From Gossip@Openhome

Java Gossip: 擴充（extends）父類別

假設您先前已經撰寫了一些2D繪圖相關類別，現在您想要將之擴充為適用於3D繪圖，3D的觀念是2D的延伸，許多 2D 繪圖時所使用的功能都可以留下來加以擴充，在Java中您不用重寫所有的類別，您可以**「擴充」（extend）**原先已定義好的類別，增加新的定義。

Java中使用"extends"作為其擴充父類別的關鍵字，其實就相當於我們一般所常稱的**繼承（Inherit）**，只不過"extends"除了繼承之外，還有將繼承下來的類別予以新增定義的意思。

例如繪圖中最基本的「點」類別，您原先已定義好一個Point2D類別，您繼承它並將之擴充為Point3D類別，在擴充的關係中，稱被擴充的類別為**「父類別」（Parent class）**或**「基礎類別」（Base class）**，而擴充父類別的類別就稱之為**「子類別」（Child class）**或**「衍生類別」（Derived class）**，在擴充時您使用"extends" 關鍵字，並指定要被擴充的類別。

直接使用一個實際例子來說明好了：

- Point2D.java

```
public class Point2D {
    private int x, y;

    public Point2D() {}
    public Point2D(int x, int y) {this.x = x; this.y = y;}
    public int getX() {return x;}
    public int getY() {return y;}
}
```

- Point3D.java

```
public class Point3D extends Point2D { // 擴充Point2D類別
    private int z;  // 新增私用資料

    public Point3D() {
        super();
    }

    // 建構函式，同時指定呼叫父類別建構函式
    Point3D(int x, int y, int z) {
        super(x, y);
        this.z = z;
    }

    // 新增函式
    public int getZ() {return z;}
}
```

- UseExtend.java

```
public class UseExtend {
    public static void main(String[] args) {
        Point3D p1 = new Point3D(1, 3, 4);
        Point3D p2 = new Point3D();

        System.out.printf("p1: (%d, %d, %d) \n",
            p1.getX(), p1.getY(), p1.getZ());
        System.out.printf("p2: (%d, %d, %d) \n",
            p2.getX(), p2.getY(), p2.getZ());
    }
}
```

執行結果：

```
p1: (1, 3, 4)
p2: (0, 0, 0)
```

當您擴充某個類別時，該類別的所有public成員都可以在衍生類別中被呼叫使用，而private成員則不可以直接在衍生類別中被呼叫使用；在這個例子 中，Point2D中已經有x, y兩個成員，您新增z成員，而方法上您新增一個public的getZ()方法，而getX()與getY()直接繼承父類別中已定義的內容。

在擴充某個類別之後，您可以一併初始父類別的建構方法，以完成相對應的初始動作，這可以使用**super()**方法來達到，它表示呼叫基底類別的建構方法。

父類別的public成員可以直接在衍生類別中使用，而private成員則不行，private類別只限於定義它的類別來存取，如果想要與父類別的 private成員溝通，就只能透過父類別中繼承下來的public函式成員，例如上例中的getX()與getY()方法。

下面這個程式是另一個示範存取父類別private成員的方法：

- Point2D.java

```
public class Point2D {
    private int x, y;
```

```
    public Point2D() {x = 0; y = 0;}
    public Point2D(int x, int y) {this.x = x; this.y = y;}
    public int getX() {return x;}
    public int getY() {return y;}
    public void setX(int x) {this.x = x;}
    public void setY(int y) {this.y = y;}
}
```

- Point3D.java

```
public class Point3D extends Point2D { // 繼承Point2D類別
    private int z;  // 新增私用資料

    public Point3D() {
        z = 0;
    }

    // 建構函式，同時指定呼叫父類別建構函式
    Point3D(int x, int y, int z) {
        super(x, y);
        this.z = z;
    }

    // 新增函式
    public int getZ() {return z;};
    public void setZ(int z) {this.z = z;}
}
```

- UseExtend.java

```
public class UseExtend {
    public static void main(String[] args) {
        Point3D p1 = new Point3D(1, 3, 4);
        Point3D p2 = new Point3D();

        System.out.printf("p1: (%d, %d, %d) \n",
            p1.getX(), p1.getY(), p1.getZ());

        p2.setX(2);
        p2.setY(4);
        p2.setZ(6);
        System.out.printf("p2: (%d, %d, %d) \n",
            p2.getX(), p2.getY(), p2.getZ());
    }
}
```

執行結果：

```
p1: (1, 3, 4)
p2: (2, 4, 6)
```

父類別的private成員並非無法在衍生類別中使用，而是必須看父類別本身是否提供有可繼承使用的管道來存取它們，就如同上例中的public成員 getX()、setX()等方法。