



第八章

物件導向程式設計

8-3 方法多載(Method overloading)



【定義】是指在同一個類別中，可以使用「多個相同名稱」的「方法」。

【規定】方法中的「參數串列」必須要符合以下規定：

1. 「資料型態」不可以相同。

多載 1	public void BMI(int h,int w) {..... }
多載 2	public void BMI(float h, float w) {.....}

【說明】

- (1)在「多載1」中BMI中的兩個參數之資料型態皆為「int」。
- (2)在「多載2」中BMI中的兩個參數之資料型態皆為「float」。

2. 「個數」必須不同。

多載 1	public void AddNum(int a,int b) {..... }
多載 2	public void AddNum(int a, int b, int c) {.....}

【說明】

- (1)在「多載1」中BMI中有「兩個參數a,b」。
- (2)在「多載2」中BMI中有「三個參數a,b,c」。

3. 「順序」也必須不同。

多載 1	public void AddNum(int a,int b, int sum) {..... }
多載 2	public void AddNum(int sum ,int a, int b) {.....}

【說明】

- (1)在「多載1」中BMI中的三個參數順序為「a,b,sum」。
- (2)在「多載2」中BMI中的三個參數順序為「sum,a,b」。

【實例】請利用「方法多載」的方式，分別計算兩個數及三個數之和。

行號	程式檔名：ch8-3/src/AddNumClass.java
01	<pre>public class AddNumClass {</pre>
02	<pre> public static void main(String[] args) {</pre>
03	<pre> //主程式</pre>
04	<pre> int a=10;int b=20;int c=30;</pre>
05	<pre> int N2=MyAdd(a,b); //呼叫MyAdd多載方法1</pre>
06	<pre> int N3=MyAdd(a,b,c); //呼叫MyAdd多載方法2</pre>
07	<pre> System.out.println("二數之和=" + N2);</pre>
08	<pre> System.out.println("三數之和=" + N3);</pre>
09	<pre>}</pre>
10	<pre>//MyAdd多載方法1</pre>
11	<pre> static int MyAdd(int x,int y)</pre>
12	<pre> {</pre>
13	<pre> int sum=0;</pre>
14	<pre> sum = x + y;</pre>
15	<pre> return sum; //回傳結果到主程式</pre>
16	<pre>}</pre>
17	<pre>//MyAdd多載方法2</pre>
18	<pre> static int MyAdd(int x,int y,int z)</pre>
19	<pre> {</pre>
20	<pre> int sum=0;</pre>
21	<pre> sum = x + y + z;</pre>
22	<pre> return sum; //回傳結果到主程式</pre>
23	<pre>}</pre>
24	<pre>}</pre>

【執行結果】

二數之和=30

三數之和=60





8-4 類別中「方法」的呼叫

基本上，主程式想要呼叫類別中的「方法」時，常見有以下兩種方法：

1. 沒有傳遞參數的方法呼叫
2. 具有傳遞參數的方法呼叫

8-4.1 沒有傳遞參數的方法呼叫

【定義】主程式呼叫類別中的「方法」時，並沒有傳遞參數給「方法」。

【語法】

(一) 主程式呼叫副程式的語法

```
public static void main(String[] args)
{
    類別名稱 物件變數名稱=new 類別名稱();
    物件變數名稱.方法名稱();
}
```

(二) 副程式的語法

```
public class 類別名稱
{
    public void 方法名稱()
    {
        .....
    }
}
```

【實作】

請設計一個主程式呼叫「方法」，如果成功的話，顯示
「Call MyClass_Method() ok! 」



【程式】

步驟一：先建立「MyClass」類別

行號	程式檔名：ch8-4A/src/MyClass.java
01	<code>public class MyClass //定義MyClass類別</code>
02	<code>{</code>
03	<code> public void Method() //定義Method方法</code>
04	<code> {</code>
05	<code> System.out.println("Call MyClass_Method() ok!");</code>
06	<code> }</code>
07	<code>}</code>

步驟二：再建立「ch8_4A」主類別及主程式

行號	程式檔名：ch8-4A/src/ch8_4A.java
01	<code>public class ch8_4A { //主類別</code>
02	<code> public static void main(String[] args) {</code>
03	<code> //主程式</code>
04	<code> MyClass MyClass1 = new MyClass(); //建立MyClass1物件屬於MyClass類別</code>
05	<code> MyClass1.Method(); //呼叫MyClass類別中的「方法」</code>
06	<code> }</code>
07	<code>}</code>

