目錄

轉義序列(Escape Sequence)

- `\'` 單引號
- `\"` 雙引號
- `\\` 反斜線
- `\n` 換行
- `\t` tab 鍵
- `\b` 倒退一格
- `\f` 換頁
- `\r` return 鍵(Enter 鍵)

編碼法

- `ASCII`
  - American Standard Code for Information Interchange
    (美國國家標準資訊交換碼)
    長度: 1 byte
- `Big5`
  - 正體中文(台灣, 香港常用)
    長度: 2 bytes
- `Unicode`
  - 萬國碼(Universal Code)
    (UTF-8, UTF-16) 長度: 2 bytes

(UTF-32) 長度: 4 bytes

基本資料形態

- Java 八大基本資料型別(Primitive Data Types)

形態名稱	大小	範圍	初始值
byte	8bits	byte i = 1;	0
short	16bits	short i = 1;	0
int	32bits	int i = 1;	0
long	64bits	long i = 1L	0L
float	32bits	int i = 1;	0.0F
double	64bits	long i = 1L	0.0(D)
boolean	1bit	boolean b = true;	false
char	16 bit / Unicode 格式	char c = 'A';	'\u0000'

區域變數 實體變數 靜態變數 常數(!)

- 區域變數 Local Variables
  - 宣告在 `方法內`
  - 又稱 `member variables` (成員變數), `attribute variables`, `暫時變數`, `自動變數`
  - 因為是暫時變數，所以每次運行後存在的記憶體位置都會不同(區域變數無法存活到下一個請求)
- 實體變數 Instance Variables
  - 宣告在 `方法外`, `類別內`, `沒有 static 修飾子`
  - 又稱為 `automatic`, `temporary`, `stack variables`, `屬性變數`
  - 實體變數有其 `持續性` ，缺點也是持續性
- 靜態變數 Class Variable
  - 又稱為類別變數
  - Java 會 `自動給初始值` ，不可在宣告後在指定

- 宣告在 `方法外`, `類別內`, `有 static 修飾子`
- 又稱 `golbal`

- 常數
  - `全大寫`, 分格用底線分開 `final MY_NAME = "Hello";`
  - 宣告後值不可變更

```
public class Hello{
  // 常數
  final double MATH_PI = 3.14;
  // 靜態變數
  static int classVariable = 0;
  // 實體變數
  int instanceVariables = 0;
  public static void main(String[] args){
    // 區域變數
    int localVariables = 0;
  }
}
```

算數運算子 二元運算子

- 算術運算子 (Arithmetic Operators)又稱為二元運算子
  - 若兩個運算元的位階不相等,則運算完後的回傳值會與 `位階高者` 相同
  - 若兩個運算元為基本型別,至少會轉換成 `int` 型別

運算子	用法	說明
`/`	a / b	5.0 / 2 = 2.5 5 / 2 = 2

運算子	用法	說明
`%`	a % b	7 % 2 = 1 9.6 % 3.5 = 2.6

遞增 遞減 指定 運算子 (!)

```
int a = 3;
System.out.println(a++); //3
System.out.println(a); //4
```

```
int a = 3;
System.out.println(++a); //4
System.out.println(a); //4
```

- 指定運算子
  - 右邊不能大於左邊 (位階)
  - 位階高低順序:
    - double > float > long > int > short > byte

```
double a = 0;
int b = 0
```

進制轉換

進制	開頭
二進制(Binary)	0b
十進制(Decimal)	
八進制(Octal)	0
十六進制(Hexadecimal)	0x

```
System.out.printf("%x\n",7);// 7      按16進制輸出
System.out.printf("%o\n",13);// 15    按8進制輸出
```

switch case(!)

- switch case 變數(n)只可為 整數 , 字元 ,不可為浮點數
  (JDK 7 以後,可以比對 字串)
- 若省略 break 敘述,則會執行下一個 case 中的敘述

```
public class test {
  public static void main(String[] args) {
    int a = 1, ans = 0;
    switch (a) {
    case 1:
      ans += 9;
    default:
      ans += 11;
    }
    System.out.println(ans); // 20
  }
}
```

```
public class test {
  public static void main(String[] args) {
    int a = 1, ans;
    switch (a) {
    case 1:
      ans = 9;
    default:
      ans = 11;
    }
    System.out.println(ans); // 11
  }
}
```

```java
public class test {
  public static void main(String[] args) {
    int a = 3, ans = 0;
    switch (a) {
    case 1:
      ans += 9;
    default:
      ans += 11;
    }
    System.out.println(ans); // 11
  }
}
```

三種迴圈(!)

- 如果執行前已可確定次數,通常會使用 for 迴圈
  - for 迴圈第一行即計次，`用了 continue 也會先計次`
- 如果執行前不確定次數,通常會使用 while 或 do...while 迴圈
- `break` 的作用是`跳離`迴圈，且只能在`迴圈`及 `switch` 中用
- `continue` 的功能是跳過 `continue` 以下的敘述，回到迴圈的起始點

```java
int sum = 0;
for (int count = 1; count <= 10; count++) {
  sum += count;
}
```

```java
int sum = 0;
int count = 1;
while (count <= 10) {
  sum += count;
  count++;
}
```

```
int sum = 0;
int count = 1;
do {
  sum += count;
  count++;
} while(count <= 10);
```

Overloading 覆載(!)

- 讓我們可以用 `統一的方法名稱` 來呼叫相同功能的方法
  - 以 `參數`，`參數型態` 區分，與回傳值無關
- 減輕命名壓力

```
public void println(int i)
public void println(float f)
public void println(String s)
```

```
public void method(int i)
public int method(int i)    //重覆宣告
public String method(int i) //重覆宣告
```

參考鏈接:

- [[重構技巧]-Replace Temp with Query]