≔ Swift 3 Collections

★ Filter / Reduce / FlatMap的实现和扩展

常用的Dictionary extension ▶

(https://www.boxueio.com/series/collection-types/ebook/128)

(https://www.boxueio.com/series/collection-types/ebook/130)

和Dictionary相关的基础知识

● Back to series (/series/collection-types)

形式的value。我们来看一个例子。

定义Dictionary

假设我们要定义一个数据结构,用来保存用户在泊学对某个视频的观看情况。可以这样:

```
enum RecordType {
    case bool(Bool)
    case number(Int)
    case text(String)
}

let record11: [String: RecordType] = [
    "uid": .number(11),
    "exp": .number(100),
    "favourite": .bool(true),
    "title": .text("Dictionary basics")
]
```

在上面代码里,我们用 [KeyType: ValueType] 的形式来定义一个 Dictionary 。当定义好 Dictionary 之后,我们就能直接用 [Key] 来访问某个key对应的值了:

```
record11["uid"]  // number(11)
record11["favourite"] // bool(true)
record11["title"]  // text("Dictionary basics")
record11["invalid"]  // nil

// Optional<RecordType>.Type
type(of: record11["favourite"])
```

上面例子中的结果都很直观。但是有一个细节却是值得我们注意的。和 Array 不同的是, [] 用在 Dictionary 的时候,会返回一个Optional类型来确保这种形式的访问安全。因此,访问不存在的key,并不会导致运行时错误。

你怎么理解这种差异呢?

这是因为索引这个概念,对 Array 和 Dictionary 来说,是截然不同的。对于 Array 来说,我们有可能使用的正常索引值只源于 Array 自身,也就是 0...<array.count ,因此,如果你使用了不在这个范围里的值,则一定是可以被定性为Bug的,何况,我们之前也看到了,对于 Array ,我们几乎不需要直接使用索引来访问元素。

而对于 Dictionary 来说,它包含的内容并不直接决定我们可以查询的内容。举个例子来说,英汉词典中也可能并不包含我们要查询的单词。所以, Dictionary 中包含的所有键值,从语义上说,并不完全决定了它的使用者会查询的值,所以,我们也无法把这类问题明确的归因于是Bug。所以,Swfit为 Dictionary 的索引查询操作,提供了optional保护。要么得到正确的结果,要么通过 nil 表示要查询的内容不存在。

常用的基本属性

作为一个集合类型, Dictionary 同样有 count 和 is Empty 两个属性读取其元素的个数以及判断其是否为空:

```
record11.count // 4
record11.isEmpty // false
```

另外,我们可以单独访问一个 Dictionary 的所有 keys 和所有 values:

● 字号

✔ 默认主题

✔ 金色主题

✔ 暗色主题

```
record11.keys
record11.values
```

这两个属性也分别是一个集合,我们可以暂时忽略掉它们具体的类型,如果要我们要访问它们的每一个元素,直接用 for 循环或 forEach 遍历就好了:

```
for key in record11.keys { print(key) }
// or
record11.keys.forEach { print($0) }
```

添加、更新和删除元素

和 Array 一样, Dictionary 也是一个值类型,当我们复制 Dictionary 对象的时候,就会拷贝 Dictionary 中的所有内容:

```
var record10 = record11
```

并且,直接使用key就可以访问和修改 Dictionary 的内容:

```
record10['favourite'] = .bool(false) // false
record11['favourite'] // true
```

如果我们希望更新value的时候,同时获得修改前的值,还可以使用 updateValue(_:forKey:) 方法:

```
record10.updateValue(.bool(true),
forKey: "favourite") // .bool(false)
```

从上面的结果可以看出修改 record10 并不会影响 record11。

当我们要在 Dictionary 中添加元素时,直接给要添加的key赋值就好了:

```
record10["watchLater"] = .bool(false)
// [
// "favourite": RecordType.bool(false),
// "exp": RecordType.number(100),
// "title": RecordType.text("Directory basics"),
// "uid": RecordType.number(11),
// "watchLater": RecordType.bool(false)
// ]
```

