RxJava中的错误处理

作者 东东东鲁 (/u/wrzw2v) + 关注
2016.03.31 21:18* 字数 765 阅读 4354 评论 7 喜欢 37
(/u/wrzw2v)

在RxJava中我们可以很方便地处理异常，只要加上 `onError` 即可。

不过，如果异常发生在操作符内部，比如 `flatMap` ，那我们怎么把这个异常传递给 `onError` 呢。

> Checked异常和Unchecked异常
>
> - Checked异常必须被显式地捕获或者传递，而unchecked异常则可以不必捕获或抛出。
> - Checked异常继承java.lang.Exception类。Unchecked异常继承自java.lang.RuntimeException类。

Unchecked异常

一般情况下，unchecked异常会自动传递给 `onError` 。例如以下代码可以打印出“Error!”。

```
Observable.just("Hello!")
          .map(input -> {throw new RuntimeException();})
          .subscribe(
              System.out::println,
              error -> System.out.println("Error!")
          );
```

也有例外的情况，那就是... 那些非常严重的错误，以致于RxJava都不能继续运行了。比如 `StackOverflowError` ，这些异常被认为是致命的，对它们来说，调用 `onError` 毫无意义，并没什么用。你可以用 `Exceptions.throwIfFatal` (http://reactivex.io/RxJava/javadoc/rx/exceptions/Exceptions.html#throwIfFatal(java.lang.Throwable))来过滤掉这些致命的异常并重新抛出，不发射关于它们的通知。

Checked异常

尽管RxJava有自己的异常处理机制，不过Checked异常还是必须由你的代码来处理，也就是说，还是要自己加 `try-catch` 。

假设我们用到这样方法：

```
String transform(String input) throws IOException;
```

我们可以把Checked异常转换为Unchecked异常，像这样：

```
Observable.just("Hello!")
    .map(input -> {
        try {
            return transform(input);
        } catch (Throwable t) {
            throw Exceptions.propagate(t);
        }
    });
```

`Exceptions.propagate()` 只是简单地做了这样一件事：如果异常是Checked异常，那就把它包装成Unchecked异常。

而对于像 `flatMap` 这样返回Observable对象的操作，可以直接返回 `Observable.error()` 。

```
Observable.just("Hello!")
    .flatMap(input -> {
        try {
            return Observable.just(transform(input));
        } catch (Throwable t) {
            return Observable.error(t);
        }
    });
```

异常的屏蔽

很多RxJava初学者都犯了一个错误，过度地使用 `onError` ，其实 `onError` 应该在数据无法继续处理下去时才使用。例如，在使用Retrofit 1 (https://github.com/square/retrofit)的时候，响应的状态码为非200的结果调用 `onError` ，这样，我们在处理非200的响应结果时就会变得十分麻烦。这个问题在Retrofit 2已经解决了，现在可以通过 `Observable<Response<Type>>` 和 `Observable<Result<Type>>` ，来处理 `onNext` 中的非200的结果返回。

也就是说，通常，你可以在发生错误的时候给 `onNext` 一个错误的标识，然后直接在 `onNext` 中处理问题，而不是跳过代码进入 `onError` ，这样还是可以不中断你的数据流，继续运行你的代码。

如何屏蔽异常而不把异常抛给 `onError` ，以下有两种选择：

- `onErrorReturn()` ，在遇到错误时发射一个特定的数据
- `onErrorResumeNext()` ，在遇到错误时发射一个数据序列

```
Observable.just("Request data...")
    .map(this::dangerousOperation)
    .onErrorReturn(error -> "Empty result");
```

当dangerousOperation产生异常时，不会触发 `onError` ，而是返回字符串"Empty result"。

当上游的 `Observable` 观察到异常通知(`onError`)时，通过 `onErrorReturn` 或 `onErrorResumeNext` 来把 `onError` 转换成与下游序列有所区分的数据。

参考

Error handling in RxJava (http://blog.danlew.net/2015/12/08/error-handling-in-rxjava/)
RxDocs (https://mcxiaoke.gitbooks.io/rxdocs/content/topics/Implementing-Your-Own-Operators.html)

赞赏支持

喜欢 37

更多分享

(http://cwb.assets.jianshu.io/notes/images/342614

写下你的评论...

7条评论 只看作者 按喜欢排序 按时间正序 按时间倒序

sonuan (/u/2403dae8b2e4)
2楼 · 2016.04.15 08:56
(/u/2403dae8b2e4)
这只适于对lambda表达式了解的人看

赞 回复

> 东东东鲁 (/u/wrzw2v)： @sonuan (/users/2403dae8b2e4) 可以去了解一下，这个不是重点。
> 2016.04.15 13:06 回复
>
> 添加新评论

4aff9e170692 (/u/4aff9e170692)
3楼 · 2016.04.18 23:13
(/u/4aff9e170692)
为什么IO异常进入了OnError有时候也会造成崩溃。

赞 回复

> 东东东鲁 (/u/wrzw2v)： @4aff9e170692 (/users/4aff9e170692) 有时候？在OnError中会不会又抛出了异常呢
> 2016.04.19 00:49 回复
>
> 4aff9e170692 (/u/4aff9e170692)： @东东东鲁 (/users/wrzw2v) 额，就是普通的rxjava+retrofit的操作，flatmap处理状态码非成功的通过Observable.error给onError，简单的判断状态吗或者异常类型，给提示信息。比如网不好时出现的SocketTimeoutException，这个类型给超时提示，但是有时候也崩溃，这个可能和OnErrorNotImplementedException有关吧。断网出现UnknownHostException直接崩愧，这个就。。。最近有点头大，已经被绕进去了
> 2016.04.19 01:26 回复
>
> 东东东鲁 (/u/wrzw2v)： @4aff9e170692 (/users/4aff9e170692) 实现了onError方法的话一般是不会崩溃的呀
> 2016.04.19 10:12 回复
>
> 添加新评论

小范屯 (/u/ec509f634662)
4楼 · 2017.01.17 19:16
(/u/ec509f634662)
在RxJava2 中Observable.just(null),也会抛异常，这个怎么处理

赞 回复

被以下专题收入，发现更多相似内容