

CH4-3

Optional Value

概要 將 Nil 放入變數中，會出現錯誤

Swift為了讓安全性更加提升，多了一種有更高安全機能的『Optional Value』

什麼是 *optional type*?

「這個東西有可能沒有值」。以 Swift 的角度來看，表示「這東西有可能為`nil`」。

一般變數在用之前常要判斷是否存在或不等於`nil`(`null`)一旦是`null`去使用變數就會出現`error`。

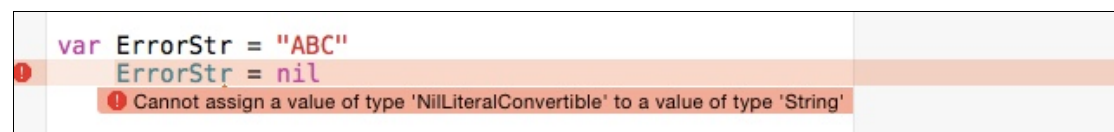
而Optional value就是讓妳省去上述麻煩，有值就回傳值，沒有就給`nil`

因此在有可能發生 `missing values (nil)` 的情況下，要用 Optional Value 的方式操作，這樣比較安全。

Optional Value：應用程序被破壞，會出現防止的安全機能。

當變數的值被設定為`nil`時，系統會啟動此安全機能，出現Error，如下圖所示：

例



```
var ErrorStr = "ABC"
ErrorStr = nil
```

❗ Cannot assign a value of type 'NilLiteralConvertible' to a value of type 'String'

使用方法 Optional Value 中，？與！的使用

(1) Option value：「？」

在 Swift 中，`?` 為 keyword，用來表示 *optional type*，`?` 為 `Optional<Int>` 的縮寫。

Option value：「？」變數的建立

```
var 變數名:型別? = 值
```

我們會在變數的型別後面加上「？」，若沒有加上的話，就無法指定變數為空。

例

```
❗ var myInt :Int = nil
```

若是加上了「？」後，顯示出來的結果就正確了！（如下圖所示）

```
var myInt2 :Int? = nil
```

若是要將Option value的值放入其他的變數中，那另一個變數在宣告時也必須加上「？」才能進行此操作。如下圖範例：

例

```
var myInt :Int? = nil
```

```
❗ var myInt2 :Int = myInt
```

❗ Value of optional type 'Int?' not unwrapped; did you mean to use '!' or '??'

因myInt2在宣告型別時並沒有加上「？」因此出現錯誤，加入後的結果如下：

```
var myInt :Int? = nil
```

```
var myInt2 :Int? = myInt
```

Option value：「？」變數的使用

變數名？

Optional value 要使用的話，也要在變數名的後面加上「？」

例

```
var testStr :String? = "hi88"  
var TestStr2 :String? = testStr.uppercaseString
```

若我們將欲使用的變數後面標上「？」，則可解決以上錯誤。如下圖：

```
var testStr :String? = "hi88"  
var TestStr2 :String? = testStr?.uppercaseString
```

要使用.uppercaseString將文字列轉換功能的時候會錯誤，但使用Option value 就可以使用轉換的這個功能了。

(2) Option value：「！」

若是你非常地確定，這個optional中有值，那我們就可以在此變數後面加上「！」來表示：「我非常肯定其含有值，請使用。」

Option value：「！」變數的使用

```
var 變數:型! = 值
```

若我們已經宣告變數為「！」，表示我們保證此變數有值，若接下來我們將它設為nil，則無法編譯成功。

例

<pre>var OptionalValueStr:String! = "hello" OptionalValueStr = nil var myStr:String = OptionalValueStr</pre>	<pre>"hello" nil ! error</pre>
<p>! Execution was interrupted, reason: EXC_BAD_INSTRUCTION (code=EXC_I386_INVOP, subc...</p>	

因此，我們必須在變數為`nil`時，指定值給變數，使變數維持在有值的狀態。
如下圖所示。

<pre>var OptionalValueStr:String! = "hello" OptionalValueStr = nil if(OptionalValueStr == nil){ OptionalValueStr = "goodbye" } var myStr:String = OptionalValueStr</pre>	<pre>"hello" nil "goodbye" "goodbye"</pre>
--	--

(3) Optional Binding (Optional 綁定)

我們可以使用Optional Binding來判斷Optional是否有值，若有值的話，我們就將其值取出給一個暫存的變數。

```
if let 暫存的變數名 = 變數名 {
    //變數的值若不是 nil 則執行此處
}else{
    //變數的值若是 nil 則執行此處
}
```

例

<pre>var ageInt:Int = 0 var OpVInt:Int? = nil if let temp = OpVInt { ageInt = OpVInt! }else{ ageInt = 18; }</pre>	<pre>0 nil 18</pre>
--	----------------------

(4) 將 String 值轉換成 Int 值的方法

有時我們會得到一個String恰好為數字，當我們要使用它時卻會產生錯誤，因此接下來我們要教大家如何將String值中的數字轉換為Int型別。

這裡我們會用到的語法為 `Int()` 只要使用此語法便可以成功轉換型別。

例 使用Int()時，出現錯誤

<pre>var ageStr:String = "23"</pre>	"23"
<pre>var ageInt:Int = Int(ageStr)</pre>	23
<pre>Value of optional type 'Int?' not unwrapped; did you mean to use '!' or '?'?</pre>	

為什麼在這裡會出現錯誤呢？

因為系統會擔心有轉型失敗的狀況，而怎麼樣才會轉型失敗呢？

如果裡面的值不是數字，而是文字的話，就很有可能轉型失敗，因此我們必須將要用來存取轉型後值的ageInt設為Optional Value變數，表示他有可能因為轉型失敗而值為nil，若真的為nil，再將他指到另一個數。

<pre>var ageStr:String = "23"</pre>	"23"
<pre>var ageInt:Int? = Int(ageStr)</pre>	23
<pre>if (ageInt == nil){ ageInt = 0 }</pre>	

<pre>var ageStr:String = "Hi"</pre>	"Hi"
<pre>var ageInt:Int? = Int(ageStr)</pre>	nil
<pre>if (ageInt == nil){ ageInt = 0 }</pre>	0