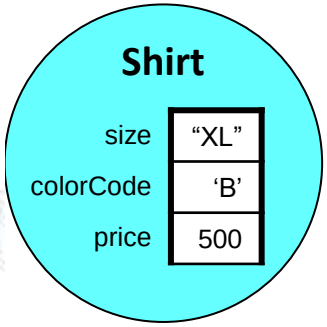
類別 vs. 物件

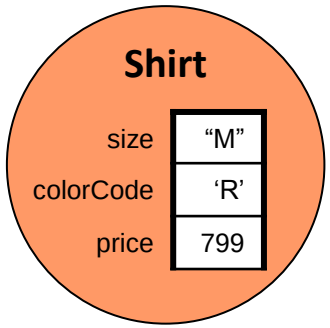
- 類別如同藍圖　物件如同房子
- 物件根據類別產生。(根據藍圖來蓋房子)
- 沒有類別，沒有物件。 (沒有藍圖，房子無法蓋)

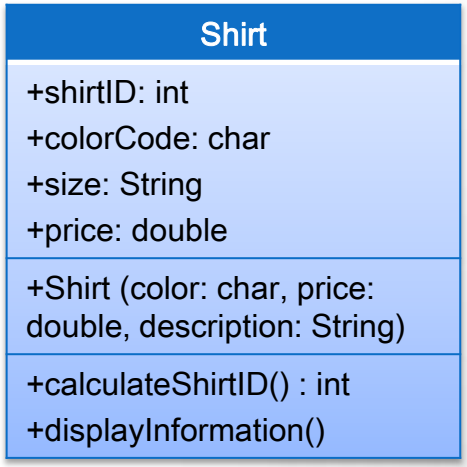


類別 vs. 物件

- 類別
  - 程式設計師以類別來定義同類型物件的共同藍圖
  - 在Java中類別也可以是一種型別定義
- 物件
  - 物件是類別的一個實體 (房子是根據藍圖建立)
  - 兩件衣服是同一個類別的不同實體 (房子A 和房子B 都是根據同一藍圖建立)

