CHAPTER 4-2
變數、常數、字串

概要　變數

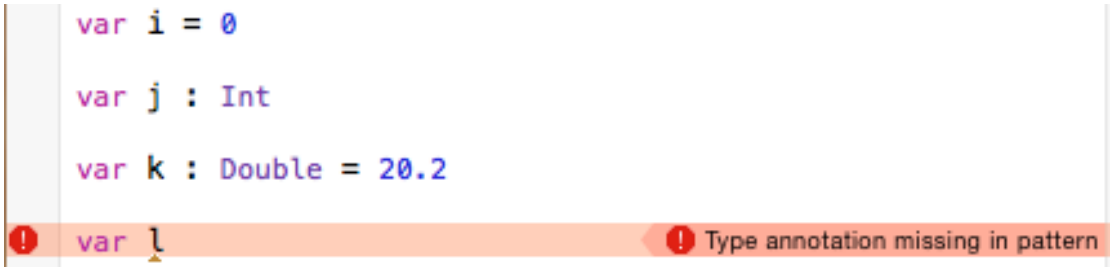
在程式中，可以使用『變數』儲存資料。什麼是變數呢？
想像變數是一個箱子，可以在裡面放置像是數值、文字等等的資料。箱子內的值（變數的值）可以被取用、被改變，因此在宣告一個變數的時候就要給初始值或是變數型態。以下是幾種宣告變數的方法。

- 直接給初始值
- 給定資料型態
- 給定資料型態並設定初始值

變數的宣告方式

```
var 變數名 = 值
```

例　如下圖所示，若只輸入變數名稱就會出現錯誤訊息。



概要　定數

定數與變數一樣，把數字或文字放入箱子內，但與變數不同的是，定數在一開始宣告時，就一定要有初始值，與變數不同的是，定數內的值是不能改變的。

定數的宣告方式

```
let 定數名 = 值
```

例 如下圖所示，若欲改變定數的值，就會出現錯誤訊息。

概要 變數及定數的型別

變數及定數有幾種基本的型別，主要是看變數及定數內想要放什麼類型的資料，來決定其為何種型別。

型別的種類

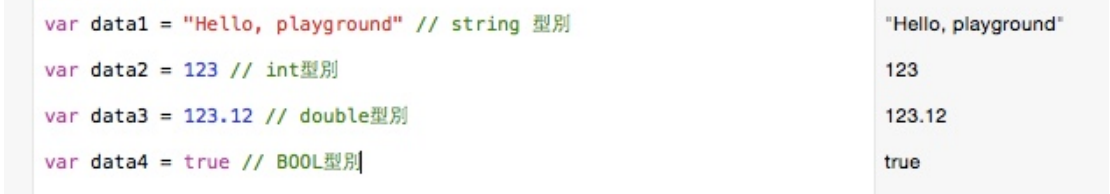
型別的種類主要分為以下四種：

主要的型別種類

整數	Int 型別	整數，例：int n=100
小數	Double 型別	帶有小數的數值，例：double x=10.5
文字	String 型別	存放文字列，例：String a=”apple”
判斷	BOOL 型別	用 true/false 來判斷。例：BOOL open=true

