作为构建大规模、高流量应用程序的平台，web已经迅速流行起来。然而，现代web应用程序在计算世界中有些独特，因为它们需要大量的异步，包括AJAX请求、动画、延迟加载的客户机资源和多路复用的web套接字。所有这些异步都伴随着复杂性的代价。

例如，一个简单的拖放实际上是三个或更多不同事件的协调:等待鼠标放下，然后听鼠标所有的移动，直到下一个鼠标升起。当前实现这类事情的命令式方法并不总是直截了当;它们很难维护，而且很少有bug是免费的。

xJS是一个理想的工具，可以帮助您以一种声明性的、易于维护的和有趣的方式管理应用程序中的异步复杂性。那么如何学习Rx呢?

这本名为《RxJS in Action》的书是迄今为止它的类型中涵盖最新版本RxJS 5的唯一资源。作为RxJS的项目负责人，我非常高兴地看到这本书向大众提供了你需要知道的关于这个库的重要信息，这样你才能成为一个有效的反应型程序员。

我们写这本书是为了帮助您理解在JavaScript中响应式编程的能力和重要性，并开发使用RxJS的技能。

PAUL说:像许多技能一样，RxJS并不是我一开始就想学习的东西，而是我偶然发现并继续学习的东西，只是由于一些事件的汇集。在我早期的编程生涯中，我一直在为一个内部工具开发一个新的UI系统，在这个系统中，项目所有者给了我很大的自由来决定我使用什么技术。因为唯一的限制是它是用。net编写的，所以我被引入了Rx。NET首先尝试构建各种UI交互。在大学期间和之后，我的经历主要是面向对象的。我理解了单例模式、装饰器、适配器等，并且听说过一种虚构的编程范例，它似乎与直觉相反，侧重于函数。但对我来说，这似乎完全是倒退。我忽略了更大的上下文，愉快地将新库插入到我的应用程序中，认为它是现有. net事件系统的简单替代品。

在后来的一个项目中，当我第一次真正开始使用JavaScript时，我回想起我使用过的使我的事件管理变得如此顺畅的库，于是我开始寻找JavaScript中的并行程序。你看，RxJS进入了我的生活!一开始，我很高兴地在任何可能的地方插入它，把它看作是替换丑陋的事件回调的一种方法。RxJS可以实现的所有功能以及反应式函数式编程的好处慢慢显现出来。首先，我收到了家人赠送的一本Clojure上的书，这本书让我对这个神秘的功能世界有了一些了解。随着我对异步模式的深入研究，以及对promise等其他异步结构的相似之处进行了扩展。

就在这个时候，当我越来越多地参与到RxJS社区中来(主要是通过在RxJS 2的旧日子里对开源库的贡献来实现的)，Manning找到了我，想写一本关于RxJS的书。这是一项相当大的事业，尤其是对我这个第一次写作的人来说。但它预先给了我回报整个社区的能力，这是我刚开始学习Rx时所没有的，因为大部分资源主要集中在Rx.NET上。我很幸运，有路易斯和我一起做这个项目，使它不那么令人畏惧。

我想让其他人了解RxJS，所以我决定在我的第一本书《JavaScript函数式编程》(Functional Programming in JavaScript)的最后，用一节介绍反应式编程和RxJS可观察性。但我的热情并没有就此停止。几个月后，曼宁找到了我，我很荣幸能与保罗搭档，我发现他是一位才华横溢的工程师，共同撰写了这本书。就这样，我又开始写我最喜欢的两个主题:JavaScript和函数式编程。