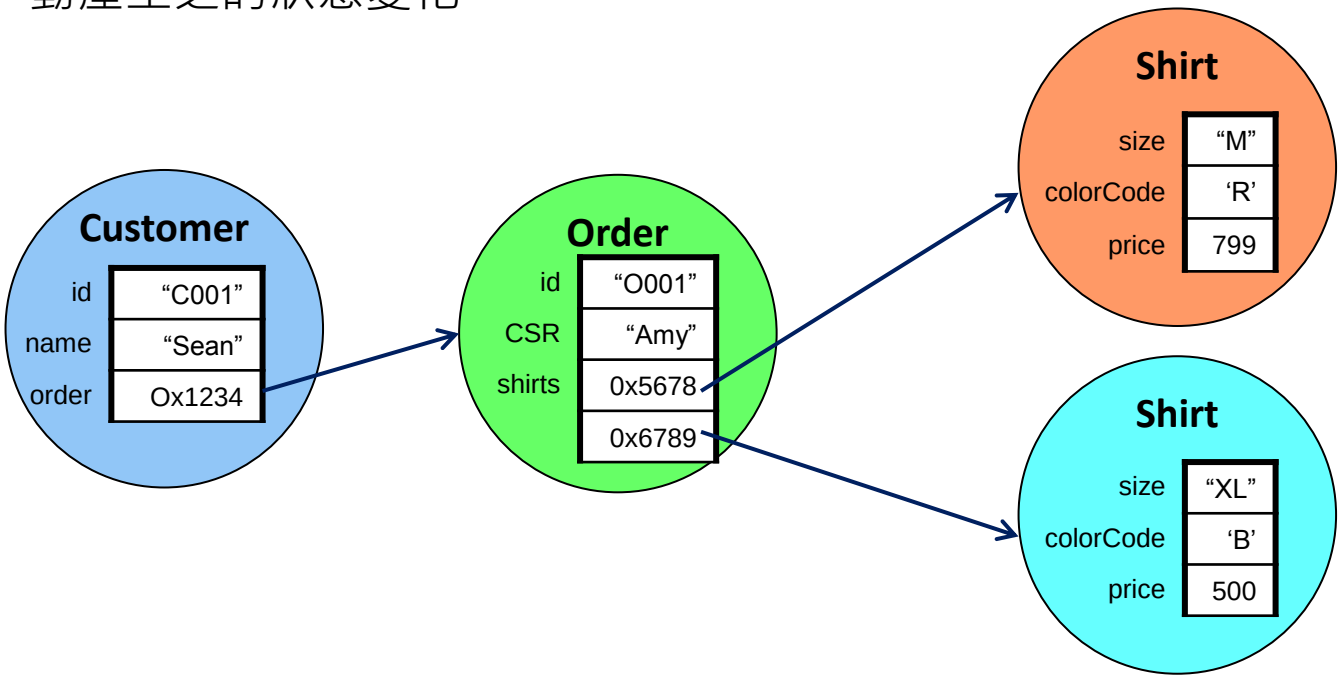






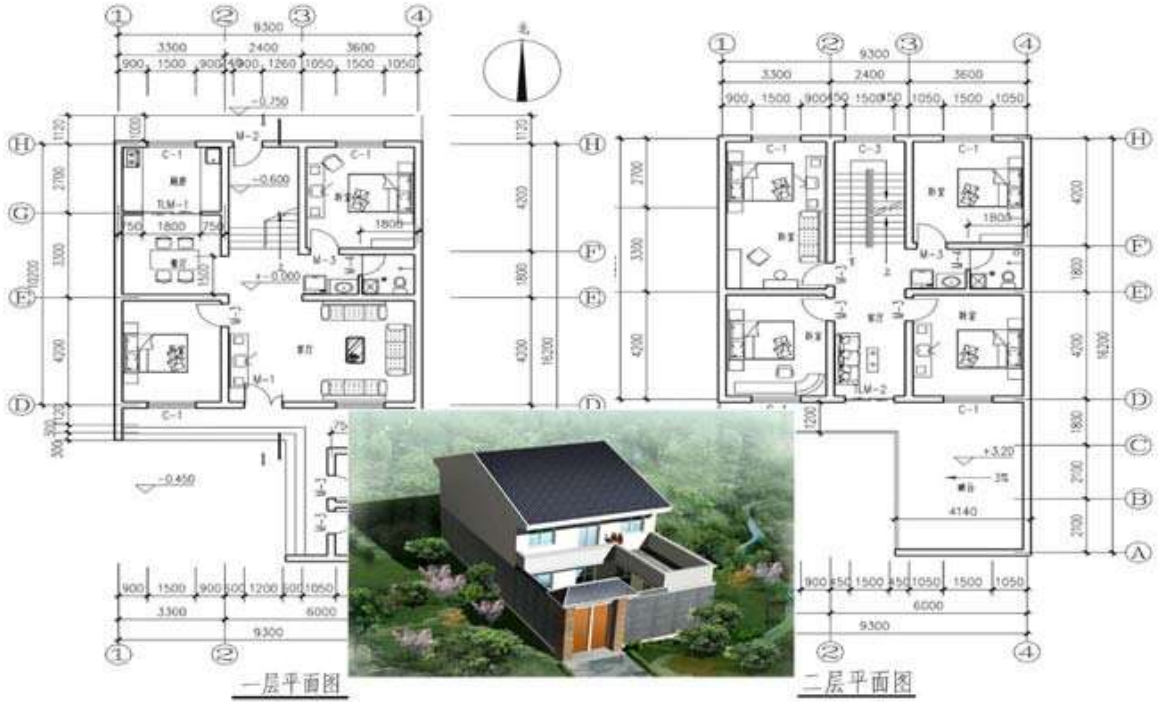
類別 vs. 物件

- Java 設計類別及類別之間的互動關係
- Java 應用程式執行時，JVM 根據類別定義建立物件，並處理物件之間互動產生之的狀態變化



類別的屬性和方法

- 類別如同藍圖，藍圖裡的重點主要有兩種
  - 屬性: 指房子的基本格局，裡面有什麼東西等等 (如三房兩廳, 有陽台, 有兩衛浴)
  - 方法: 房子的功能如何，如可以煮飯 洗澡

