Swift 字符(Character) Swift 字典

Swift 数组

Swift 数组使用有序列表存储同一类型的多个值。相同的值可以多次出现在一个数组的不同位置中。

Swift 数组会强制检测元素的类型，如果类型不同则会报错，Swift 数组应该遵循像Array<Element>这样的形式，其中Element是这个数组中唯一允许存在的数据类型。

如果创建一个数组，并赋值给一个变量，则创建的集合就是可以修改的。这意味着在创建数组后，可以通过添加、删除、修改的方式改变数组里的项目。如果将一个数组赋值给常量，数组就不可更改，并且数组的大小和内容都不可以修改。

创建数组

我们可以使用构造语法来创建一个由特定数据类型构成的空数组：

```
var someArray = [SomeType]()
```

以下是创建一个初始化大小数组的语法：

```
var someArray = [SomeType](repeating: InitialValue, count: NumbeOfElements)
```

以下实例创建了一个类型为 Int ，数量为 3，初始值为 0 的空数组：

```
var someInts = [Int](repeating: 0, count: 3)
```

以下实例创建了含有三个元素的数组：

```
var someInts:[Int] = [10, 20, 30]
```

访问数组

我们可以根据数组的索引来访问数组的元素，语法如下：

```
var someVar = someArray[index]
```

index 索引从 0 开始，即索引 0 对应第一个元素，索引 1 对应第二个元素，以此类推。

我们可以通过以下实例来学习如何创建，初始化，访问数组：

```
import Cocoa

var someInts = [Int](repeating: 10, count: 3)
```

```
var someVar = someInts[0]

print( "第一个元素的值 \(someVar)" )
print( "第二个元素的值 \(someInts[1])" )
print( "第三个元素的值 \(someInts[2])" )
```

以上程序执行输出结果为：

```
第一个元素的值 10
第二个元素的值 10
第三个元素的值 10
```

修改数组

你可以使用 append() 方法或者赋值运算符 += 在数组末尾添加元素，如下所示，我们初始化一个数组，并向其添加元素：

```
import Cocoa

var someInts = [Int]()

someInts.append(20)
someInts.append(30)
someInts += [40]

var someVar = someInts[0]

print( "第一个元素的值 \(someVar)" )
print( "第二个元素的值 \(someInts[1])" )
print( "第三个元素的值 \(someInts[2])" )
```

以上程序执行输出结果为：

```
第一个元素的值 20
第二个元素的值 30
第三个元素的值 40
```

我们也可以通过索引修改数组元素的值：

```
import Cocoa

var someInts = [Int]()

someInts.append(20)
someInts.append(30)
someInts += [40]
```

```
// 修改最后一个元素
someInts[2] = 50

var someVar = someInts[0]

print( "第一个元素的值 \(someVar)" )
print( "第二个元素的值 \(someInts[1])" )
print( "第三个元素的值 \(someInts[2])" )
```

以上程序执行输出结果为：

```
第一个元素的值 20
第二个元素的值 30
第三个元素的值 50
```

遍历数组

我们可以使用for-in循环来遍历所有数组中的数据项：

```
import Cocoa

var someStrs = [String]()

someStrs.append("Apple")
someStrs.append("Amazon")
someStrs.append("Runoob")
someStrs += ["Google"]

for item in someStrs {
   print(item)
}
```

以上程序执行输出结果为：

```
Apple
Amazon
Runoob
Google
```

如果我们同时需要每个数据项的值和索引值，可以使用 String 的 enumerate() 方法来进行数组遍历。实例如下：

```
import Cocoa

var someStrs = [String]()
```

```
someStrs.append("Apple")
someStrs.append("Amazon")
someStrs.append("Runoob")
someStrs += ["Google"]

for (index, item) in someStrs.enumerated() {
    print("在 index = \(index) 位置上的值为 \(item)")
}
```

以上程序执行输出结果为：

```
在 index = 0 位置上的值为 Apple
在 index = 1 位置上的值为 Amazon
在 index = 2 位置上的值为 Runoob
在 index = 3 位置上的值为 Google
```

合并数组

我们可以使用加法操作符（+）来合并两种已存在的相同类型数组。新数组的数据类型会从两个数组的数据类型中推断出来：

```
import Cocoa

var intsA = [Int](repeating: 2, count:2)
var intsB = [Int](repeating: 1, count:3)

var intsC = intsA + intsB

for item in intsC {
    print(item)
}
```

以上程序执行输出结果为：

```
2
2
1
1
1
```

count 属性

我们可以使用 count 属性来计算数组元素个数：

```
import Cocoa

var intsA = [Int](count:2, repeatedValue: 2)
var intsB = [Int](count:3, repeatedValue: 1)

var intsC = intsA + intsB

print("intsA 元素个数为 \(intsA.count)")
print("intsB 元素个数为 \(intsB.count)")
print("intsC 元素个数为 \(intsC.count)")
```

以上程序执行输出结果为：

```
intsA 元素个数为 2
intsB 元素个数为 3
intsC 元素个数为 5
```

isEmpty 属性

我们可以通过只读属性 isEmpty 来判断数组是否为空，返回布尔值：

```
import Cocoa

var intsA = [Int](count:2, repeatedValue: 2)
var intsB = [Int](count:3, repeatedValue: 1)
var intsC = [Int]()

print("intsA.isEmpty = \(intsA.isEmpty)")
print("intsB.isEmpty = \(intsB.isEmpty)")
print("intsC.isEmpty = \(intsC.isEmpty)")
```

以上程序执行输出结果为：

```
intsA.isEmpty = false
intsB.isEmpty = false
intsC.isEmpty = true
```

← Swift 字符(Character) Swift 字典 →

点我分享笔记