HenCoder Plus 第 18 课 讲义

RxJava

基本用法

```
@GET("users/{username}/repos")
Single<List<Repo>> getRepos(@Path("username") String username);

...

api.getRepos("rengwuxian")
        .subscribeOn(Schedulers.newThread())
        .observeOn(AndroidSchedulers.mainThread())
        .subscribe(new SingleObserver<List<Repo>>() {
            @Override
            public void onSubscribe(Disposable disposable) {
                textView.setText("正在请求");
                MainActivity.this.disposable.add(disposable);
            }

            @Override
            public void onSuccess(List<Repo> repos) {
                textView.setText(repos.get(0).name);
            }

            @Override
            public void onError(Throwable e) {
                textView.setText(e.getMessage());
            }
        });
```

框架结构

RxJava 的整体结构是一条链，其中：

1. 链的最上游：生产者 Observable
2. 链的最下游：观察者 Observer
3. 链的中间：各个中介节点，既是下游的 Observable，又是上游的 Observer

操作符 Operator（map() 等等）：

1. 基于原 Observable 创建一个新的 Observable
2. Observable 内部创建一个 Observer

3. 通过定制 Observable 的 subscribeActual() 方法和 Observer 的 onXxx() 方法，来实现自己的中介角色（例如数据转换、线程切换）

Disposable：

可以通过 dispose() 方法来让上游停止工作，达到「丢弃」的效果。

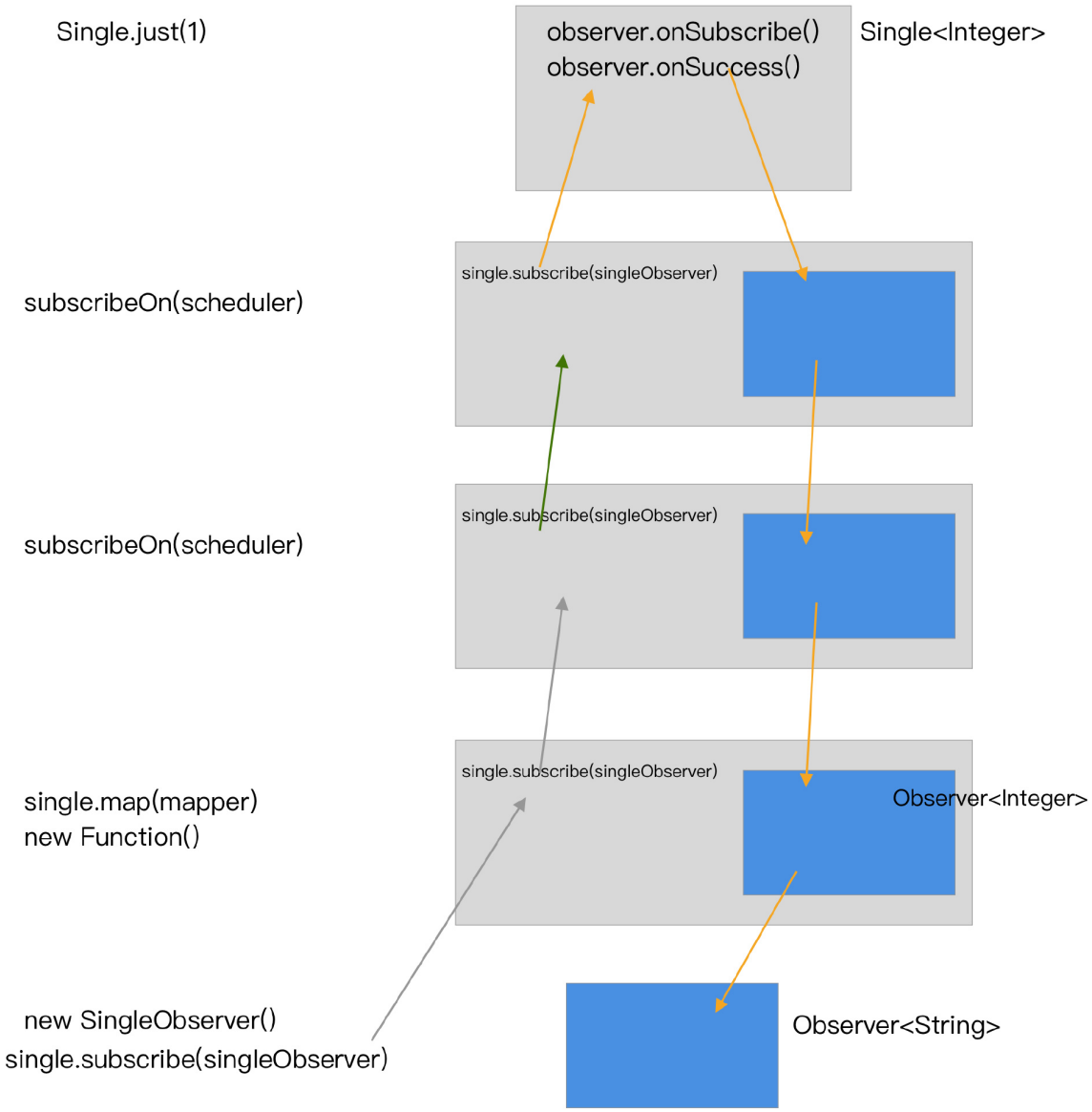
subscribeOn()

原理

在 Scheduler 指定的线程里启动 subscribe()