【注意】主程式呼叫「方法」時，不一定要傳遞參數，如上面的例子中，主程式中的「MyClass」類別中的Method()方法時，並沒有參數的傳遞。

8-4.2 具有傳遞參數的方法



【定義】主程式呼叫「方法」的同時，「主程式」會傳遞參數給「方法」。

【優點】提高副程式的實用性與彈性。

【語法】

(一) 主程式呼叫「方法」的語法

```
public static void main(String[] args)
{
    類別名稱 物件變數名稱=new 類別名稱();
    物件變數名稱.方法名稱(參數 1,參數 2,... );
}
```

(二) 定義「方法」的語法

```
public class 類別名稱
{
    public void 方法名稱(資料型態 參數 1, 資料型態 參數 2,...)
    {.....}
}
```

【實例】請設計一個主程式呼叫一支副程式時，將主程式的實際參數傳遞給副程式的形式參數，並且副程式計算形式參數的總合。

【撰寫程式】

步驟一：先建立「MyClass」類別

行號	程式檔名：ch8-4B/src/MyClass.java
01	<code>public class MyClass //定義MyClass類別</code>
02	<code>{</code>
03	<code> public void MyAdd(int x,int y) //定義方法成員</code>
04	<code> {</code>
05	<code> int sum=0;</code>
06	<code> sum = x + y;</code>
07	<code> System.out.println("x+y=" + sum);</code>
08	<code> }</code>
09	<code>}</code>

步驟二：再建立「ch8_4B」主類別及主程式



行號	程式檔名：ch8-4B/src/ch8_4B.java
01	<code>public class ch8_4B { //主類別</code>
02	<code> public static void main(String[] args) {</code>
03	<code> //主程式</code>
04	<code> int a=10;int b=20;</code>
05	<code> MyClass MyClass2 = new MyClass(); //建立MyClass2物件屬於MyClass類別</code>
06	<code> MyClass2.MyAdd(a,b); //呼叫MyClass類別中的「方法」</code>
07	<code> }</code>
08	<code>}</code>

【執行結果】

x+y=30

【註】即使x與y參數的資料型態相同，也不可以合併宣告。否則會產生錯誤。

8-5 類別方法來傳值呼叫(Call By Value)



【引言】

在第六章中，我們了解學會如何利用結構化程式設計的方式，來進行「傳值呼叫」，而在本單元中，我們將利用「物件導向」的方式來進行，亦即利用類別方法來傳值呼叫(Call By Value)。其「運作原理」與第六章介紹的概念相同。

【語法】



(一) 主程式呼叫「方法」的語法

```
public static void main(String[] args)
{
    類別名稱 物件變數名稱=new 類別名稱();
    物件變數名稱.方法名稱(參數 1,參數 2,...);
}
```

(二) 定義「方法」的語法

【說明】其作法與前之單元介紹的「具有傳遞參數的方法」相同。



【實例】傳值呼叫

【撰寫程式】

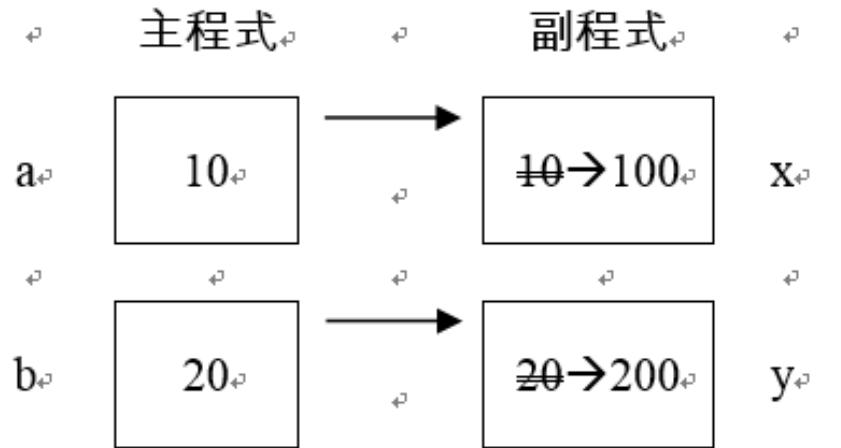
步驟一：先建立「MyClass」類別

行號	程式檔名：ch8-5/src/MyClass.java
01	public class MyClass //定義MyClass類別
02	{
03	public void CallByValue(int x, int y) //定義方法成員
04	{
05	x = 100;
06	y = 200;
07	}
08	}

