迭代器模式、观察者模式

Rxjava的异步实现，是通过扩展的观察者模式实现的

Flatmap

Filter

Map

Subscribe 订阅

Subscribeon

Observer 观察者

Observable 可观察者，被观察者

Observeon

事件回调方法：

onNext

onCompleted

onError

基本实现：

1. 创建observer 处理在事件触发的时候的行为

Observer<String> observer = new Observer<String>() {}

（注：Observer 和 Subscriber 是完全一样的）

1. 创建oberveable，决定什么时候触发事件以及触发怎样的事件

Observable observable = Observable.create(new Observable.OnSubscribe<String>() {

@Override

public void call(Subscriber<? super String> subscriber) {

subscriber.onNext("Hello");

subscriber.onNext("Hi");

subscriber.onNext("Aloha");

subscriber.onCompleted();

}

});

just(T...): 将传入的参数依次发送出来

Observable observable = Observable.just("Hello", "Hi", "Aloha");

// 将会依次调用：

// onNext("Hello");

// onNext("Hi");

// onNext("Aloha");

// onCompleted();

* from(T[]) / from(Iterable<? extends T>) : 将传入的数组或 Iterable 拆分成具体对象后，依次发送出来。

String[] words = {"Hello", "Hi", "Aloha"};

Observable observable = Observable.from(words);

// 将会依次调用：

// onNext("Hello");

// onNext("Hi");

// onNext("Aloha");

// onCompleted();

1. 订阅Subscribe

observable.subscribe(observer);

// 或者：

observable.subscribe(subscriber);