效果

- 切换起源 Observable 的线程；
- 当多次调用 subscribeOn() 的时候，只有最上面的会对起源 Observable 起作用。



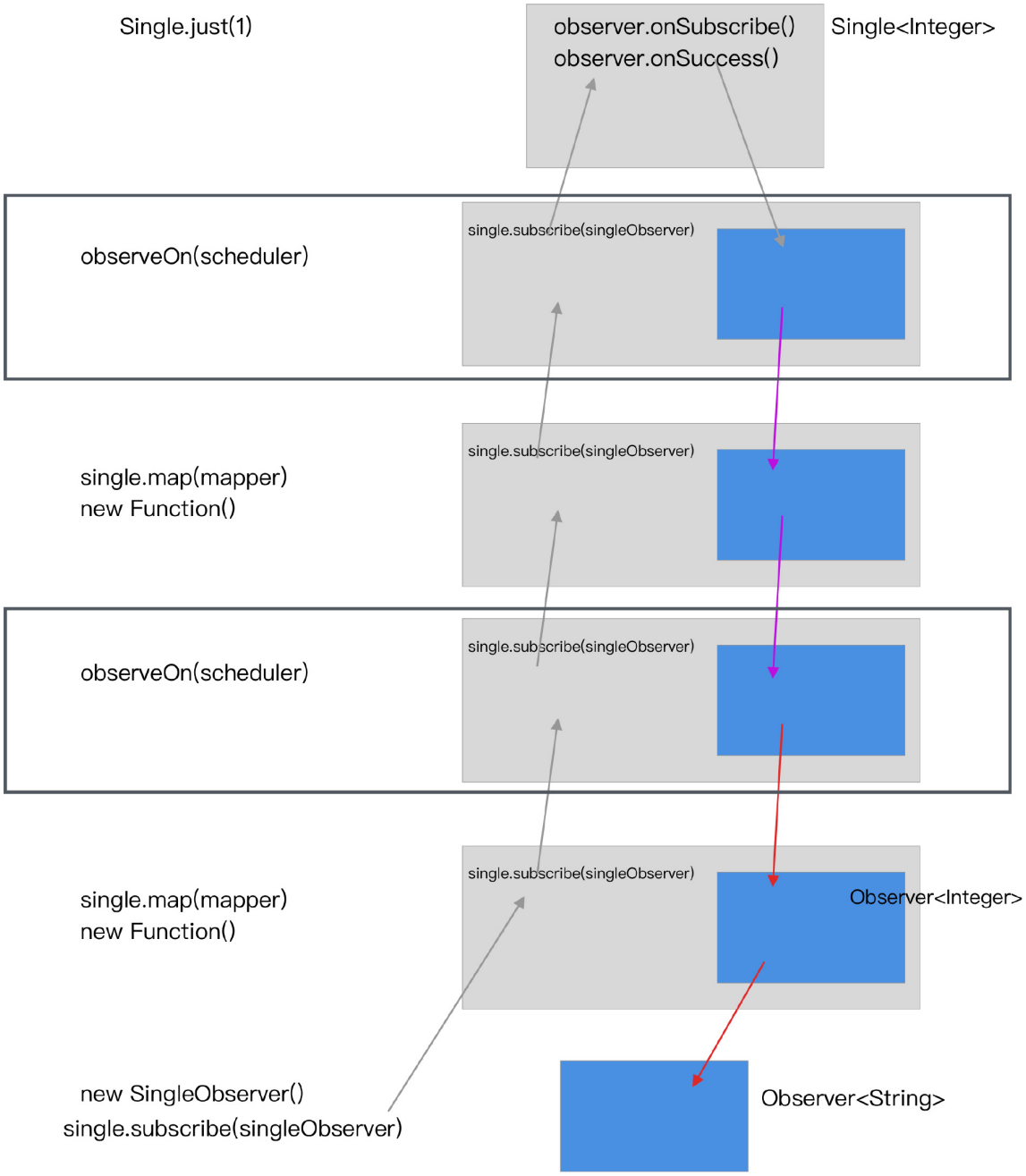
observeOn()

原理

在内部创建的 Observer 的 onNext() onError() onSuccess() 等回调方法里，通过 Scheduler 指定的线程来调用下级 Observer 的对应回调方法

效果

- 切换 observeOn() 下面的 Observer 的回调所在的线程
- 当多次调用 observeOn() 的时候，每个都会进行一次线程切换，影响范围是它下面的每个 Observer (除非又遇到新的 observeOn())



Scheduler 的原理

1. Schedulers.newThread() 和 Schedulers.io()：
    - 当 scheduleDirect() 被调用的时候，会创建一个 Worker，Worker 的内部会有一个 Executor，由 Executor 来完成实际的线程切换；
    - scheduleDirect() 还会创建出一个 Disposable 对象，交给外层的 Observer，让它能执行 dispose() 操作，取消订阅链；
    - newThread() 和 io() 的区别在于，io() 可能会对 Executor 进行重用。
2. AndroidSchedulers.mainThread()：

    通过内部的 Handler 把任务 post 到主线程去做。

问题和建议?

课上技术相关的问题，都可以在学员群里和大家讨论，我一旦有时间也都会来解答。如果我没来就 @我一下吧！

具体技术之外的问题和建议，都可以找丢物线（微信：diuwuxian），丢丢会为你解答技术以外的一切。

更多内容：

- 网站：https://hencoder.com
- 微信公众号：HenCoder

HenCoder

给高级 Android 工程师的进阶手册

微信公众号：HenCoder
微博：扔物线
知乎专栏：HenCoder
稀土掘金：扔物线
http://hencoder.com