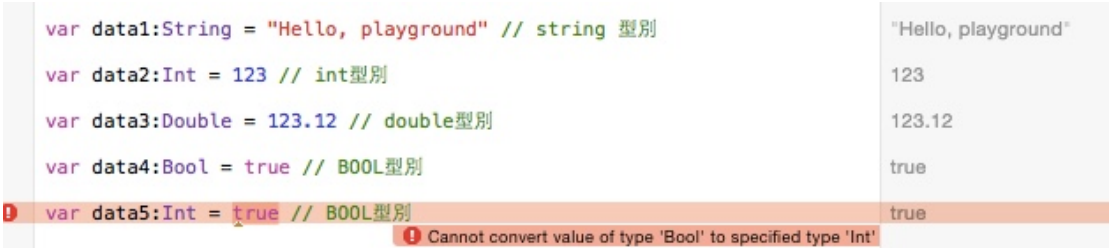
推測型別

Swift 中的變數若未指定型別，它會依照特性幫變數自動推測型別。如下圖範例所示，雖我們未指定型別，但 swift 已幫我們自動推測出型別。



指定型別

我們也可以手動指定型別，但如果指定的型別錯誤，swift 也會判斷檢查並提醒。如下圖範例所示：



轉換型別

若我們想要將 a=100 設定為有小數點的型別，但卻又擔心 swift 判斷檢查 a 為 int 型別，那該怎麼辦呢？別擔心，使用轉換型別的方式即可解決囉！

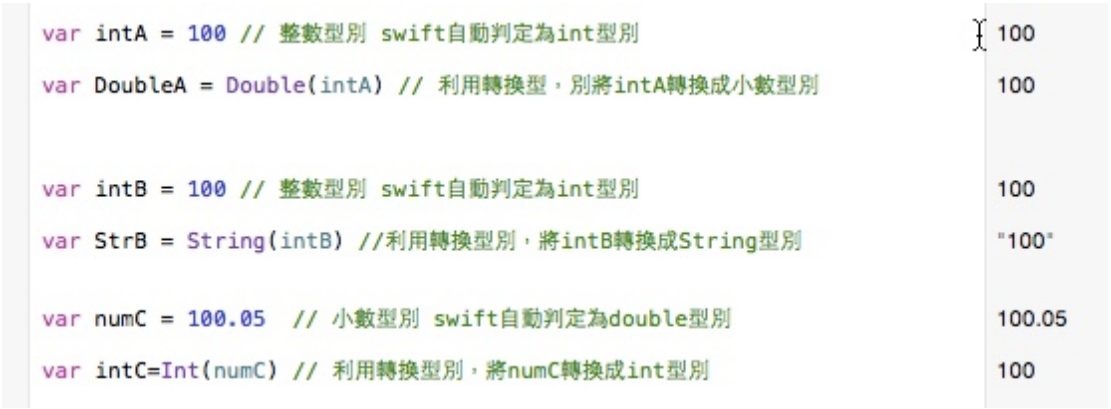
var 轉換後變數名 = 欲轉換的型別(欲轉換的變數名)

例

下圖第一部分為 int 型別轉換成 double 型別；

第二部分為 int 型別轉換為 String 型別；

第三部分為 double 型別轉換為 int 型別。



小技巧　－　兩個以上的變數值初始化

變數值的初始化

var 變數 1=值 1,變數 2=值 2

例

```
var a=1, b=2 //將值放入a,b兩數內
```

( )對應變數值初始化：tuple

var 變數 1=值 1,變數 2=值 2

例

```
var(a,b)=(1,2) //將值放入a,b兩數內（使用對應的方式）
```

文字型別

文字型別的建立

var 變數名=”文字列”

例　將 Hi 文字放入建立好的 Sting 型別變數 Histr 內。

```
var Histr = "Hi"  //將文字Hi放入
```

```
"Hi"
```

文字型別的結合

若我們要將文字列組合在一起，必須使用『＋』符號。

var 變數名 = ”文字列” + ”文字列”

例 將文字 Hi 及 how are you? 兩組文字結合在一起。

```
var Histr = "Hi"  //將文字Hi放入
var Histr1 = Histr + " how are you?"
```

```
"Hi"
"Hi how are you?"
```

將變數放入文字型別中

若我們要將變數放入一組文字類別中，必須使用『\()』符號。

var 變數名 = ”文字列 \(變數) 文字列”

例 將變數穿插在文字中間，使其結合在一起。

```
var FoodStr = "薯條"

var LikeStr = "我喜歡吃\(FoodStr)，太好吃了！"
```

```
"薯條"

"我喜歡吃薯條，太好吃了！"
```

概要 變數的有效範圍

變數在不同的地方指定數值，有可能會改變變數的值。
因此我們這裡要介紹變數的有效範圍是哪裡呢？請大家先看看以下的例子。

例

```
func test(){
    var x=10 //在最外圈指定x為10

    if(1==1){

    print(x)  //列印出的結果x為10

    }

    |
}
test()
```

```
10

"10\n"
```

圖中變數在一開始時我們將它設定為 10，而在判斷式的括號內，將值列印出來。
從這裡可以得知，x 即使到了括號內，依然是有效範圍，可以得出 10 的結果。
再看看以下的例子：

例

```
func test(){
    var x=10 //在最外圈指定x為10

    if(1==1){

    var x = 20 //在判斷式內重新指定x為20
    print(x)  //列印出的結果x為20

    }

    print(x) //最外圈依舊顯示x為10，不受判斷式內的改變影響

}
test()
```

```
10

20
"20\n"

"10\n"
```

圖中變數在一開始時我們將它設定為 10，而在判斷式的括號內，我們將同一變數的值改變為 20。但此改變的有效範圍只限於判斷式內，在判斷式外為 x 的值依然為 10，不會受到判斷式的影響。
由這個例子就可以知道，在指定值給變數時，要注意變數的有效範圍，才不會結果與想要的答案不同喔！