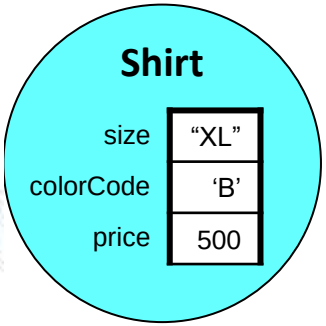


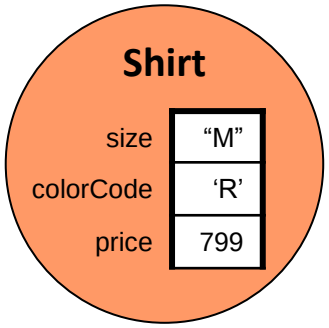
物件導向設計第一步

- 建立類別 (Build Class)
  - 為物件建立類別 (藍圖)
  - 定義類別名稱，如Shirt

- 設計類別 (Design Class)
  - 視覺化所設計的類別
  - 設定屬性，如ShritID，Size
  - 設定方法，如DisplayInformation.

Shirt
+shirtID: int +colorCode: char +size: String +price: double +description : String
+Shirt (color: char, price: double, description: String)
+calculateShirtID() : int +displayInformation()



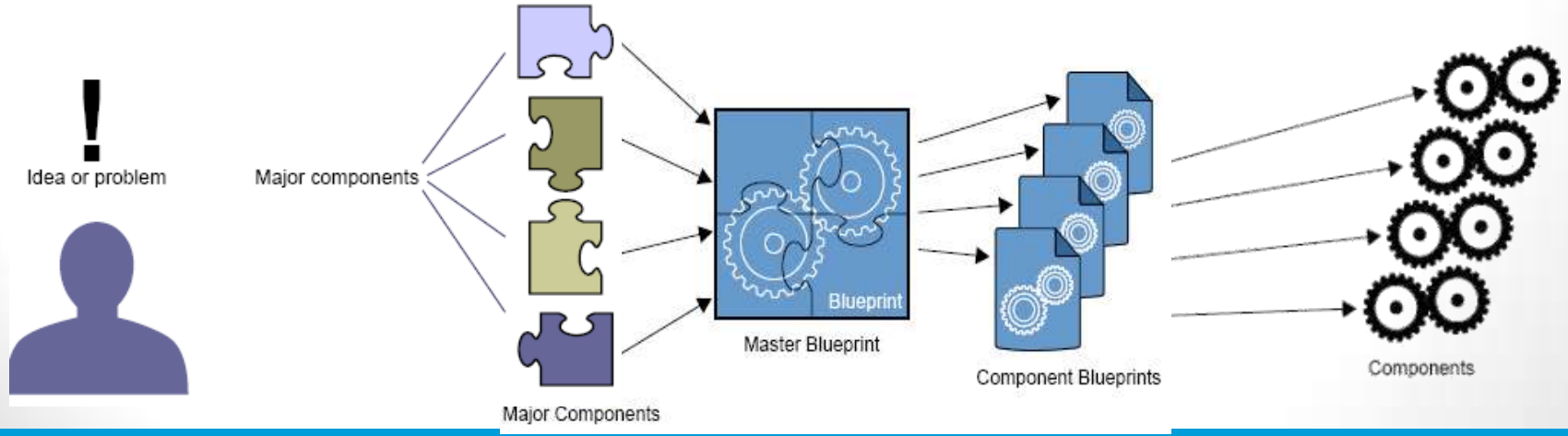


Content

- 類別與物件
- 物件導向開發
- Java程式結構

物件導向開發三階段

- Object Oriented Analysis：物件導向分析
- Object Oriented Design：物件導向設計
- Object Oriented Programming：物件導向程式開發



