

# 赵彦军

做一只快乐的程序猿！！

博客园 首页 新随笔 联系 管理

随笔- 210 文章- 0 评论- 86

昵称: 赵彦军  
园龄: 3年2个月  
粉丝: 89  
关注: 10  
[+加关注](#)

## RxJava 和 RxAndroid 一 (基础)

### 1、RxJava 项目地址

<https://github.com/ReactiveX/RxJava>

### 2、RxAndroid 项目地址

<https://github.com/ReactiveX/RxAndroid>

### 3、RxJava 和 RxAndroid 的关系

RxAndroid是RxJava的一个针对Android平台的扩展，主要用于 Android 开发

### 4、RxJava和EventBus的区别？

<https://www.zhihu.com/question/32179258/answer/54989242>

### 5、RxAndroid的使用方法

```
compile 'io.reactivex:rxandroid:1.2.0'
```

### 6、如何查看RxAndroid最新版本？

<http://search.maven.org/#search%7Cga%7C1%7Ca%3A%22rxandroid%22>

### 7、RxAndroid具体使用方法

[http://gank.io/post/560e15be2dca930e00da1083#toc\\_14](http://gank.io/post/560e15be2dca930e00da1083#toc_14)

<http://blog.csdn.net/theone10211024/article/details/50435325>

<http://huxian99.github.io/tags/RxJava/>

<https://github.com/mcxiaoke/RxDocs>

### 8、创建观察者

```
1 package lib.com.myapplication;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5
6 import rx.Observer;
7 import rx.Subscriber;
8
9 public class Main2Activity extends AppCompatActivity {
10
11     @Override
```

## 我的标签

[android\(30\)](#)  
[赵彦军\(26\)](#)  
[zhaoyanjun\(25\)](#)  
[java\(10\)](#)  
[IO流\(8\)](#)  
[android studio\(7\)](#)  
[markdown\(6\)](#)  
[rxandroid\(5\)](#)  
[rxbinding\(5\)](#)  
[rxjava\(5\)](#)  
[更多](#)

## 随笔分类 (349)

[Android\(169\)](#)  
[Android Studio\(100\)](#)  
[Android 错误\(4\)](#)  
[Android 动画](#)  
[android 框架\(6\)](#)  
[CVS \( SVN 和 Git \)\(19\)](#)  
[IOS\(7\)](#)  
[IOS 错误\(1\)](#)  
[Java\(23\)](#)  
[python\(1\)](#)  
[产品与设计\(4\)](#)  
[生活\(3\)](#)  
[网站与服务器\(10\)](#)  
[运营与测试\(2\)](#)

## 随笔档案 (210)

[2017年3月 \(2\)](#)  
[2017年2月 \(9\)](#)  
[2017年1月 \(2\)](#)  
[2016年12月 \(2\)](#)  
[2016年11月 \(6\)](#)  
[2016年10月 \(7\)](#)  
[2016年9月 \(7\)](#)  
[2016年8月 \(7\)](#)  
[2016年7月 \(4\)](#)  
[2016年6月 \(9\)](#)  
[2016年5月 \(8\)](#)  
[2016年4月 \(11\)](#)  
[2016年3月 \(20\)](#)  
[2016年2月 \(8\)](#)  
[2016年1月 \(8\)](#)  
[2015年12月 \(5\)](#)  
[2015年11月 \(6\)](#)  
[2015年10月 \(6\)](#)  
[2015年9月 \(5\)](#)  
[2015年8月 \(12\)](#)  
[2015年7月 \(15\)](#)  
[2015年6月 \(31\)](#)  
[2015年5月 \(19\)](#)  
[2015年4月 \(1\)](#)

## 积分与排名

积分 - 147693

排名 - 1485

## 最新评论

### 1. Re:java 接口的作用和好处

@赵彦军来写个快排。我出面试题的第一个...