这样, record10 中的内容,就变成了5项。而当我们要删除特定的key时,直接把它的值设置为 nil:

```
record10["watchLater"] = nil
// [
// "favourite": RecordType.bool(false),
// "exp": RecordType.number(100),
// "title": RecordType.text("Directory basics"),
// "uid": RecordType.number(11)
// ]
```

这里,并不是把特定key的值设置为 nil (毕竟 Dictionary 中value部分的类型也不是optional),而是删除特定的key。当某个key的value被设置成 nil 后,这个key也就从 Dictionary 中删除了。

遍历Dictionary

由于 Dictionary 同时包含了key和value,因此,我们也有多重方式来遍历 Dictionary 。最简单的,就是遍历 Dictionary 中的每一个元素:

```
for (k, v) in record10 {
    print("\(k): \(v)")
}
record10.forEach { print("\($0): \($1)") }
```

从上面的例子可以看到,遍历 Dictionary 和遍历 Array 是类似的。当我们使用 for 循环遍历时,它的每一个元素都用一个tuple来表示,封装了每一个元素的key和value。而当使用 forEach 方法时,它会给它的closure参数传递两个值,分别是每一个元素的key和value。

但是,由于 Dictionary 是一个无序集合(unordered collection),因此当我们编辑了 Dictionary 之后,每次遍历,访问元素的顺序都可能是不同的。如果我们希望按照固定的顺序来访问 Dictionary 中的元素,一个最简单的办法,就是对key排序后,再进行遍历:

```
for key in record10.keys.sorted() {
   print("\(key): \(record10[key])")
}
```

What's next?

在了解了 Dictionary 的基本用法之后,下一节,我们通过 extension 给 Dictionary 添加一些标准库中没有但却常用的操作,以此进一步理解 Dictionary 的用法。

₭ Filter / Reduce / FlatMap的实现和扩展

常用的Dictionary extension ▶

(https://www.boxueio.com/series/collection-types/ebook/128)

(https://www.boxueio.com/series/collection-types/ebook/130)



职场漂泊的你,每天多学一点。

从开发、测试到运维,让技术不再成为你成长的绊脚石。我们用打磨产品的精神去传播知识,把最新的移动开发技术,通过简单的图表, 清晰的视频,简明的文字和切实可行的例子一 一向你呈现。让学习不仅是一种需求,也是一种享受。

泊学动态

一个工作十年PM终创业的故事(二) (https://www.boxueio.com/after-the-full-upgrade-to-swift3)

Mar 4, 2017

人生中第一次创业的"10有" (https://www.boxueio.com/founder-chat)

Jan 9, 2016

猎云网采访报道泊学 (http://www.lieyunwang.com/archives/144329)

Dec 31, 2015

What most schools do not teach (https://www.boxueio.com/what-most-schools-do-not-teach)

Dec 21, 2015

一个工作十年PM终创业的故事(一) (https://www.boxueio.com/founder-story)

May 8, 2015

泊学相关

关于泊学

加入泊学

泊学用户隐私以及服务条款 (HTTPS://WWW.BOXUEIO.COM/TERMS-OF-SERVICE)

版权声明 (HTTPS://WWW.BOXUEIO.COM/COPYRIGHT-STATEMENT)

联系泊学

Email: 10[AT]boxue.io (mailto:10@boxue.io)

QQ: 2085489246

2017 © Boxue, All Rights Reserved. 京ICP备15057653号-1 (http://www.miibeian.gov.cn/) 京公网安备 11010802020752号 (http://www.beian.gov.cn/portal/registerSystemInfo? recordcode=11010802020752)

友情链接 SwiftV (http://www.swiftv.cn) | Seay信息安全博客 (http://www.cnseay.com) | Swift.gg (http://swift.gg/) | Laravist (http://laravist.com/) | SegmentFault (https://segmentfault.com) | 靛青K的博客 (http://blog.dianqk.org/)