包裝類別與工具類別(1)

TYIC

包裝類別

雖然在 Java 中幾乎所有東西都是物件，但基本資料型別卻不是
這導致基本資料型別無法像物件一樣呼叫方法
所以出現了包裝類別(wrapper class)來解決這個問題
8 種基本資料型別對應了 8 種包裝類別，分別為：
Byte、Short、Character、Integer、
Long、Float、Double、Boolean
這些包裝類別皆位於 java.lang 套件內，所以可以直接使用

包裝類別

欲創建包裝類別，須呼叫包裝類別的公開靜態方法 "valueOf"
必有一個多載具有唯一參數，且為對應的基本資料型別
有些包裝類別有多載該方法，可能的參數有字串等
這個動作稱為裝箱(boxing)
而呼叫包裝類別的公開動態方法 "xxxValue"(xxx為基本資料型別)
將包裝類別變為基本資料型別
就被稱為拆箱(unboxing)

自動拆箱

包裝類別可以像基本資料型別一樣進行各式運算

```java
public class Main1 {
    public static void main(String[] args) {
        System.out.println(100 + Integer.valueOf(200));
        System.out.println(Integer.valueOf(100) / 200);
        System.out.println(Integer.valueOf(100) % Integer.valueOf(200));
    }
}
```

自動拆箱

這是因為編譯器會在包裝類別運算前呼叫

"xxxValue"(xxx為基本資料型別) 方法，稱為自動拆箱

```java
public class Main1 {
    public static void main(String[] args) {
        System.out.println(100 + Integer.valueOf(200).intValue());
        System.out.println(Integer.valueOf(100).intValue() / 200);
        System.out.println(Integer.valueOf(100).intValue() % Integer.valueOf(200).intValue());
    }
}
```

自動裝箱

將包裝類別賦值給基本資料型別的變數時也會自動拆箱

而將基本資料型別賦值給包裝類別的變數時則會自動裝箱

```java
public class Main2 {
    public static void main(String[] args) {
        final Integer TWO_HUNDRED = 100;
        System.out.println(add(100, TWO_HUNDRED));
    }

    public static Integer add(Integer a, Integer b) {
        return a + b;
    }
}
```

自動裝箱　　自動拆箱

```java
public class Main2 {
    public static void main(String[] args) {
        final Integer TWO_HUNDRED = Integer.valueOf(100);
        System.out.println(add(100, TWO_HUNDRED.intValue()));
    }

    public static Integer add(Integer a, int b) {
        return Integer.valueOf(a + b);
    }
}
```