

Java程式設計進階

列舉及巢狀類別

鄭安翔

ansel_cheng@hotmail.com

課程大綱

1) 列舉 **Enum**

2) 巢狀類別

列舉 Enum

■ 問題

- ❑ 資料邏輯上的錯誤，編譯時期檢查不出來
- ❑ 用封裝可避免執行時期的錯誤，但維護不易
 - 列舉資料變動，程式即須修改
- ❑ 不易維護資料之對應關係

■ Java 5.0 加入列舉 enum

- ❑ 保證資料安全性
- ❑ 自動維護有限數量的列舉值對應關係
- ❑ 可使用 **Switch** 及 **For Each** 結構處裡列舉資料

沒有列舉 Enum 之前

```
public class Employee {  
    private String name;  
    private int id;  
    private String dept;  
  
    public void setName(String name) {  
        this.name = name;  
    }  
    public String getName() {  
        return name;  
    }  
  
    public void setId(int id) {  
        this.id = id;  
    }  
    public int getId() {  
        return id;  
    }  
  
    public void setDept(String dept) {  
        this.dept = dept;  
    }  
    public String getDept() {  
        return dept;  
    }  
}
```

```
public class PreEnumExample{  
    public static void main(String[] args) {  
        Employee e1 = new Employee();  
        e1.setName("Sean");  
        e1.setId(123);  
        e1.setDept("Sales");  
  
        Employee e2 = new Employee();  
        e2.setName("Peggy");  
        e2.setId(124);  
        e2.setDept("Hello World");  
    }  
}
```

資料型態沒有錯誤
但屬性值不合邏輯
編譯器檢查不出來

```
public class Employee {
    private String name;
    private int id;
    private String dept;
    public void setName(String name) {
        this.name = name;
    }
    public String getName() {
        return name;
    }

    public void setId(int id) {
        this.id = id;
    }
    public int getId() {
        return id;
    }
    public void setDept(String dept) {
        if (dept.equals("Engineering") ||
            dept.equals("Marketing") ||
            dept.equals("Sales") ||
            dept.equals("HR") ) {
            this.dept = dept;
        } else {
            System.out.println("無效部門名稱!")
        }
    }
    public String getDept() {
        return dept;
    }
}
```

```
public class PreEnumExample{
    public static void main(String[] args) {
        Employee e1 = new Employee();
        e1.setName("Sean");
        e1.setId(123);
        e1.setDept("Sales");

        Employee e2 = new Employee();
        e2.setName("Peggy");
        e2.setId(124);
        e2.setDept("Hello World");
    }
}
```

列舉資料變動
程式即需修改

```

public class Employee {
    private String name;
    private int id;
    private int dept;
    public void setName(String name) {
        this.name = name;
    }
    public String getName() {
        return name;
    }
    public void setId(int id) {
        this.id = id;
    }
    public int getId() {
        return id;
    }
    public void setDept(int dept) {
        if(dept>0 && dept<5){ this.dept = dept; }
    }
    public String getDept() {
        String deptStr="";
        switch(dept){
            case 1: deptStr="Engn";
            case 2: deptStr="Mark";
            case 3: deptStr="Sales";
            case 4: deptStr="HR"; break;
        }
        return deptStr;
    }
}

```

列舉資料變動
程式即需修改

```

public class Department {
    public static final int ENGINEERING = 1;
    public static final int MARKETING = 2;
    public static final int SALES = 3;
    public static final int HR = 4;
}

```

```

public class PreEnumExample{
    public static void main(String[] args) {
        Employee e1 = new Employee();
        e1.setName("Sean");
        e1.setId(123);
        e1.setDept(Department.SALES);

        Employee e2 = new Employee();
        e2.setName("Peggy");
        e2.setId(124);
        e2.setDept(-3);
    }
}

```

資料型態沒有錯誤
但屬性值不合邏輯
編譯器檢查不出來

列舉型別

■ 列舉型別語法

修飾字 **enum** 列舉型別名稱 {
 常數名稱一, 常數名稱二, 常數名稱三 ...
};