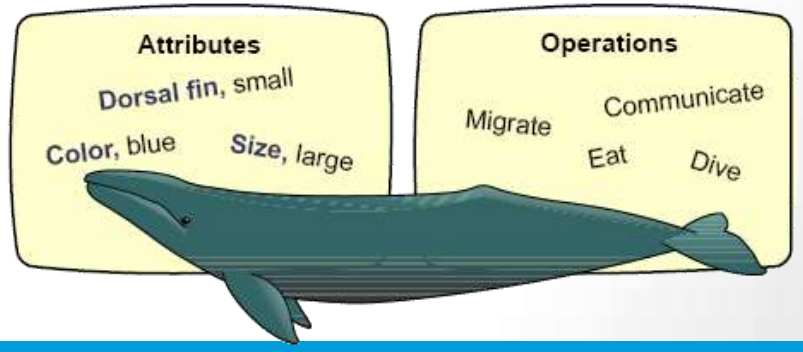
物件導向分析步驟

1. 辨識物件 (Identify Object)
2. 確認物件 (Recognized Object)
3. 物件之屬性及行為 (Identify Object's Attribute & Operation)

辨識物件

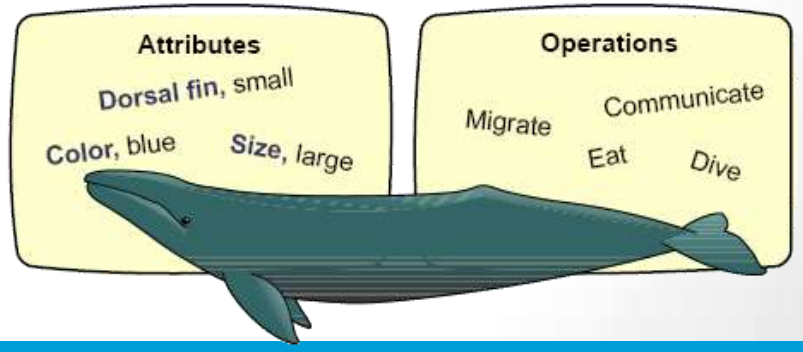
- 物件
  - 物件可能是實體或抽象概念，通常為名詞，如Account, Shirt
  - 物件下有兩個性質：物件屬性與物件方法
- 物件屬性代表其特徵。**屬性**通常也為名詞，如color, size
- 物件方法指物件可以做的事情或行為，**方法**通常為動詞或名詞與動詞的組合

