链滴

RxJava 操作符之 -- 算数操作符

作者：hiquanta
原文链接：https://ld246.com/article/1490866784057
来源网站：链滴

Average

<!--more-->

计算原始Observable发射数字的平均值并发射它: 它属于不同的rxjava-math模块

collect

Collect操作符类似于Reduce，但是其目的不同，collect用来将源Observable发射的数据给收集到一数据结构里面，需要使用两个参数：

1. 一个产生收集数据结构的函数。
2. 一个接收第一个函数产生的数据结构和源Observable发射的数据作为参数的函数。

```
List<Integer> list=new ArrayList<>();
        list.add(1);
        list.add(1);
        list.add(1);
        list.add(1);
        list.add(1);
        Observable.from(list).collect(() -> new ArrayList<>(), new Action2<List<Integer>, Intege
>() {

            @Override
            public void call(List<Integer> t1, Integer t2) {
                t1.add(t2);

            }
        }).subscribe(new Action1<List<Integer>>() {

            @Override
            public void call(List<Integer> t) {
                System.out.println(t.toString());

            }


        });
```

结果

```
[1, 1, 1, 1, 1]
```

Concat

不交错的发射两个或多个Observable的发射物

```
Observable<Integer> obser1 = Observable.just(1, 2, 3);
        Observable<Integer> obser2 = Observable.just(4, 5, 6);
        Observable<Integer> obser3 = Observable.just(7, 8, 9);
```

```
Observable.concat(obser1,obser2,obser3).subscribe(new Action1<Integer>() {

    @Override
    public void call(Integer t) {
        System.out.println(t);
    }
});
```

结果

```
1
2
3
4
5
6
7
8
9
```

Count

Max

Min

Sum

Reduce

按顺序对Observable发射的每项数据应用一个函数并发射最终的值

```
Observable.just(1,2,3,9).reduce(new Func2<Integer, Integer, Integer>() {

    @Override
    public Integer call(Integer t1, Integer t2) {
      // TODO Auto-generated method stub
      return t1+t2;
    }
  }).subscribe(new Action1<Integer>() {

    @Override
    public void call(Integer t) {
      System.out.println(t);

    }
  });
```

结果