--cutd

### 2. Re:android 显示 PDF 文件

@赵彦军 大哥，我又来啦，哈哈哈，你有没有遇到pdfview加载出来的文件，放大后会虚化的问题？

--CurtisWgh

### 3. Re:android 显示 PDF 文件

你好啊，可以麻烦你发送下apk吗？另有几个问题想请教~

190697454@qq.com

--Leep\_H

### 4. Re:android 显示 PDF 文件

@赵彦军 我是在刚开始加载的时候，就会出现pdf完全被黑色覆盖住了，也没找出来问题在哪，哈哈哈

--CurtisWgh

### 5. Re:android 显示 PDF 文件

@CurtisWgh你说的这个问题，我也遇到过，pdf在放大和缩小的时候，也会有黑色斑块出现，工作忙，没有过于深究里面的原因。...

--赵彦军

## 阅读排行榜

1. Java 枚举类的基本使用(29556)
2. Android Butterknife 8.4.0 使用方法总结(23252)
3. RxJava 和 RxAndroid 四（RxBinding的使用）(13642)
4. RxJava 和 RxAndroid 三（生命周期控制和内存优化）(10485)
5. Android Http请求框架二：xUtils 框架网络请求(9236)

## 推荐排行榜

1. Android studio 使用Gradle发布Android开源项目到JCenter 总结(5)
2. 【转载】安卓APP架构(2)
3. Android DEX 基础(2)
4. Android 自定义线程池的实战(2)
5. Android AsyncTask 深度理解、简单封装、任务队列分析、自定义线程池(2)

```

12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_main2);
15
16         //创建观察者 2 种方法
17         Observer<String> observer = new Observer<String>() {
18             @Override
19             public void onCompleted() {
20
21             }
22
23             @Override
24             public void onError(Throwable e) {
25
26             }
27
28             @Override
29             public void onNext(String s) {
30
31             }
32         };
33
34         // Subscriber 继承 Observer , 对Observer类做了扩展
35         Subscriber<String> subscriber = new Subscriber<String>() {
36             @Override
37             public void onCompleted() {
38
39             }
40
41             @Override
42             public void onError(Throwable e) {
43
44             }
45
46             @Override
47             public void onNext(String s) {
48
49             }
50
51         };
52
53
54     }
55 }
```

- 从上文可以看到，Subscriber继承Observer, 只是 Subscriber对Observer做了一些扩展。Subscriber的使用和Observer完全一样。
- Subscriber 多了一个 onStart 方法
- onStart(): 这是 Subscriber 增加的方法。它会在 subscribe 刚开始，而事件还未发送之前被调用，可以用于做一些准备工作，例如数据的清零或重置。这是一个可选方法，默认情况下它的实现为空。需要注意的是，如果对准备工作的线程有要求（例如弹出一个显示进度的对话框，这必须在主线程执行），onStart() 就不适用了，因为它总是在 subscribe 所发生的线程被调用，而不能指定线程。要在指定的线程来做准备工作，可以使用 doOnSubscribe() 方法，具体可以在后面的文中看到。

```

1 // Subscriber 继承 Observer , 对Observer类做了扩展
2     Subscriber<String> subscriber = new Subscriber<String>() {
3         @Override
4         public void onCompleted() {
5
6         }
7
8         @Override
9         public void onError(Throwable e) {
```

```

10
11     }
12
13     @Override
14     public void onNext(String s) {
15
16     }
17
18     @Override
19     public void onStart() {
20         super.onStart();
21     }
22 } ;

```

## 9、创建被观察者

```

1 //create方式
2 Observable<String> observable = Observable.create(new Observable.OnSubscribe<String>() {
3     @Override
4     public void call(Subscriber<? super String> subscriber) {
5         subscriber.onNext( "aa" );
6         subscriber.onNext( "bb" );
7         subscriber.onNext( "cc" );
8         subscriber.onCompleted();
9     }
10 });
11
12 //just方式 最多支持10个数据
13 Observable<String> observable1 = Observable.just( "aa" , "bb" , "cc" );
14 // 将会依次调用:
15 // onNext("aa");
16 // onNext("bb");
17 // onNext("cc");
18 // onCompleted();
19
20 //from方式
21 //1:集合
22 List<String> list = new ArrayList<>() ;
23 list.add( "aa" );
24 list.add( "bb" );
25 list.add( "cc" );
26
27 Observable<String> observable2 = Observable.from( list );
28
29 //2: 数组
30 String[] words = { "aa", "bb", "cc" };
31 Observable<String> observable3 = Observable.from( words );