  例如display, submit order

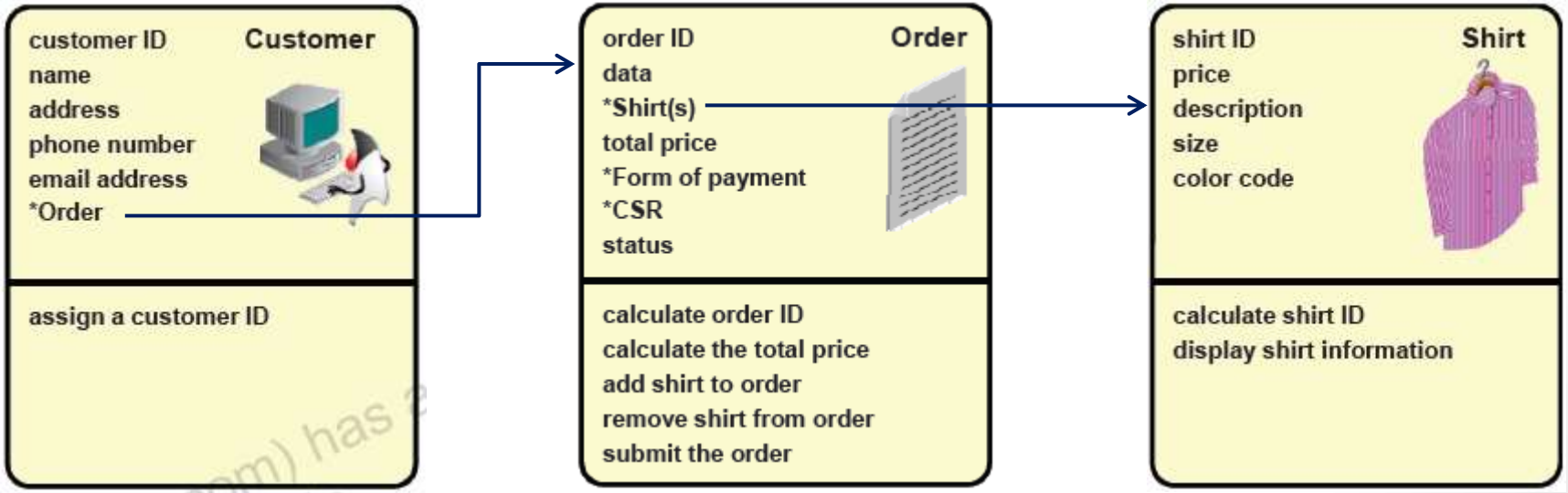


確認物件

- 與問題領域的相關性
  - 物件是否存於問題領域邊界之內
  - 物件是否為解決問題所必需的
  - 物件是否為用戶和系統之間的相互作用的一部分



物件之屬性及行為


customer ID
Customer
name
address
phone number
email address
*Order
assign a customer ID
order ID
Order
data
*Shirt(s)
total price
*Form of payment
*CSR
status
calculate order ID
calculate the total price
add shirt to order
remove shirt from order
submit the order
shirt ID
Shirt
price
description
size
color code
calculate shirt ID
display shirt information

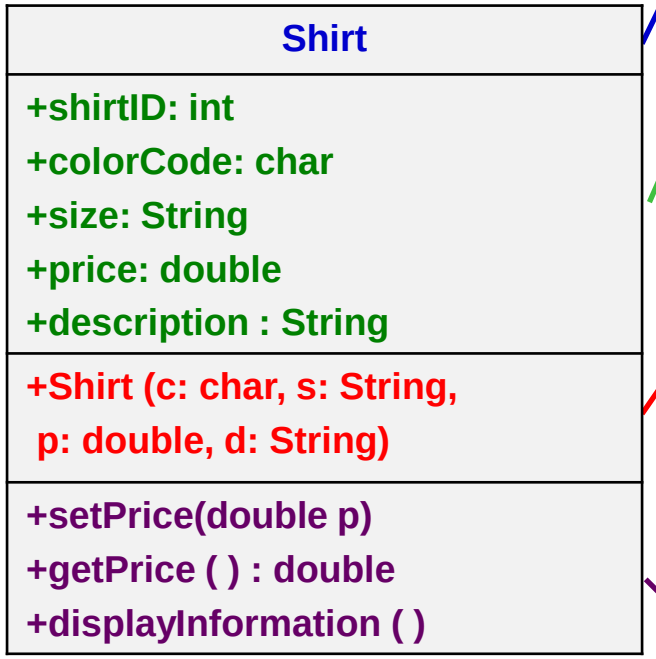
Content

- 類別與物件
- 物件導向開發
- Java程式結構

Java 程式結構

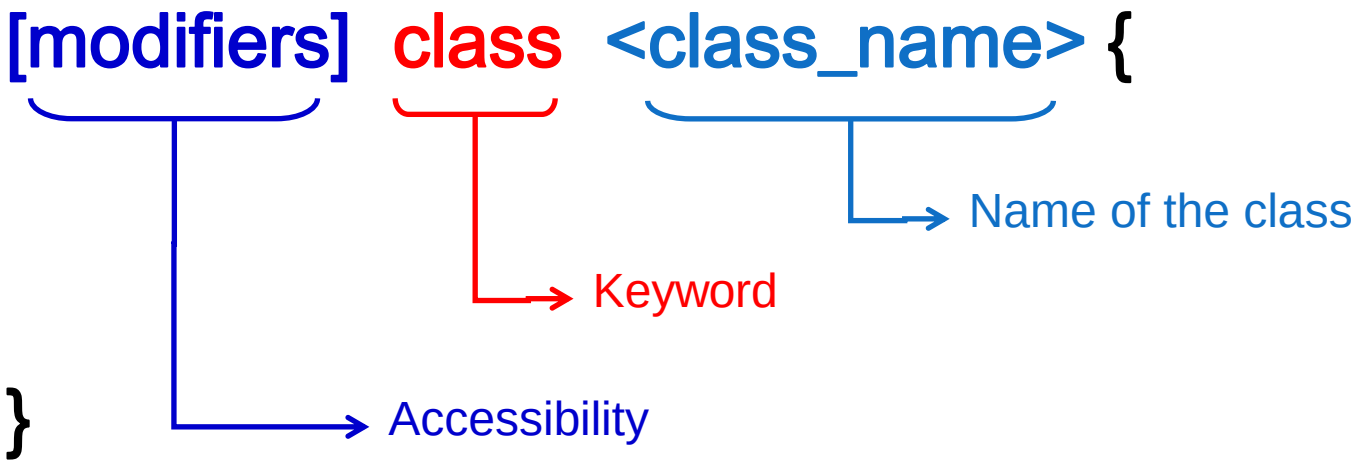
- 類別宣告 Class
- 屬性宣告 Attributes
- 方法宣告 Methods
- 主類別與程式進入點

