Swift 闭包 Swift 结构体

Swift 枚举

枚举简单的说也是一种数据类型，只不过是这种数据类型只包含自定义的特定数据，它是一组有共同特性的数据的集合。

Swift 的枚举类似于 Objective C 和 C 的结构，枚举的功能为:

- 它声明在类中，可以通过实例化类来访问它的值。
- 枚举也可以定义构造函数（initializers）来提供一个初始成员值；可以在原始的实现基础上扩展它们的功能。
- 可以遵守协议（protocols）来提供标准的功能。

语法

Swift 中使用 enum 关键词来创建枚举并且把它们的整个定义放在一对大括号内：

```
enum enumname {
   // 枚举定义放在这里
}
```

例如我们定义以下表示星期的枚举：

```
import Cocoa

// 定义枚举
enum DaysofaWeek {
    case Sunday
    case Monday
    case TUESDAY
    case WEDNESDAY
    case THURSDAY
    case FRIDAY
    case Saturday
}

var weekDay = DaysofaWeek.THURSDAY
weekDay = .THURSDAY
switch weekDay
{
case .Sunday:
    print("星期天")
case .Monday:
    print("星期一")
case .TUESDAY:
    print("星期二")
```

```
case .WEDNESDAY:
    print("星期三")
case .THURSDAY:
    print("星期四")
case .FRIDAY:
    print("星期五")
case .Saturday:
    print("星期六")
}
```

以上程序执行输出结果为：

```
星期四
```

枚举中定义的值（如 Sunday，Monday，……和Saturday）是这个枚举的**成员值**（或**成员**）。case关键词表示一行新的成员值将被定义。

> *注意：和 C 和 Objective-C 不同，Swift 的枚举成员在被创建时不会被赋予一个默认的整型值。在上面的DaysofaWeek例子中，Sunday，Monday，……和Saturday不会隐式地赋值为0，1，……和6。相反，这些枚举成员本身就有完备的值，这些值是已经明确定义好的DaysofaWeek类型。*

```
var weekDay = DaysofaWeek.THURSDAY
```

weekDay的类型可以在它被DaysofaWeek的一个可能值初始化时推断出来。一旦weekDay被声明为一个DaysofaWeek，你可以使用一个缩写语法（.）将其设置为另一个DaysofaWeek的值：

```
var weekDay = .THURSDAY
```

当weekDay的类型已知时，再次为其赋值可以省略枚举名。使用显式类型的枚举值可以让代码具有更好的可读性。

枚举可分为相关值与原始值。

相关值与原始值的区别

相关值	原始值
不同数据类型	相同数据类型
实例: enum {10,0.8,"Hello"}	实例: enum {10,35,50}
值的创建基于常量或变量	预先填充的值
相关值是当你在创建一个基于枚举成员的新常量或变量时才会被设置，并且每次当你这么做得时候，它	原始值始终是相同

的值可以是不同的。	的

相关值

以下实例中我们定义一个名为 Student 的枚举类型，它可以是 Name 的一个字符串（String），或者是 Mark 的一个相关值（Int，Int，Int）。

```
import Cocoa

enum Student{
    case Name(String)
    case Mark(Int,Int,Int)
}
var studDetails = Student.Name("Runoob")
var studMarks = Student.Mark(98,97,95)
switch studMarks {
case .Name(let studName):
    print("学生的名字是: \(studName)。")
case .Mark(let Mark1, let Mark2, let Mark3):
    print("学生的成绩是: \(Mark1),\(Mark2),\(Mark3)。")
}
```

以上程序执行输出结果为：

```
学生的成绩是: 98,97,95。
```

原始值

原始值可以是字符串，字符，或者任何整型值或浮点型值。每个原始值在它的枚举声明中必须是唯一的。

在原始值为整数的枚举时，不需要显式的为每一个成员赋值，Swift会自动为你赋值。

例如，当使用整数作为原始值时，隐式赋值的值依次递增1。如果第一个值没有被赋初值，将会被自动置为0。

```
import Cocoa

enum Month: Int {
    case January = 1, February, March, April, May, June, July, August, September, October, November, Dec
ember
}

let yearMonth = Month.May.rawValue
print("数字月份为: \(yearMonth)。")
```

以上程序执行输出结果为：

```
数字月份为：5。
```

Swift 闭包 Swift 结构体

点我分享笔记