■ 列舉型別定義

- 可以獨立定義於一個原始檔內
- 或者在某個類別內定義 (結尾加分號)
- 不可以定義於方法內

■ 列舉型別本身就是類別，只是JVM做了一些處理，因此在使用列舉型別時，直接當類別看待

範例 - Enum

```
public class Employee {  
    private String name;  
    private int id;  
    private Department dept;  
  
    public void setName(String name) {  
        this.name = name;  
    }  
    public String getName() {  
        return name;  
    }  
    public void setId(int id) {  
        this.id = id;  
    }  
    public int getId() {  
        return id;  
    }  
    public void setDept ( Department dept) {  
        this.dept = dept;  
    }  
    public Department getDept() {  
        return dept;  
    }  
}
```

傳入列舉
型別參數

傳回值為
列舉型別

```
public enum Department {  
    ENGINEER,  
    MARKETING,  
    SALES,  
    HR  
}
```

元素不用雙引號
結尾不用分號

```
public class EnumExample1 {  
    public static void main(String[] args) {  
        Employee e1 = new Employee();  
        e1.setName("Sean");  
        e1.setId(123);  
        e1.setDept(Department.ENGINEER);  
  
        Employee e2 = new Employee();  
        e2.setName("Peggy");  
        e2.setId(124);  
        e2.setDept("HR");  
    }  
}
```

指定列舉
常數

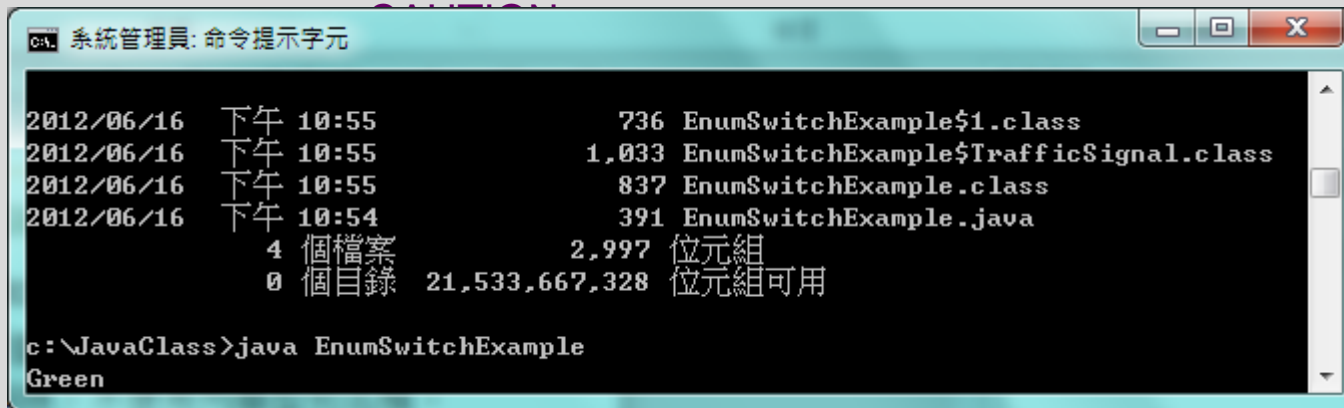
編譯時期
的錯誤

列舉型別和 switch 敘述

- 列舉型別和 switch 敘述合用時
 - switch 敘述的鍵值，也可以是列舉型別的變數。
 - 在使用case 標籤時，不使用列舉型別名稱。

```
public class EnumSwitchExample {  
    public enum TrafficSignal {STOP, CAUTION, GO};  
    public static void main(String[] args) {  
        TrafficSignal theLight = TrafficSignal.GO;  
        switch (theLight){  
            case STOP:  
                System.out.println("Red");  
                break;
```

不使用
TrafficSignal.STOP



```
c:\JavaClass>dir  
2012/06/16 下午 10:55              736 EnumSwitchExample$1.class  
2012/06/16 下午 10:55             1,033 EnumSwitchExample$TrafficSignal.class  
2012/06/16 下午 10:55              837 EnumSwitchExample.class  
2012/06/16 下午 10:54              391 EnumSwitchExample.java  
               4 個檔案             2,997 位元組  
               0 個目錄            21,533,667,328 位元組可用  
  
c:\JavaClass>java EnumSwitchExample  
Green
```