Java 程式結構



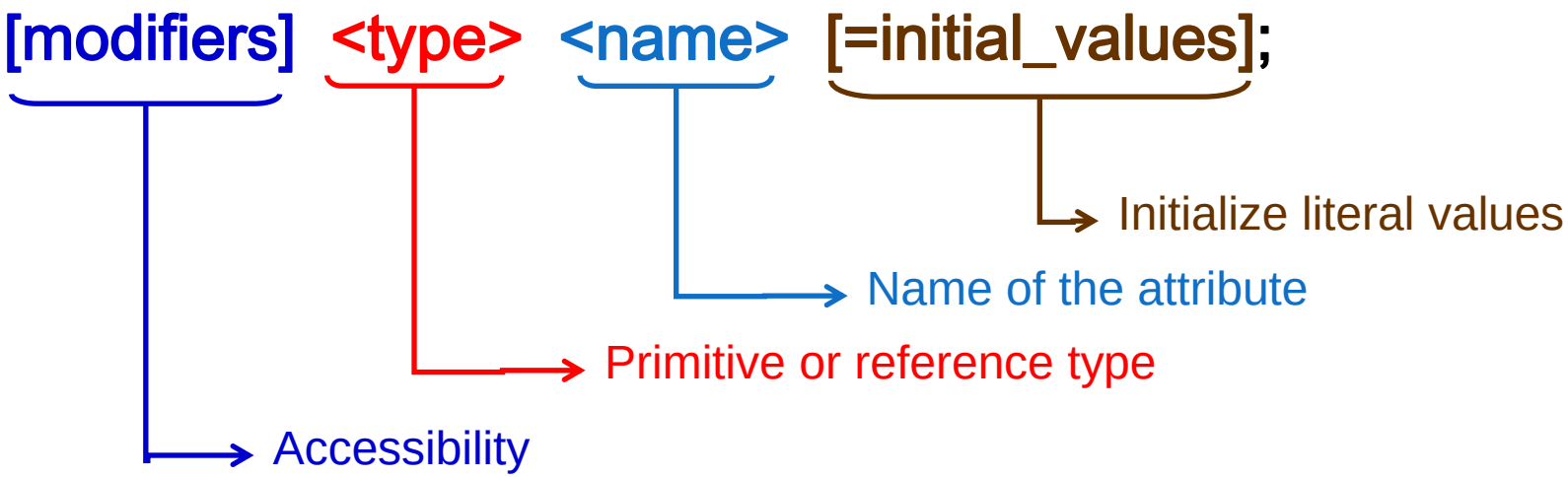

```
01  public class Shirt {
02
03      public int shirtID = 0;                          物件屬性
04      public char colorCode = 'G';                     (欄位)
05      public String size = "XL" ;
06      public double price = 299.00;
07      public String description = "Polo Shirt";
08
09      public Shirt(char color, String size,
10                   double price, String desc) {
11          colorCode = c;
12          size = s;
13          price = p;
14          description = d;
15      }                                                建構子
16
17      public void setPrice(double p) {                 物件方法
18          price = p;                                   (操作)
19      }
20      public double getPrice( ) {
21          return price;
22      }
23      public void displayInformation() {
24          System.out.println("Shirt ID:" + shirtID);
25          System.out.println("Color:" + colorCode);
26          System.out.println("Size:" + size);
27          System.out.println("Price:" + price);
28      }
29
30  }
```

