初识RxJS--基础铺垫篇

一个初识觉得很鸡肋，但是看完后，觉得很棒的前端工具库。

在介绍这个工具库前，我们先来说两种模式。分别是pull拉取模式和push推送模式。

pull拉取模式和push推送模式都是数据传输交流的方式。但是不同点在于，pull拉取模式更注重的是数据的消费者。而push推送模式则相反，它更注重的是数据的生产者。

在pull模式下，数据的消费者决定何时从数据的生产者处活的数据，但是数据的生产者并不知道消费者什么时候会来获取这些数据。在我们日常开发中，最长用的pull模式例子就是函数。函数对于数据的处理，并且返回处理结果，我们可以将这个函数看做一个数据的生产者。而当我们调用这个函数的时候，得到了数据的返回值，那我们的调用其实就是对于数据结果的一种消费，可以称我们的调用是数据的消费者。很明显，我们的函数并不知道自己在什么时候会被调用执行，一切都取决于我么数据的消费者去调用执行这个函数，因此它是一种最典型的pull模式。

那么相反，push模式，如果用过git的同学，应该都知道这个命令，它将我们的提交推送到远程仓库。我们可以看做是对数据的一种发送。push模式和这个命令其实相似。它是由数据的生产者主动的将数据发送给消费者，但是消费者并不清楚自己何时可以获取到数据。在这里，也有很多常见的例子。比如一个Promise对象。我们通过调用Promise对象的then函数，去获取Promise推送给我们的值，但是我们并不知道何时才能获取到这个值。今天所聊得RxJS其实就是一种push模式的代表。由数据的生产者主动的去推送数据。

相比较Promise对象来说，RxJs有一大优势。就是它会不停的推送我们的数据值，它允许自己的数据生产者有多个值的产出。我们都知道，无论是函数还是Promise对象，它只会返回一个值，要么我们用return返回函数的值，要么我们用resovle或者reject返回一个Promise的值。这一点，RxJs就要更胜一筹了。

那么究竟什么是RxJs?按照官方的总结，它是一个通过使用可观察对象序列来编写异步和基于事件的程序库。在这里面，主要包含着几个概念：观察者模式、迭代器模式、使用集合的函数式编程、函数式编程。简单概括一下这几个概念：

* 观察者模式：通过一个对象管理所有依赖于它的对象，并且在自身发生变化的时候主动的向依赖于它的对象发送通知。一般通过调用依赖于它的对象所提供的某种方法来实现通知流程。这种模式一般被用来实时事件处理系统。
* 迭代器模式：让用户通过特定的接口巡访容器中的每一个元素而不需要了解其底层实现。例如foreach循环。它可以作用于对象或者数组。但是作用在对象或者数组时，它的回调函数中所获得到的参数又是不同的。可以说foreach就是一个迭代器，我们通过对象或者数组的foreach方法获取到对象或数组中的具体值，但是我们并不需要关心foreach是如何实现的。
* 函数式编程：函数式编程是使用集合的函数式编程的一个基础，所以我们先聊一下函数式编程的概念。它是一种将复杂的执行过程拆分到细小的部分，通过简单的执行逻辑计算每一步的结果，然后将计算结果传递给下一个函数，从而一步步的演进，得出最终的结果。
* 使用集合的函数式编程：首先它是一个数据集合，其次，它通过函数式编程的模式，将一些细小的逻辑操作得到的值传递给下一个逻辑操作，从而算出整个集合的结果。RxJs就是这样的一种模式。

通过了解这些概念，我们先对RxJs做一个简单的概括。

1. 它是一个数据集合。
2. 它采用函数式编程的方式，通过拆分成一些细小的逻辑计算方式，一步步的演进，得出最终结果。
3. 它可以不停的去计算、推送得到的结果给数据消费者。
4. 它采用观察者模式，一旦自身的数据集合发生了改变，会立即推送消息给数据消费者。

后面我会逐步介绍RxJs的一些其他基础概念。在了解了它的一个基本实现逻辑后，其他的基础概念也就很好理解了。同学们，期待吧！哈哈。

自恋一下。如果有其他见解，欢迎留言。最后，附上一些本篇涉及到的文档资料地址。