Swift 变量 Swift 常量

Swift 可选(Optionals)类型

Swift 的可选（Optional）类型，用于处理值缺失的情况。可选表示"那儿有一个值，并且它等于 x "或者"那儿没有值"。

Swfit语言定义后缀？作为命名类型Optional的简写，换句话说，以下两种声明是相等的：

```
var optionalInteger: Int?
var optionalInteger: Optional<Int>
```

在这两种情况下，变量 optionalInteger 都是可选整数类型。注意，在类型和 **?**之间没有空格。

Optional 是一个含有两种情况的枚举，None 和 Some(T)，用来表示可能有或可能没有值。任何类型都可以明确声明为（或者隐式转换）可选类型。当声明一个可选类型的时候，要确保用括号给？操作符一个合适的范围。例如，声明可选整数数组，应该写成 (Int[])? 写成 Int[]? 会报错。

当你声明一个可选变量或者可选属性的时候没有提供初始值，它的值会默认为 nil。

可选项遵照 LogicValue 协议，因此可以出现在布尔环境中。在这种情况下，如果可选类型T?包含类型为T的任何值（也就是说它的值是 Optional.Some(T) ），这个可选类型等于 true，反之为 false。

如果一个可选类型的实例包含一个值，你可以用后缀操作符 ！来访问这个值，如下所示：

```
optionalInteger = 42
optionalInteger! // 42
```

使用操作符！去获取值为nil的可选变量会有运行时错误。

你可以用可选链接和可选绑定选择性执行可选表达式上的操作。如果值为nil，任何操作都不会执行，也不会有运行报错。

让我们来详细看下以下实例来了解 Swift 中可选类型的应用:

```
import Cocoa

var myString:String? = nil

if myString != nil {
    print(myString)
}else{
    print("字符串为 nil")
}
```

以上程序执行结果为：

```
字符串为 nil
```

可选类型类似于Objective-C中指针的nil值，但是nil只对类(class)有用，而可选类型对所有的类型都可用，并且更安全。

强制解析

当你确定可选类型确实包含值之后，你可以在可选的名字后面加一个感叹号（!）来获取值。这个感叹号表示"我知道这个可选有值，请使用它。"这被称为可选值的强制解析（forced unwrapping）。

实例如下：

```
import Cocoa

var myString:String?

myString = "Hello, Swift!"

if myString != nil {
   print(myString)
}else{
   print("myString 值为 nil")
}
```

以上程序执行结果为：

```
Optional("Hello, Swift!")
```

强制解析可选值，使用感叹号（!）：

```
import Cocoa

var myString:String?

myString = "Hello, Swift!"

if myString != nil {
   // 强制解析
   print( myString! )
}else{
   print("myString 值为 nil")
}
```

以上程序执行结果为：

```
Hello, Swift!
```

注意：
使用!来获取一个不存在的可选值会导致运行时错误。使用!来强制解析值之前，一定要确定可选包含一个非n

```
il的值。
```

自动解析

你可以在声明可选变量时使用感叹号（!）替换问号（？）。这样可选变量在使用时就不需要再加一个感叹号（!）来获取值，它会自动解析。

实例如下：

```
import Cocoa

var myString:String!

myString = "Hello, Swift!"

if myString != nil {
   print(myString)
}else{
   print("myString 值为 nil")
}
```

以上程序执行结果为：

```
Hello, Swift!
```

可选绑定

使用可选绑定（optional binding）来判断可选类型是否包含值，如果包含就把值赋给一个临时常量或者变量。可选绑定可以用在if和while语句中来对可选类型的值进行判断并把值赋给一个常量或者变量。

像下面这样在if语句中写一个可选绑定：

```
if let constantName = someOptional {
    statements
}
```

让我们来看下一个简单的可选绑定实例：

```
import Cocoa

var myString:String?

myString = "Hello, Swift!"

if let yourString = myString {
```

```
   print("你的字符串值为 - \(yourString)")
}else{
   print("你的字符串没有值")
}
```

以上程序执行结果为：

```
你的字符串值为 - Hello, Swift!
```

Swift 变量 Swift 常量

点我分享笔记