理解Reactive编程思想

Back to series (/series/reactive-programming-in-swift)

抛开那些复杂晦涩的理论，在这段视频里，我们通过对比一个大家熟悉的例子，帮助大家熟悉Reactive Programming的编程思想。

字号
字号
默认主题
金色主题
暗色主题

从过滤一个常量数组开始

假设我们有一个包含数字1-10的字符串数组：

```
let stringArray = ["1", "2", "3", "4", "5", "6", "7", "8", "9", "10"]
```

如果我们要找出其中的偶数，并得到一个整数数组怎么做呢？

思路很简单，我们先把strArray变成一个整数数组，然后筛选出其中的偶数就可以了。借助函数式编程，我们可以很容易实现这个过程：

```
let even = stringArray.map {
    Int($0)!
}.filter {
    $0 % 2 == 0
}
```

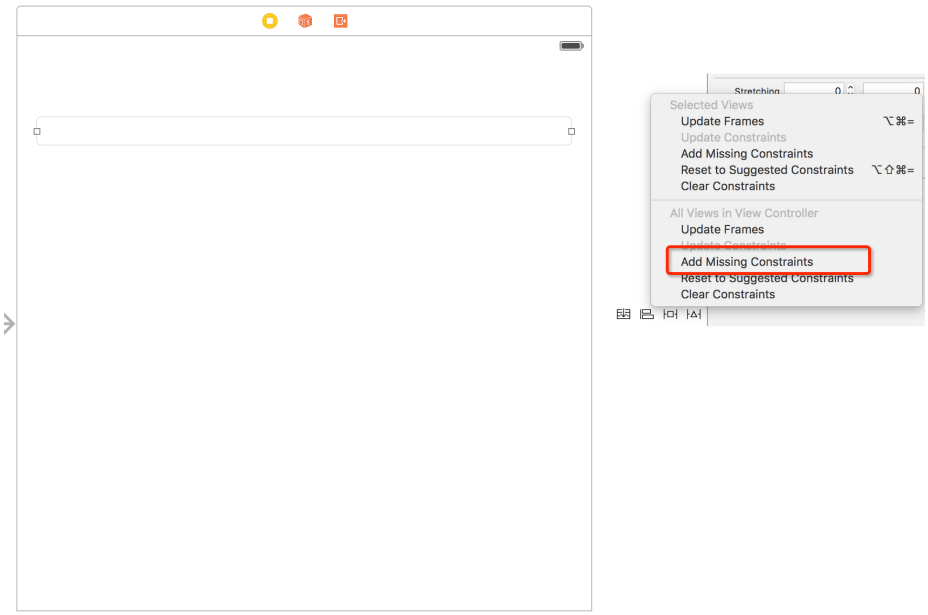
得到结果后，我们把它打印到控制台上：

```
print(evenArray)
```

然后，Command + R编译执行，我们就能在控制台看到对应的结果了：

找到用户输入字符中的偶数

接下来，我们打开main.storyboard，从Object library里拖一个UITextField进来，并使用Xcode给它自动添加约束：



这次，如果我们希望自动获得用户的输入，并且只把用户输入的偶数打印在控制台上，怎么做呢？

最直接的想法当然是使用UITextField的一个delegate响应用户输入事件：

1. 把输入的字符转换成整数；
2. 筛选出可以被2整除的输入；

首先，我们给ViewController添加一个extension：

```
extension ViewController:
    UITextFieldDelegate {
}
```

然后，实现 `textField(_: UITextField, shouldChangeCharactersInRange range: NSRange, replacementString string: String)` 方法。这个方法会在每次用户按下键盘，字母在屏幕上显示出来之前被调用：

```
public func textField(textField: UITextField,
    shouldChangeCharactersInRange range: NSRange,
    replacementString string: String) -> Bool {
        // 1. Map input to Int
        if let n = Int(string) {
            // 2. Filter even numbers
            if n % 2 == 0 {
                print(n)
            }
        }

        return true
}
```