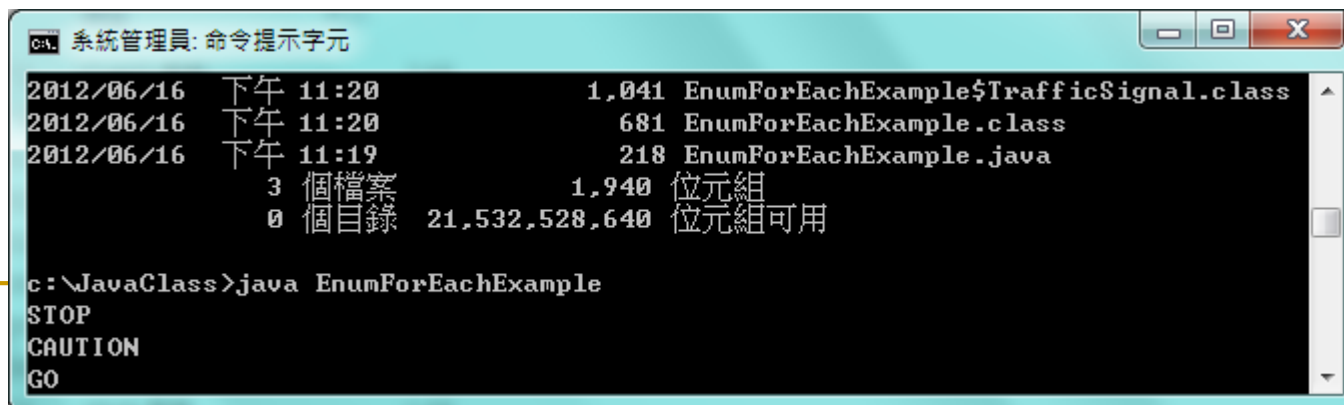
```
}
```

列舉型別和 For Each 迴圈

- 列舉型別和 For Each 迴圈合用時
 - 列舉型別提供一個static方法values()，回傳一個包含所有列舉值的陣列

```
public class EnumForEachExample {  
    public enum TrafficSignal {STOP, CAUTION, GO};  
  
    public static void main(String[] args) {  
        for(TrafficSignal t : TrafficSignal.values()){  
            System.out.println(t);  
        }  
    }  
}
```

傳回包含所有列舉值的
TrafficSignal[]



```
系統管理員: 命令提示字元  
2012/06/16 下午 11:20 1,041 EnumForEachExample$TrafficSignal.class  
2012/06/16 下午 11:20 681 EnumForEachExample.class  
2012/06/16 下午 11:19 218 EnumForEachExample.java  
3 個檔案 1,940 位元組  
0 個目錄 21,532,528,640 位元組可用  
  
c:\JavaClass>java EnumForEachExample  
STOP  
CAUTION  
GO
```

列舉型別特性

- 列舉型別本身是類別，繼承`java.lang.Enum`。
 - 實作`java.io.Serializable`及 `java.util.Comparable`
 - 所有列舉值皆為`public static final`
 - 可以使用`!=`、`==`或`equals()`測試是否相等
 - 不可以使用`>`、`>=`、`<`、`<=`運算子。
 - 列舉型別的值轉換成字串時，會轉換成和值的名稱相同的字串。

列舉宣告屬性、方法及建構子


- 列舉可以宣告屬性、方法及建構子
 - 列舉值需先宣告，才能宣告屬性、方法及建構子
 - 列舉值宣告時，可帶參數列，參數列需與列舉的建構式對應，通常用來設定列舉的屬性值
 - 列舉的建構式需為`private`

有欄位、方法及建構子的列舉型別

```
public enum TrafficSignal {  
  
    STOP("red"), CAUTION("yellow"), GO("green");  
    private final String light;  
    private TrafficSignal(String t){  
        light = t;  
    }  
    public String format(String message){  
        return message + " " + light;  
    }  
}
```

