在RxJava的模型世界里，有四种角色：

1. Observable

表示某个对象是可观察的。也就是生产者角色。

1. Observer

观察者，消费者角色。

1. Subscriber

订阅者，消费者角色。

1. Subject

订阅对象，某个对象可被订阅，说明它是生产者角色。

它们的内在联系如下：

1. 我们先创建一个可观察的对象Observable；
2. 创建一个订阅者Observer；
3. 观察者订阅观察对象；
4. 观察者向观察对象订阅的内容就是订阅（Subscription）。
5. Subject是这样一种实体：它可以被观察者订阅，而可以通过这个对象手动调用Subject提供的next()方法来向观察者发送数据，也就是说使用Subject，我们可以在任何我们想要的时候去通知订阅者并向它传送数据；

我们可以使用伪代码来描述一下：

Observable observable = Observable.create();

Observer obserber = new Observer();

Subscription subscription = observable.subscribe(obserber);

Observable提供了三种类型的方法可以创建一个可观察的对象：

第一种是create()方法，这种一般是没有数据源需要自己创建数据源的时候调用；

第二种是from()方法，这种一般是有数据源（列表、数组或者future对象），我们直接使用from方法传入数据源创建；

第三种方法是just()方法，这种一般是已知需要向订阅者发送的对象，可以使用此方法生成Observable对象；just()方法支持最多10个传入对象，需要注意的是，Observable向Observer发送订阅对象的时候，其顺序就是我们just传入的参数顺序；

当我们需要一个Observable毫无理由的不再发射数据正常结束时，我们可以使

用 empty() 。我们可以使用 never() 创建一个不发射数据并且也永远不会结束

的Observable。我们也可以使用 throw() 创建一个不发射数据并且以错误结束的

Observable。

RxJava提供了四种Subject：

1. PublishSubject 这是一个基础Subject
2. BehaviorSubject 简单的说，BehaviorSubject会首先向他的订阅者发送截至订阅前最新的一个数据对象（或初始值）,然后正常发送订阅后的数据流。

3） ReplaySubject 它会缓存它所订阅的所有数据,向任意一个订阅它的观察者重发:

4） AsyncSubject 当Observable完成时AsyncSubject只会发布最后一个数据给已经订阅的每一个观察者。