步驟二：再建立「ch8_6」主類別及主程式

行號	程式檔名：ch8-5/src/ch8_5.java
01	<code>public class ch8_5 { //主類別</code>
02	<code> public static void main(String[] args) {</code>
03	<code> //主程式</code>
04	<code> int a=10;int b=20;</code>
05	<code> //建立MyClass_Value物件屬於MyClass類別</code>
06	<code> MyClass MyClass_Value = new MyClass();</code>
07	<code> System.out.println("==呼叫前==");</code>
08	<code> System.out.println("a=" + a + " " + "b=" + b);</code>
09	<code> //呼叫MyClass類別中的「方法」</code>
10	<code> MyClass_Value.CallByValue(a, b);</code>
11	<code> System.out.println("==呼叫後==");</code>
12	<code> System.out.println("a=" + a + " " + "b=" + b);</code>
13	<code>}</code>
14	<code>}</code>

【執行過程】



【說明】當主程式呼叫方法時，實際參數a所佔用的記憶體位址中的內容(值)會傳遞給方法中的形式參數x，而實際參數b傳送給形式參數y，因為是傳值呼叫，所以主程式的實際參數與方法的形式參數不會佔用相同的記憶體位址。因此，方法中的形式參數值改變，也不會影響到主程式中的實際參數值。

【執行結果】

```
====呼叫之前====
```

```
a=10  b=20
```

```
====呼叫之後====
```

```
a=10  b=20
```

8-6 參考呼叫(Call By Reference)



【引言】

在第六章中，我們了解學會如何利用結構化程式設計的方式，來進行「傳值呼叫」，而在本單元中，我們將利用「物件導向」的方式來進行，亦即利用類別方法來參考呼叫(Call By Reference)。其「運作原理」與第六章介紹的概念相同。

(一) 主程式呼叫「方法」的語法

```
public static void main(String[] args)
{
    類別名稱 物件變數名稱=new 類別名稱();
    物件變數名稱.方法名稱(陣列名稱 1, 陣列名稱 2,⋯);
}
```

(二) 定義「方法」的語法

```
public class 類別名稱
{
    public void 方法名稱(資料型態 1 陣列名稱 1, 資料型態 2 陣列名稱 2,⋯)
    {.....}
}
```



【實例】參考呼叫

【撰寫程式】

步驟一：先建立「MyClass」類別

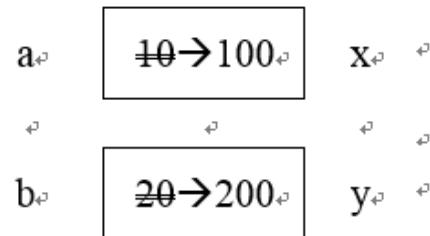
行號	程式檔名：ch8-6/src/MyClass.java
01	public class MyClass //定義MyClass類別
02	{
03	public void CallByRef(int x[], int y[]) //定義方法成員
04	{
05	x[0] = 100;
06	y[0] = 200;
07	}
08	}

步驟二：再建立「ch8_6」主類別及主程式

行號	程式檔名：ch8-6/src/ch8_6.java
01	<code>public class ch8_6 { //主類別</code>
02	<code> public static void main(String[] args) {</code>
03	<code> //主程式</code>
04	<code> int []a = new int[1];</code>
05	<code> int []b = new int[1];</code>
06	<code> a[0]=10; b[0]=20;</code>
07	<code> //建立MyClass_Value物件屬於MyClass類別</code>
08	<code> MyClass MyClass_Value = new MyClass();</code>
09	<code> System.out.println("==呼叫前==");</code>
10	<code> System.out.println("a=" + a[0] + " " + "b=" + b[0]);</code>
11	<code> //呼叫MyClass類別中的「方法」</code>
12	<code> MyClass_Value.CallByRef(a, b);</code>
13	<code> System.out.println("==呼叫後==");</code>
14	<code> System.out.println("a=" + a[0] + " " + "b=" + b[0]);</code>
15	<code> }</code>
16	<code>}</code>

【執行過程】

主程式與副程式(佔用相同的記憶體位址)



【說明】

當主程式呼叫MyClass類別的方法時，實際參數a所佔用的記憶體位址中的內容(位址)會傳遞給方法中的形式參數x，而實際參數b傳送給形式參數y，因為是參考呼叫，所以主程式的實際參數與方法的形式參數會佔用相同的記憶體位址。因此，方法中的形式參數值改變，也會影響到主程式中的實際參數值。

【執行結果】

```
====呼叫之前====
```

```
a=10  b=20
```

```
====呼叫之後====
```

```
a=100  b=200
```



【課後評量】

1. 請先建立判斷兩科成績是否及格的「類別」，再新增一個屬於該類別的「物件」，用來傳回學生的國文與英文之總平均成績是否及格。
2. 利用類別與物件的方法來實作「西元年生日」轉成「年齡」程式。