

## ☰ What and Why in Swift 3.1

◀ 返回视频

SE-0045 Sequence中新添加的两个筛选元素的方法 ▶

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

(https://www.boxueio.com/series/what-is-new-in-swift-31/ebook/208)

# SE-0080 数值类型的failable initialize

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

Apple终于发布了Xcode 8.3以及Swift 3.1。如果你没时间仔细通读一遍release note，至少，Swift 3.1中的一些新特性还是值得了解的。在这个系列里，我们就向大家介绍它们。当然，Swift 3.1和Swift 3在源代码级别是兼容的，因此，如果你的项目已经更新到了Swift 3，这次更新应该不会给你带来太多麻烦。但是，Xcode 8.3去掉了对Swift 2.3的支持，所以，如果你还停留在更早版本的Swift上，就还是谨慎更新的为好。

## 所有的数字类型都有了failable initializer

这个改动，来自于SE-0080 (<https://github.com/apple/swift-evolution/blob/master/proposals/0080-failable-numeric-initializers.md>)。Swift为所有的数字类型定义了failable initializer，当构造失败的时候，就会返回 nil。

我们先来直观感受下这个特性：

```
let a = 2.33
let b = Int(a)           // 2
let c = Int(exactly: a) // nil

let d = 2.0
let e = Int(exactly: d) // 2
```

在上面这段代码里，Int(exactly:) 就是为数字类型新添加的failable initializer。我们可以从注释的结果中看到，它比 Int(a) 这样的用法对数值的检查更为严格，决不允许在生成数字的时候造成任何损失。因此，Int(exactly: a) 的结果就是 nil，而 Int(exactly: d) 的结果，就是2。

为什么要添加这个改动呢？很多时候，我们要在运行时尝试把一个诸如 Any 这样的松散类型转换成数字类型。最典型的一个场景，就是解析服务器返回的json，它的类型，是 [string: Any]，当我们需要把其中的某个字符串键值精确的转换为数字类型时，这个特性用起来就会很方便。来看下面这个例子：

我们先定义一个表示苹果的 struct，它只有一个表示重量的属性 weight，我们要求它的重量必须是整数：

```
struct Apple {
    var weightInGram: Int
}
```

然后，为了可以根据服务器返回的结果直接创建 Apple 对象，我们给它添加一个这样的failable initializer：

```
struct Apple {
    var weightInGram: Int

    init?(json: [String: Any]) {
        guard let weightInString = json["weight"] as? String,
              let weightInDouble = Double(weightInString),
              let weight = Int(exactly: weightInDouble)
        else {
            return nil
        }

        self.weightInGram = weight
    }
}
```

在这里，我们先把返回的 String “无损”的转换成了 Double，如果这一步转换成功了，至少我们可以确定的是得到了一个数字，然后，再用 Int(exactly:) 方法，进一步把 Double 转换成 Int，这一步，也只有在无损的情况下才可以转换成功。这样，就可以得到重量正好是整数值的苹果了。我们可以用下面的代码试一下：

- 🔍 字号
- 🔍 字号
- 🖌 默认主题
- 🖌 金色主题
- 🖌 暗色主题

首先，模拟一个服务器返回的 Data：

```
let applesData = "[{\\"weight\\":\\"500.0\\"},{\\"weight\\":\\"500.1\\"},{\\"weight\\":\\"499.9\\"}]"
let data = applesData.data(using: .utf8)!
```

其次，定义一个过滤出重量正好是整数苹果的方法：

```
func filterApple(from data: Data) -> [Apple] {
    guard let json = try? JSONSerialization.jsonObject(
        with: data,
        options: .allowFragments),
        let dataArray = json as? [[String: Any]]
    else {
        return []
    }

    return dataArray.flatMap(Apple.init)
}
```

最后，用刚才定义好的 data 来试一下：

```
let apples = filterApple(from: data)

print(apples.count) // 1
apples.forEach { print($0) } // Apple(weightInGram: 500)
```

从注释的结果就可以看到，最终我们只得到了一个重量正好是500克的苹果。

---

◀ 返回视频

SE-0045 Sequence中新添加的两个筛选元素的方法 ▶

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

(https://www.boxueio.com/series/what-is-new-in-swift-31/ebook/208)

---



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——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)