列舉值參數列
對應其建構式,
用來設定列舉
之屬性值

```
public class TrafficSignalTest {  
    public static void main(String[] args) {  
        System.out.println(TrafficSignal.GO.format("This light is "));  
        System.out.println(TrafficSignal.STOP.format("That light is "));  
    }  
}
```



系統管理員: 命令提示字元

日期時間	檔案名稱	大小 (位元組)
2014/01/08 下午 07:56	TrafficSignal.class	1,292
2014/01/08 下午 07:55	TrafficSignal.java	249
2014/01/08 下午 07:56	TrafficSignalTest.class	615
2014/01/08 下午 07:55	TrafficSignalTest.java	218

7 個檔案 18,755,661 位元組
6 個目錄 20,210,343,936 位元組可用

```
C:\JavaClass>java TrafficSignalTest  
This light is green  
That light is red
```

課程大綱

1) 列舉 Enum

2) 巢狀類別

- **成員位置巢狀類別**
 - **Non-Static巢狀類別**
 - **Static 巢狀類別**
- **區域式巢狀類別**
- **匿名類別**

Nested class

■ 巢狀類別

- 定義在某個類別中的類別
- 用來輔助外部類別
 - 類別需伴隨另一個類別存在才有意義
 - **helper**類別
- 常用於**GUI(Graphical User Interface)**程式中

Nested class

■ 優點

- 巢狀類別可以直接存取外部類別的成員(屬性與方法)
- 另一層封裝：將耦合度高的輔助類別封裝在類別中
- 另一層類別階層：比套件更緊密的類別關係
- 提高程式的可讀性及維護性

巢狀類別分類

- 巢狀類別依其宣告位置及特性可分為四種：
 - 成員位置巢狀類別
 - **non-static**巢狀類別,又稱為內部類別(**Inner Class**)
 - **static**巢狀類別
 - 區域式巢狀類別 **Local Class**
 - 匿名類別 **Anonymous Class**

範例 – 內部類別

- 類別定義為外部類別(outer class)的成員
 - 有一般成員的特性: public, protected, <default>, private, final, abstract
- 編譯後,多產生類別檔
 - OuterClass\$Inner.class
 - OuterClass\$流水號.class

```
public class Car {  
    private boolean running = false;  
    private Engine engine = new Engine();  
    private class Engine {  
        public void start() {  
            running = true;  
        }  
    }  
    public void start() {  
        engine.start();  
    }  
}
```



系統管理員: 命令提示字元

```
c:\JavaClass>javac Car.java  
  
c:\JavaClass>dir  
磁碟區 C 中的磁碟是 OS  
磁碟區序號: 7278-9798  
  
c:\JavaClass 的目錄  
  
2013/06/15 上午 12:00 <DIR> .  
2013/06/15 上午 12:00 <DIR> ..  
2013/06/15 上午 12:00 160 Car$1.class  
2013/06/15 上午 12:00 476 Car$Engine.class  
2013/06/15 上午 12:00 528 Car.class  
2013/06/15 上午 12:00 279 Car.java  
4 個檔案 1,443 位元組  
2 個目錄 16,077,279,232 位元組可用  
  
c:\JavaClass>
```

Static巢狀類別

- Static巢狀類別為外部類別的類別成員
 - 擁有一般類別成員的特性: public, protected, <default>, private, final, abstract
- 編譯後,多產生一個類別檔
 - Outer\$Inner.class

```
public class Outer2 {  
    private static int size;  
  
    public static int getSize() {  
        return size;  
    }  
  
    public static class Inner2 {  
        public void incrSize() {  
            size++;  
        }  
    }  
}
```

區域式巢狀類別

■ 區域類別 Local Class

- 宣告在方法之中的巢狀類別
- 將區域類別物件參考傳回,只能視為父類別(Object),因為在方法外無法取得區域類別定義
- 存取包含它的方法中的區域變數,該變數必須是**final**
 - 區域變數只能在包含它的方法中使用
 - 區域式巢狀類別產生之物件生命可能比包含它的方法長
 - **Java 8**之後不再發生編譯錯誤,但方法中變數自動成為區域常數,稱為**effective final**