```

- Call()方法: 当 Observable 被订阅的时候, OnSubscribe 的 call() 方法会自动被调用, 事件序列就会依照设定依次触发 (对于上面的代码, 就是观察者Subscriber 将会被调用三次 onNext() 和一次 onCompleted())。这样, 由被观察者调用了观察者的回调方法, 就实现了由被观察者向观察者的事件传递, 即观察者模式。

## 10、订阅

由于观察者可以由两种方式被创建, 所以订阅的方式也有两种

```

1 observable.subscribe( observer );
2 observable.subscribe( subscriber );

```

- Observable.subscribe(Subscriber) 的内部实现是这样的 (仅核心代码) :

```

1 // 注意: 这不是 subscribe() 的源码, 而是将源码中与性能、兼容性、扩展性有关的代码剔除后的核心代码
2 // 如果需要看源码, 可以去 RxJava 的 GitHub 仓库下载。

```

```

3 public Subscription subscribe(Subscriber subscriber) {
4     subscriber.onStart();
5     onSubscribe.call(subscriber);
6     return subscriber;
7 }

```

1. 在subscribe() 中，首先会调用 onStart() 方法，这个方法前文已经介绍了，是可选的。接着会调用 call()方法，我们已经分析了在call()方法中会调用多次 onNext()，最后调用 onCompleted().看到这里你就会突然明白原来subscribe() 方法其实相当于依次执行了：onStart() --> onNext () --> onCompleted ()
2. 从这也可以看出，在 RxJava 中，Observable 并不是在创建的时候就立即开始发送事件，而是在它被订阅的时候，即当 subscribe() 方法执行的时候。
3. Observer 和 Subscriber 具有相同的角色，而且 Observer 在 subscribe() 过程中最终会被转换成 Subscriber对象
4. 将传入的 Subscriber 作为 Subscription 返回。这是为了方便 unsubscribe().

## 11、RxBus

你是否听说过EventBus，他是android 中的事件总线。用rxjava同样可以实现android的事件总线功能，也就是RxBus。

关于rxbus 的基本说明在这

里 <http://nerds.weddingpartyapp.com/tech/2014/12/24/implementing-an-event-bus-with-rxjava-rxbus/>

然而这并没有什么卵用！

下面是RxBus的封装版

```

1 package lib.com.myapplication;
2 import android.support.annotation.NonNull;
3 import android.util.Log;
4 import java.util.ArrayList;
5 import java.util.Collection;
6 import java.util.List;
7 import java.util.concurrent.ConcurrentHashMap;
8 import rx.Observable;
9 import rx.subjects.PublishSubject;
10 import rx.subjects.Subject;
11
12 /**
13  * Created by ${zyj} on 2016/5/6.
14  */
15 public class RxBus {
16
17     private static final String TAG = RxBus.class.getSimpleName();
18     private static RxBus instance;
19     public static boolean DEBUG = false;
20
21     public static RxBus get() {
22         if (instance == null) {
23             synchronized (RxBus.class) {
24                 if (instance == null) {
25                     instance = new RxBus();
26                 }
27             }
28         }
29         return instance;
30     }
31
32     private RxBus() {
33     }
34

