What and Why in Swift 3.1

SE-0080 数值类型的failable initialize
(https://www.boxueio.com/series/what-is-new-in-swift-31/ebook/207)

SE-0103 临时转换成escaping的closure
(https://www.boxueio.com/series/what-is-new-in-swift-31/ebook/209)

SE-0045 Sequence中新添加的两个筛选元素的方法

Back to series (/series/what-is-new-in-swift-31)

字号
字号
默认主题
金色主题
暗色主题

在Swift 3.1中，根据SE-0045 (https://github.com/apple/swift-evolution/blob/master/proposals/0045-scan-takewhile-dropwhile.md)中的描述，给 Sequence 类型添加了两个方法。在了解这两个方法之前，我们先来学习两个Swift标准库中的泛型函数。用它们可以方便的创建无限元素个数的 Sequence 类型。

假设，我们要创建一个序列，其中每个位置的元素都是当前索引位置的阶乘，就可以这样：

```
let factorialArray = sequence(state: (1, 1), next: {
    (state: inout (Int, Int)) -> Int? in

    defer {
        state.0 = state.0 * state.1
        state.1 += 1
    }

    return state.0
})
```

其中， sequence(state:next:) 就是我们要介绍的第一个方法，它完整的声明是这样的：

```
func sequence<T, State>(
    state: State,
    next: @escaping (inout State) -> T?) -> UnfoldSequence<T, State>
```

简单来说，就是以 state 为初始状态，不断调用 next 指定的closure方法生成序列中的元素。

可以在声明中看到，sequence带有两个泛型参数， T 和 State 。 T 表示生成的 Sequence 中元素的类型， State 则用于在多次调用 next 之间，传递状态。值得注意的是， next 中的参数是有 inout 修饰的，因此，在closure中的修改，会传递到下一次 next 调用。

理解了这个函数的用法之后，我们之前的代码就很简单了。我们用一个包含两个成员的tuple表示生成序列的状态，其中，第一个成员表示生成的序列中的值，第二个成员用于记录在每次调用 next 时计数。于是：

首先，把 (1, 1) 传递给 next ，这里由于我们使用了 defer 关键字， return state.0 会先被评估，也就是 return 1 ，然后执行 defer 中的代码，此时 state 就变成了 (1, 2) ，此时我们计算的是0的阶乘；

其次， (1, 2) 传递给 next ，执行和第一步同样的逻辑，得到 return 1 ，但是 state 就变成了 (2, 3) ，此时我们计算的是1的阶乘；

这样，以此类推，我们就能计算每个索引位置上的阶乘了，并且，通过这个结果，我们也能进一步理解 defer 关键字的作用和工作方式。

此外，还有一个和 sequence(state:next:) 类似的方法，叫做 sequence(first:next:) ，它们唯一的不同，就是后者在多次调用 next 方法时，不会传递被修改的状态，我们只能基于 first 参数，来生成序列。

现在，有了 factorialArray 之后，我们就可以使用SE-0045 (https://github.com/apple/swift-evolution/blob/master/proposals/0045-scan-takewhile-dropwhile.md)中新添加的两个方法了。

除了像 factorialArray.prefix(10) 这样截取前10个元素之外，我们还可以在读取序列时设置更复杂的条件，例如，找到阶乘中所有小于10000的元素：

```
factorialArray
    .prefix(while: { $0 < 10000 })
    .forEach { print($0) }
// 1
// 1
// 2
// 6
// 24
// 120
// 720
// 5040
```

这里的 `prefix(while:)` 就是第一个新添加的方法。当然，如果使用trailing closure，看上去就会更简单一些：

```
factorialArray
    .prefix { $0 < 10000 }
    .forEach { print($0) }
// 1
// 1
// 2
// 6
// 24
// 120
// 720
// 5040
```

除了指定包含条件之外， Sequence 中还添加了一个指定排除条件的方法 `drop(while:)`，例如，为了在阶乘小于10000的结果里再去除掉所有小于100的，就可以这样：

```
factorialArray
.prefix { $0 < 10000 }
.drop { $0 < 100 }
.forEach { print($0) }
// 120
// 720
// 5040
```

这样，就比我们先在序列中截取出元素，然后再进行条件判断简单和直观多了。

◀ SE-0080 数值类型的failable initialize
(https://www.boxueio.com/series/what-is-new-in-swift-31/ebook/207)

SE-0103 临时转换成escaping的closure ▶
(https://www.boxueio.com/series/what-is-new-in-swift-31/ebook/209)

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)