下标

@M了个J

https://github.com/CoderMJLee

http://cnblogs.com/mjios

码拉松

下标（subscript）

- 使用subscript可以给任意类型（枚举、结构体、类）增加下标功能，有些地方也翻译为：下标脚本
  - subscript的语法类似于实例方法、计算属性，本质就是方法（函数）

```swift
class Point {
    var x = 0.0, y = 0.0
    subscript(index: Int) -> Double {
        set {
            if index == 0 {
                x = newValue
            } else if index == 1 {
                y = newValue
            }
        }
        get {
            if index == 0 {
                return x
            } else if index == 1 {
                return y
            }
            return 0
        }
    }
}
```

```swift
var p = Point()
p[0] = 11.1
p[1] = 22.2
print(p.x) // 11.1
print(p.y) // 22.2
print(p[0]) // 11.1
print(p[1]) // 22.2
```

- subscript中定义的返回值类型决定了
  - get方法的返回值类型
  - set方法中newValue的类型

- subscript可以接受多个参数，并且类型任意

下标的细节

- subscript可以没有set方法，但必须要有get方法

```
class Point {
    var x = 0.0, y = 0.0
    subscript(index: Int) -> Double {
        get {
            if index == 0 {
                return x
            } else if index == 1 {
                return y
            }
            return 0
        }
    }
}
```

- 如果只有get方法，可以省略get

```
class Point {
    var x = 0.0, y = 0.0
    subscript(index: Int) -> Double {
        if index == 0 {
            return x
        } else if index == 1 {
            return y
        }
        return 0
    }
}
```

下标的细节

- 可以设置参数标签

```
class Point {
    var x = 0.0, y = 0.0
    subscript(index i: Int) -> Double {
        if i == 0 {
            return x
        } else if i == 1 {
            return y
        }
        return 0
    }
}
```

```
var p = Point()
p.y = 22.2
print(p[index: 1]) // 22.2
```

- 下标可以是类型方法

```
class Sum {
    static subscript(v1: Int, v2: Int) -> Int {
        return v1 + v2
    }
}
print(Sum[10, 20]) // 30
```

结构体、类作为返回值对比

```swift
class Point {
    var x = 0, y = 0
}
class PointManager {
    var point = Point()
    subscript(index: Int) -> Point {
        get { point }
    }
}
```

```swift
var pm = PointManager()
pm[0].x = 11
pm[0].y = 22
// Point(x: 11, y: 22)
print(pm[0])
// Point(x: 11, y: 22)
print(pm.point)
```

```swift
struct Point {
    var x = 0, y = 0
}
class PointManager {
    var point = Point()
    subscript(index: Int) -> Point {
        set { point = newValue }
        get { point }
    }
}
```

接收多个参数的下标

```swift
class Grid {
    var data = [
        [0, 1, 2],
        [3, 4, 5],
        [6, 7, 8]
    ]
    subscript(row: Int, column: Int) -> Int {
        set {
            guard row >= 0 && row < 3 && column >= 0 && column < 3 else {
                return
            }
            data[row][column] = newValue
        }
        get {
            guard row >= 0 && row < 3 && column >= 0 && column < 3 else {
                return 0
            }
            return data[row][column]
        }
    }
}
```

```swift
var grid = Grid()
grid[0, 1] = 77
grid[1, 2] = 88
grid[2, 0] = 99
print(grid.data)
```