```

```

35     private ConcurrentHashMap<Object, List<Subject>> subjectMapper = new ConcurrentHa:
36
37     @SuppressWarnings("unchecked")
38     public <T> Observable<T> register(@NonNull Object tag, @NonNull Class<T> clazz) {
39         List<Subject> subjectList = subjectMapper.get(tag);
40         if (null == subjectList) {
41             subjectList = new ArrayList<>();
42             subjectMapper.put(tag, subjectList);
43         }
44
45         Subject<T, T> subject;
46         subjectList.add(subject = PublishSubject.create());
47         if (DEBUG) Log.d(TAG, "[register]subjectMapper: " + subjectMapper);
48         return subject;
49     }
50
51     public void unregister(@NonNull Object tag, @NonNull Observable observable) {
52         List<Subject> subjects = subjectMapper.get(tag);
53         if (null != subjects) {
54             if (observable != null && subjects.contains(observable)){
55                 subjects.remove((Subject) observable);
56             }
57
58             if (isEmpty(subjects)) {
59                 subjectMapper.remove(tag);
60             }
61         }
62
63         if (DEBUG) Log.d(TAG, "[unregister]subjectMapper: " + subjectMapper);
64     }
65
66     public void post(@NonNull Object content) {
67         post( content.getClass().getName(), content);
68     }
69
70     @SuppressWarnings("unchecked")
71     public void post(@NonNull Object tag, @NonNull Object content) {
72         List<Subject> subjectList = subjectMapper.get(tag);
73
74         if (!isEmpty(subjectList)) {
75             for (Subject subject : subjectList) {
76                 subject.onNext(content);
77             }
78         }
79         if (DEBUG) Log.d(TAG, "[send]subjectMapper: " + subjectMapper);
80     }
81
82     private boolean isEmpty(Collection collection) {
83         return null == collection || collection.isEmpty();
84     }
85 }

```

#### RxBus的使用

```

1  package lib.com.myapplication;
2  import android.os.Bundle;
3  import android.support.v7.app.AppCompatActivity;
4  import rx.Observable;
5  import rx.functions.Action1;
6
7  public class Activity1 extends AppCompatActivity {
8
9      String tag = "tag" ;
10     Observable<String> ob ;
11
12     @Override

```

```
13     protected void onCreate(Bundle savedInstanceState) {
14         super.onCreate(savedInstanceState);
15         setContentView(R.layout.activity1);
16
17         //创建被观察者
18         ob = RxBus.get().register( tag , String.class );
19         //订阅观察事件
20         ob.subscribe(new Action1<String>() {
21             @Override
22             public void call(String s) {
23                 System.out.println( "fff-- " + s );
24             }
25         });
26
27         //发送内容
28         RxBus.get().post( tag , "我是内容" );
29     }
30
31     @Override
32     protected void onDestroy() {
33         super.onDestroy();
34         //取消订阅
35         RxBus.get().unregister( tag , ob );
36     }
37 }
```

- 在Activity销毁的时候，要取消订阅服务。否则 post() 次数会随着post()调用逐渐增加
- 除了上面的简单使用外，还可以使用 **Schedulers**、**AndroidSchedulers** 进行线程切换

## [RxJava 和 RxAndroid 二 \(操作符的使用\)](#)

分类: [Android](#), [Android Studio](#)

标签: [rxjava](#), [rxandroid](#), [rxjava操作符](#), [markdown](#), [赵彦军](#), [zhaoyanjun](#), [android](#), [rxbinding](#)



赵彦军  
关注 - 10  
粉丝 - 89

0

0

[+加关注](#)

« 上一篇: [Android studio 如何查看当前git 分支的4种方式](#)

» 下一篇: [Android studio .gitignore 文件的内容](#)

posted @ 2016-02-01 15:32 [赵彦军](#) 阅读(7908) 评论(0) [编辑](#) [收藏](#)

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

【推荐】50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库

【推荐】群英云服务器性价比王, 2核4G5M BGP带宽 68元首月!

【福利】阿里云免费套餐升级, 更多产品, 更久时长



**最新IT新闻:**

- GoPro发布一款VR相机，很专业很小巧
  - Apple TestFlight支持iOS App的A/B测试
  - 天舟一号与天宫二号12时23分顺利完成自动交会对接
  - 美国知乎Quora融资8500万美元：估值18亿刀 向国际扩张
  - Old Computer Museum是展示旧技术辉煌过去的门户
- » 更多新闻...



**最新知识库文章:**

- 唱吧DevOps的落地，微服务CI/CD的范本技术解读
  - 程序员，如何从平庸走向理想？
  - 我为什么鼓励工程师写blog
  - 怎么轻松学习JavaScript
  - 如何打好前端游击战
- » 更多知识库文章...

Copyright ©2017 赵彦军