实现的逻辑很简单，我们把用户每次的输入转换成一个整数，然后判断它是否可以被2整除。

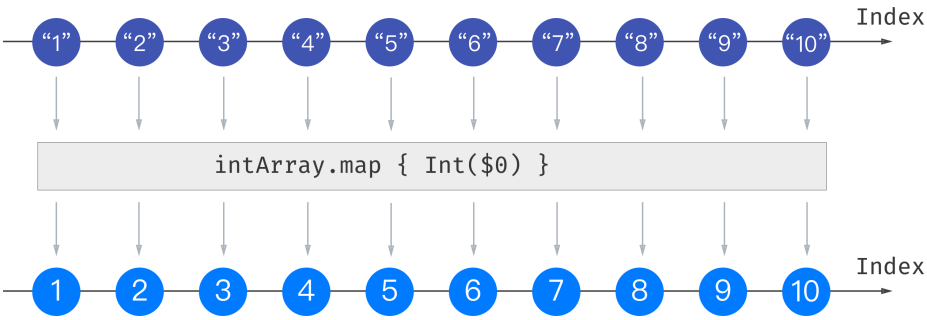
完成之后，Command + R编译执行，输入不同的数字，我们就可以在控制台看到只有偶数被输出了：

为什么要举这两个例子呢?

