**Java物件**

**類別型態**

Java除了基本型態還有類別型態，java撰寫程式幾乎都在使用物件，產生物件必須先定義類別，類別是物件的設計圖，物件是類別的實例。

**定義類別**

類別定義使用**class關鍵字**，要建立物件使用**new 關鍵字。**

Ex:

class Clothes{ --------------------🡪定義類別

String color;

Char size;

}

new Clothes();

Clothes c1;

new Clothes()為建立Clothes物件，Clothes c1這叫參考名稱、參考變數，或直接叫參考。也可以用**=**指定

Ex:

Clothes c1=new Clothes();

**建構式**

如果想在建立物件時，一併進行初始化，像是指定資料值時，可建立建構式，它是與類別名稱同名的方法

Ex:

class Clothes{

String color;

Char size;

Clothes(String color, Char size){

this.color=color;

this.size=size;

}

}

**Clothes sun=new Clothes(“red”,”s”);**

注意建構式裡因為color參數與資料成員color同名，所以得用**this**表示，要將color參數指定給這個物件的color成員。

**物件指定與相等性**

基本型態

=用於基本型態是將值複製給變數，==是比較兩個變數儲存的值是否相同

Ex:

Int a=10;

Int b=10;

Int c=a;

System.out.print(a==b);🡪true

System.out.print(a==c);🡪true

類別型態

=用於指定參考名稱參考某物件，而==是用於參考名稱是否參考同一物件

Ex:

clothes c1=new clothes(2,'s');

clothes c2=new clothes(2,'h');

System.out.println(c1==c2);🡪false(c1和c2分別參考不一樣的物件)

c1=c2;

System.out.println(c1==c2);🡪true

(c1名稱參考到c2物件所以為true)

**裝箱**

當使用collection，需要放入整數，但collection需要持有都是物件，這時可將基本型態裝箱成物件

Ex:

Int a=10;

Integer wrapper=new Integer(a);

**自動裝箱、拆箱**

編譯器會自動判斷是否能進行自動裝拆箱(編譯器蜜糖)

Integer wrapper=10; //自動裝箱

Int foo=wrapper; //自動拆箱

**Null**

Integer i=null;

Int j=i;

可編譯但執行過程會錯誤，編譯器會將之展開為:

Object localobject=null;

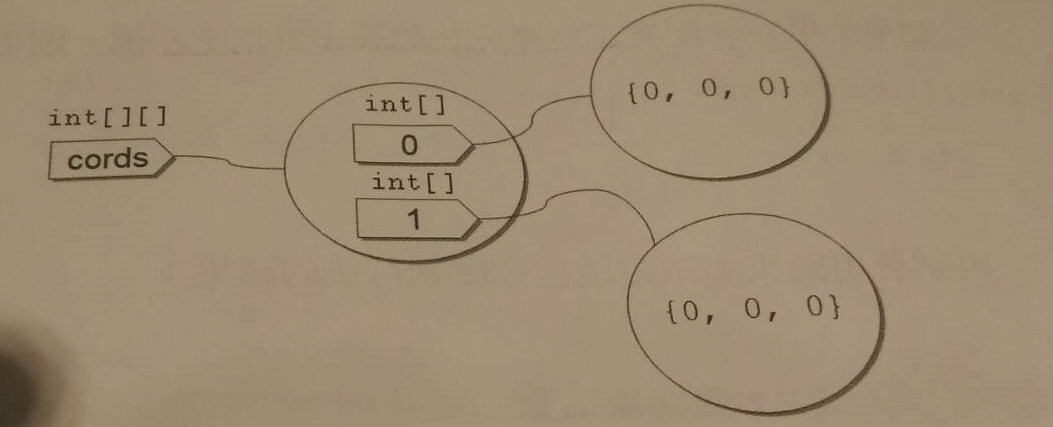
Int i= localobject.intValue();

因為null是一種特殊物件，任何類別宣告的參考名稱都可參考至null，表示該名稱沒有參考任何實體物件，由於i並沒有參考到任何物件，所以無法使用intValue方法。編譯器蜜糖提供方便性但隱藏細節得小心。

**陣列物件**

陣列在java就是物件，具有索引的資料結構，使用int[]宣告，會在記憶體分配長度為X的int連續空間。代表建立物件的類別定義由jvm動態產生。

多維陣列基本上都是由一維陣列模擬



**陣列複製**

Int[] scores1={1,2,3,4};

Int[] scores2=scores1;

上述不是複製陣列只是將s1參考的陣列也給s2參考。要達到複製陣列作法是另行建立新陣列再用逐一將s1元素指定給s2，也可此用system.arraycopy()

Int[] scores1={1,2,3,4};

Int[] scores2=new int[scores1.length];

For(int i=0;i<scores1.length;i++){

Scores2[i]=scores1[i];

}

Int[] scores1={1,2,3,4};

Int[] scores2=new int[scores1.length];

System.arraycopy(scores1,0,scores2,0,scores1.length);