**Android RxJava**

参考：<http://gank.io/post/560e15be2dca930e00da1083>

<https://www.jianshu.com/p/6a6f7a4be38d>

<https://blog.csdn.net/carson_ho/article/details/79168723>

1. RxJava是一个实现java响应式编程的库，让异步的、基于事件的程序以可观察序列的形式组织起来。
2. 示例代码

Observable.*create*(new ObservableOnSubscribe<String>(){  
 @Override  
 public void subscribe(ObservableEmitter<String> e) throws Exception {}  
}).map(new Function<String, Bitmap>() {  
 @Override  
 public Bitmap apply(@io.reactivex.annotations.NonNull String s) throws Exception {  
 return null;  
 }  
})  
.subscribeOn(Schedulers.*io*())  
.observeOn(AndroidSchedulers.*mainThread*())  
.subscribe(new Observer<Bitmap>() {  
 @Override  
 public void onSubscribe(Disposable d) {  
  
 }  
 @Override  
 public void onNext(Bitmap s) {}  
 @Override  
 public void onError(Throwable e) {}  
 @Override  
 public void onComplete() {}  
});

1. 执行流程
2. 通过create()将自定义观察者observable封装成ObservableCreate；
3. subscribe()中传入观察者observer实现订阅；
4. subscribe()方法中调用被观察ObservableCreate#subscribeActual()，ObservableCreate#subscribeActual()先调用观察者observer#onSubscribe()，然后调用自定义的被观察observable#subscribe ()
5. just
6. from
7. 变换
8. map()
9. flapMap()
10. lift()
11. 线程切换
12. 背压：一种 控制事件流速 的策略，在 异步订阅关系 中，控制事件发送&接收的速度

<https://www.jianshu.com/p/ceb48ed8719d>

通过缓存区队列（默认长度128）控制被观察者的事件发送速度，通过设置FlowableEmitter#request()控制观察者的事件接受速度。