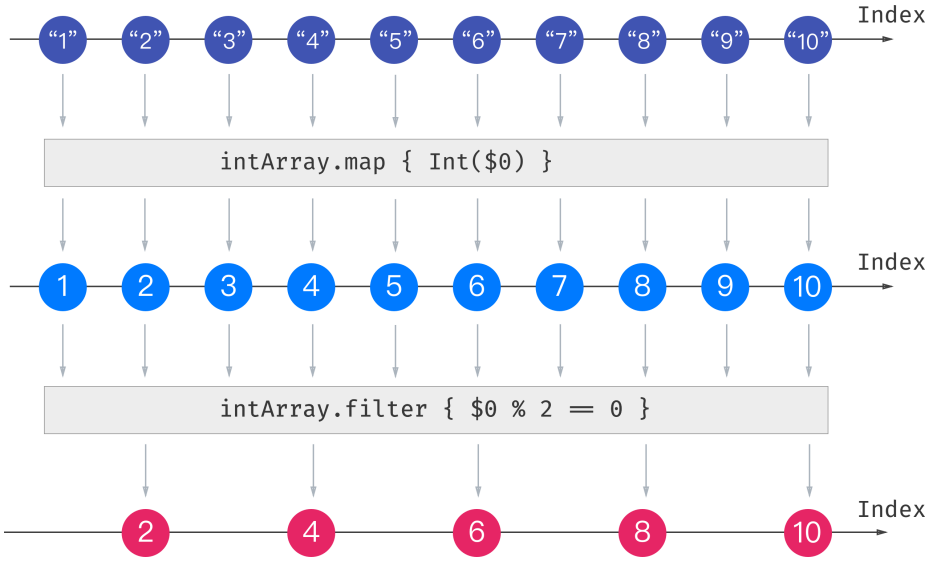
常量数组是一个以索引为维度的串行结构

首先，对于常量数组来说，它是一个串行的，以索引为维度的内容序列，因此，我们对它的处理逻辑也是串行的。

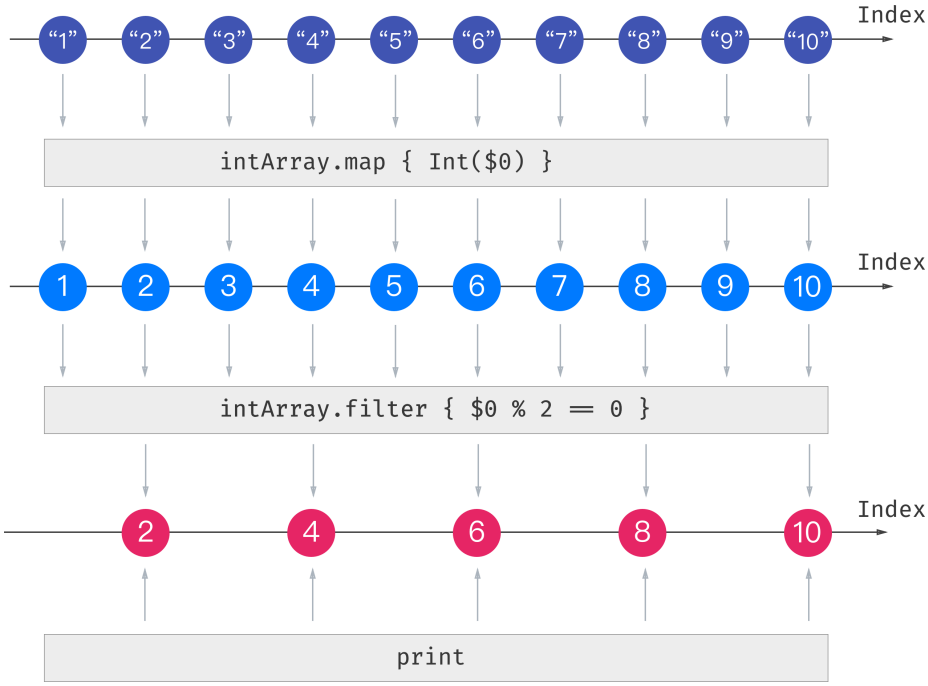
我们先把数组中每个元素变成整数：



然后再逐个筛选中偶数；

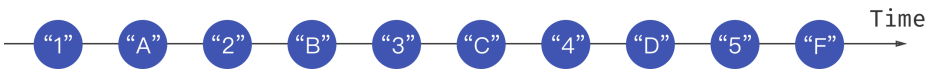


对于得到的结果，我们使用 `print` 函数把它打印到了终端上。我们可以把print语句理解为这是我们对这个结果的某种"订阅"行为（一旦得到了某个偶数数组，就把它输出到控制台上）。