Class Declaration 類別宣告



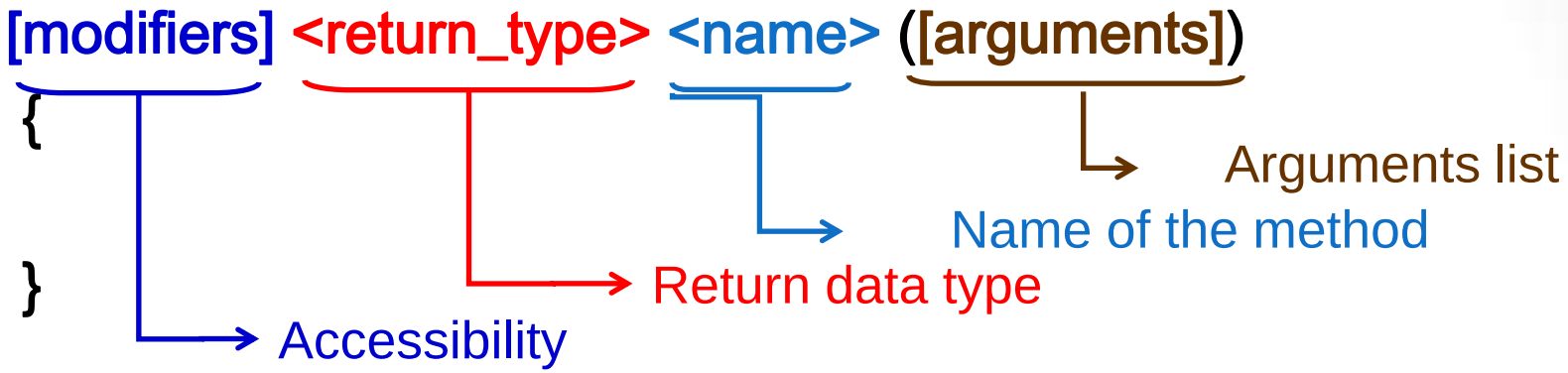
- 一個Java檔案,可包含一個以上的類別宣告。
- 多個類別宣告在一個Java檔案中,只能有一個public的類別。
- Modifier為public的類別, 需存在class_name.java的檔案中。

Attribute Declaration 屬性宣告



- 程式中用變數來存放需使用到的資料。
- 變數定義在class body內用來表示屬性。
- 存取權限modifier有public, protected, (default), private 。

Method Declaration 方法宣告



- 存取權限有 public, protected, (default), private。
- 傳回值型態需與方法區段內return的資料型態相符。
- 方法如沒有傳回值，傳回值型態為void。
- 參數列 (Arguments List)
  - 格式為Type Name。
  - 可有0~N個, 超過一個時，用逗點隔開。

主類別與程式進入點

- 主類別
  - 每個Java應用程式都需要一個主類別，作為程式的進入點，也稱為應用程式的啟始類別
  - Java SE的應用程式中，主類別會包含 main()方法
    - 主類別中的main方法，建立所需其他物件
    - 利用物件之間的互動來完成工作
- 類別在下列情況會加上main()方法 (程式進入點)
  - 用來開始應用程式 (應用程式的起始類別)
  - 執行程式來測試類別

```
01  public class OrderEntry{
02    public static void main (String[] args) {
03      Order order = new Order();
04      Shirt s1 = new Shirt(……);
05    }
06  }
```

```
01  public class Order {
02        ......
03  }
```

```
01  public class Shirt {
02        ......
03  }
```

主類別與程式進入點

```
public static void  main  (String args[]) {

}
```

- 符合標準的main(), 才可被當作程式的進入點
- main() 方法中陳述句
  - 建立所需之其他物件
    - Greeting hello = new Greeting();
  - 呼叫物件的方法來互動
    - hello.greet();

Q & A