链滴

# RxJava2.0 操作符之 -- 过滤操作符

作者：hiquanta

原文链接：https://ld246.com/article/1500286537190

来源网站：链滴

# debounce

## 仅在过了一段指定的时间还没发射数据时才发射一个数据

```
 Observable.create(new ObservableOnSubscribe() {
   public void subscribe(@NonNull ObservableEmitter e) throws Exception {
       try {
          //产生结果的间隔时间分别为100、200、300...900毫秒
 for (int i = 1; i < 100; i++) {
              e.onNext(i);
 Thread.sleep(i * 100);
 }
          e.onComplete();
 }catch(Exception ex){
          e.onError(ex);
 }
   }
}).debounce(500, TimeUnit.MILLISECONDS).subscribe(RxUtils.getObserver());
try {
    Thread.sleep(Integer.MAX_VALUE);
} catch (InterruptedException e1) {
    e1.printStackTrace();
}
```

运行结果

```
onSubscribe
Thread:Thread[main,5,main]
onNext:6
Thread:Thread[RxComputationThreadPool-1,5,main]
onNext:7
Thread:Thread[RxComputationThreadPool-1,5,main]
onNext:8
Thread:Thread[RxComputationThreadPool-1,5,main]
onNext:9
```

# distinct

## 抑制（过滤掉）事件源发出的重复的数据项

```
Observable.just(1, 2, 1, 1, 2, 3,3,3,5).distinct().subscribe(RxUtils.getObserver());
```

```
    onSubscribe
    Thread:Thread[main,5,main]
    onNext:1
    Thread:Thread[main,5,main]
    onNext:2
    Thread:Thread[main,5,main]
    onNext:3
    Thread:Thread[main,5,main]
```

```
onNext:5
Thread:Thread[main,5,main]
onComplete
Thread:Thread[main,5,main]
```

# elementAt

## 只发射第N项数据

```
Observable observable=  Observable.just(1,2,3,4,5,6).elementAt(2).toObservable();
observable.subscribe(RxUtils.getObserver());

onSubscribe
Thread:Thread[main,5,main]
onNext:3
Thread:Thread[main,5,main]
onComplete
Thread:Thread[main,5,main]
```

# filter

## 只发射通过了谓词测试的数据项

```
Observable.just(1, 2, 3, 4, 5).filter(new Predicate() {
  public boolean test(@NonNull Integer integer) throws Exception {
      return integer>3;
  }
}).subscribe(RxUtils.getObserver());

onSubscribe
Thread:Thread[main,5,main]
onNext:4
Thread:Thread[main,5,main]
onNext:5
Thread:Thread[main,5,main]
onComplete
Thread:Thread[main,5,main]
```

# first

## 只发射第一项（或者满足某个条件的第一项）数据

```
Observable.just(1, 2, 3).first(5).toObservable().subscribe(RxUtils.getObserver());

onSubscribe
Thread:Thread[main,5,main]
onNext:1
Thread:Thread[main,5,main]
onComplete
Thread:Thread[main,5,main]
```

# ignoreElements

## 不发射任何数据，只发射Observable的终止通知

```
Observable.just(1,2,3).ignoreElements().toObservable().subscribe(RxUtils.getObserver());
```

```
onSubscribe
Thread:Thread[main,5,main]
onComplete
Thread:Thread[main,5,main]
```

# last

## 只发射最后一项（或者满足某个条件的最后一项）数据

```
Observable.just(1, 2, 3).last(5).toObservable().subscribe(RxUtils.getObserver());
```

```
onSubscribe
Thread:Thread[main,5,main]
onNext:3
Thread:Thread[main,5,main]
onComplete
Thread:Thread[main,5,main]
```

# sample

## 定期发射Observable最近发射的数据项

```
Observable.create(new ObservableOnSubscribe() {
    public void subscribe(@NonNull ObservableEmitter e) throws Exception {
        try {
            // 前8个数字产生的时间间隔为1秒，后一个间隔为3秒
  for (int i = 1; i < 9; i++) {
                e.onNext(i);
  Thread.sleep(1000);
  }
            Thread.sleep(2000);
  e.onNext(9);
  e.onComplete();
  } catch (Exception ex) {
            e.onError(ex);
  }
    }
}).sample(2200, TimeUnit.MILLISECONDS) .subscribe(RxUtils.getObserver());
try {
    Thread.sleep(Integer.MAX_VALUE);
} catch (InterruptedException e1) {
    e1.printStackTrace();
}
```

```
Thread:Thread[main,5,main]
onNext:3
Thread:Thread[RxComputationThreadPool-1,5,main]
onNext:5
Thread:Thread[RxComputationThreadPool-1,5,main]
onNext:7
Thread:Thread[RxComputationThreadPool-1,5,main]
onNext:8
Thread:Thread[RxComputationThreadPool-1,5,main]
onComplete
```

# skip

# 抑制Observable发射的前N项数据

```
Observable.just(1,2,3,4,5,6,7,8).skip(3).subscribe(RxUtils.getObserver());
```

```
onSubscribe
Thread:Thread[main,5,main]
onNext:4
Thread:Thread[main,5,main]
onNext:5
Thread:Thread[main,5,main]
onNext:6
Thread:Thread[main,5,main]
onNext:7
Thread:Thread[main,5,main]
onNext:8
Thread:Thread[main,5,main]
onComplete
Thread:Thread[main,5,main]
```

# skipLast

# suppress the final $n$ items emitted by an Observable

```
Observable.just(1, 2, 3, 4, 5, 6, 7).skipLast(3).subscribe(RxUtils.getObserver());
```

```
onSubscribe
Thread:Thread[main,5,main]
onNext:1
Thread:Thread[main,5,main]
onNext:2
Thread:Thread[main,5,main]
onNext:3
Thread:Thread[main,5,main]
onNext:4
Thread:Thread[main,5,main]
onComplete
Thread:Thread[main,5,main]
```

# Take

# emit only the first n items emitted by an Observable

```
Observable.just(1,2,3,4,5,6,7,8).take(3).subscribe(RxUtils.getObserver());
```

```
 onSubscribe
Thread:Thread[main,5,main]
onNext:1
Thread:Thread[main,5,main]
onNext:2
Thread:Thread[main,5,main]
onNext:3
Thread:Thread[main,5,main]
onComplete
Thread:Thread[main,5,main]
```

# takeLast

# 发射Observable发射的最后N项数据

```
Observable.just(1,2,3,4,5,6,7,8).takeLast(3).subscribe(RxUtils.getObserver());
```

```
onSubscribe
Thread:Thread[main,5,main]
onNext:6
Thread:Thread[main,5,main]
onNext:7
Thread:Thread[main,5,main]
onNext:8
Thread:Thread[main,5,main]
onComplete
Thread:Thread[main,5,main]
```