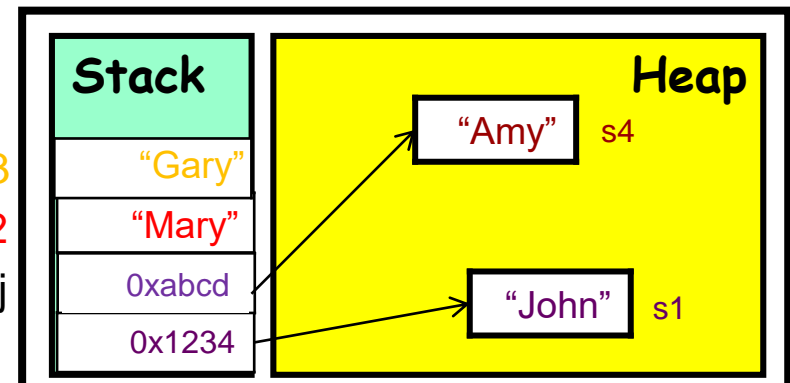
範例 – 區域類別

```
public class Outer3 {  
    private String s1= "John";  
  
    public Object getInner(){  
        String s2 = "Mary";  
        final String s3 = "Gary"  
        class Inner3 {  
            String s4 = "Amy";  
            public String toString() {  
                return "(" +  
                    s1 + "," +  
                    s2 + "," +  
                    s3 + "," +  
                    s4 + ")";  
            }  
        }  
        return new Inner3();  
    }  
}
```

```
public class TestOuter3 {  
  
    public static void main(String[] args){  
        Outer3 o = new Outer3();  
        Object obj = o.getInner();  
        System.out.println(obj.toString());  
    }  
}
```

getInner() [s3
 s2
main() [obj
 o

RAM



匿名類別 Anonymous class

- 對一些簡單而不會重複使用的類別,將類別命名工作交給Java編譯程式

- 語法

```
new 欲繼承之類別或實作之介面名稱() {
```

```
    /*....程式碼....*/
```

```
};
```

- Anonymous class 命名

- outerClass\$流水號.class

範例 – 匿名類別

```
public class Outer3 {  
    private String s1= "John";  
  
    public Object getInner(){  
        String s2 = "Mary";  
        final String s3 = "Gary";  
        class Inner3 {  
            String s4 = "Amy";  
            public String toString() {  
                return "(" +  
                    s1 + "," +  
                    // s2 + "," +  
                    s3 + "," +  
                    s4 + ")";  
            }  
        }  
        return new Inner3();  
    }  
}
```

```
public class Outer4 {  
    private String s1= "John";  
  
    public Object getInner(){  
        String s2 = "Mary";  
        final String s3 = "Gary";  
        return new Object(){  
            String s4 = "Amy";  
            public String toString() {  
                return "(" +  
                    s1 + "," +  
                    // s2 + "," +  
                    s3 + "," +  
                    s4 + ")";  
            }  
        };  
    }  
}
```

範例 – 匿名類別

```
public class Programmer {
```

```
    private Brain myBrain = new MyBrain();
```

```
    private class MyBrain extends Brain {  
        public void think(){  
            System.out.println("Java!");  
        }  
    }
```

```
    abstract class Brain {  
        abstract void think();  
    }
```

```
public class Programmer {
```

```
    private Brain myBrain = new Brain() {  
        public void think(){  
            System.out.println("Java!");  
        }  
    };
```

```
    abstract class Brain {  
        abstract void think();  
    }
```



系統管理員: 命令提示字元

```
C:\Java>dir  
磁碟區 C 中的磁碟是 WIN7_UHD  
磁碟區序號: A28C-E4BA  
  
C:\Java 的目錄  
  
2014/01/08 下午 08:59 <DIR> .  
2014/01/08 下午 08:59 <DIR> ..  
2014/01/08 下午 08:59 204 Brain.class  
2014/01/08 下午 08:59 524 Programmer$1.class  
2014/01/08 下午 08:59 324 Programmer.class  
2014/01/08 下午 08:58 199 Programmer.java  
4 個檔案 1,251 位元組  
2 個目錄 20,058,845,184 位元組可用
```