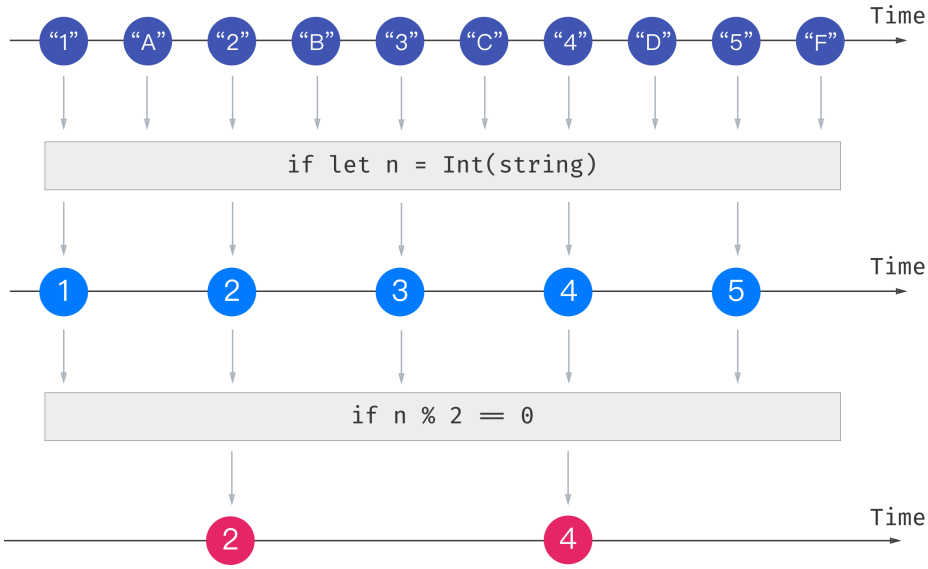


异步事件是一个以时间为维度的串行结构

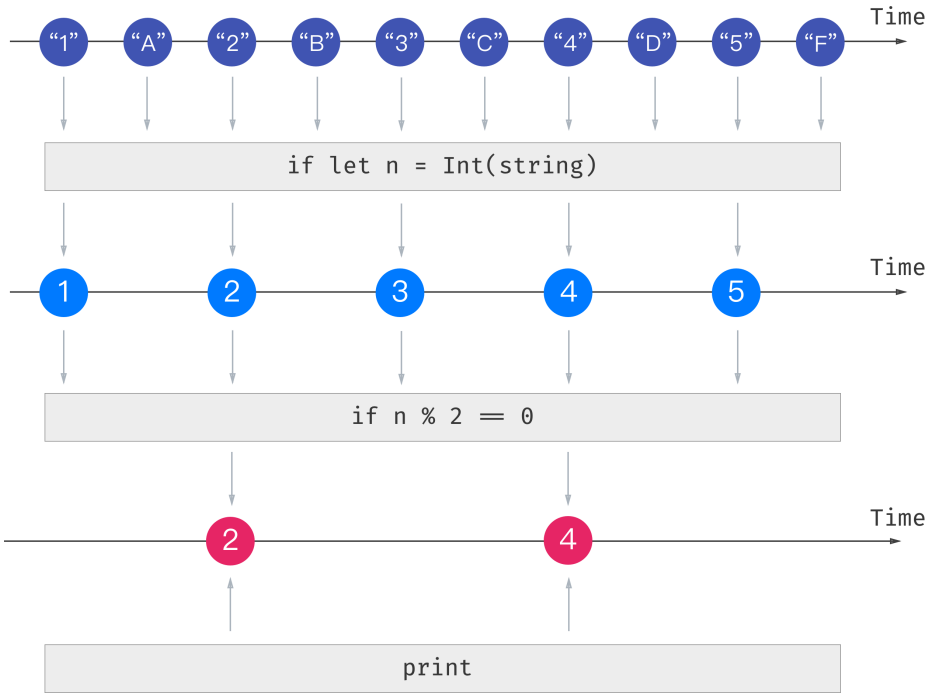
接下来，对于 `UITextField` 来说，尽管用户输入是异步的，但如果我们以用户输入发生的时间作为维度，把所有已经发生的用户输入事件放在时间轴上，不难发现，我们也可以把得到的结果理解为是一个"常量数组"（因为过去已经发生的输入事件是不能被修改的）。



然后我们把这个数组中的元素先转换为整数，然后筛选出其中的偶数，就能得到我们想要的结果了。



因此，似乎存在一种可能性，我们可以定义一个以时间为维度的数组，它的元素是异步发生的事件。当事件发生时，我们就把它自动添加到这个数组里。然后，我们可以做两个事情，一方面，可以对添加进来的事件进行"二次加工"，筛选出符合我们要求的事件形态；另一方面，我们还可以添加对应事件的处理方法，这也就是某种形式的"订阅"。



这看似玄幻，实则它就是Reactive Programming的重要编程思想。理解它，也是我们使用Reactive Programming的最大挑战。

接下来?

我们通过把异步事件和一个常量数组进行对比，理解了Reactive Programming的思考方式：*用同步的方式，编写处理异步事件的代码*。

在下一段视频中，我们将以这个思想为基础：向大家进一步介绍Reactive Programming用到的基本概念和用法：

- 如何定义"以时间为维度的数组"?
- 如何处理加工"事件数组"中的元素?
- 如何"订阅"事件?

不得不说，这些概念和名字同样给我们理解Reactive Programming带来了一些麻烦。

返回视频 (/series/reactive-programming-in-swift)

Hello world in RxSwift (https://www.boxueio.com/series/reactive-programming-in-swift/ebook/74)

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/)