

Programação reativa com Java e Android

Leonardo Lima

- Bacharel Sistemas de Informação (UNIVEM)
- Especialização em desenvolvimento web e aplicativos móveis
- Persys (2,5 anos)
 - Desenvolvedor Android e iOS nativo
 - Desenvolvedor backend (Java, Scala, ?)

Agenda

- Cenário atual
- Programação reativa
- Reactive Extensions
- RxJava
- RxJava no Android
- Nosso projeto

Cenário atual

Cenário atual

Usuários querem dados em tempo real. Tweets agora. Pedidos agora. Preços agora.

Como desenvolvedor você quer disparar mensagens e esquecê-las. Você não quer ser bloqueado esperando um resultado. Você quer os resultados entregues a você quando estiverem prontos. Melhor ainda, quando trabalhando com conjuntos de dados, você quer recebê-los individualmente assim que prontos. Você não quer esperar o lote inteiro ser processado para ver o primeiro item.

Programação reativa

Programação reativa: programação com streams de dados assíncronos.

Não é um conceito novo. *Event buses* ou até cliques de mouse são na verdade ***streams* assíncronos**, que você pode **observar** e **realizar operações**. Reactive Extensions é essa ideia com esteroides.

Você consegue criar *streams* de tudo, não somente de eventos de clique ou *hover*. *Streams* são baratas e ubíquas. **Tudo pode ser um *stream***: variáveis, entrada de usuários, propriedades, caches, estruturas de dados, etc.

Imagine que seu feed do Facebook é um *stream* da mesma forma que cliques são. Você pode ouvir esse stream e reagir de acordo.

Programação imperativa

int a = b + c;

Significa que está sendo atribuído à variável **a** o valor de **b + c** no momento em que a expressão é avaliada. Posteriormente o valor de **b** e **c** podem ser alterados sem efeitos em **a**.

Programação reativa

=B1+C1

Softwares de planilhas podem armazenar em suas células valores literais ou fórmulas que são calculada com base no valor de outras células. Quando o valor de outras células muda, o valor da célula que contém a fórmula também muda.

Reactive Extensions

Reactive Extensions (ReactiveX, Rx)

ReactiveX é uma biblioteca para compor programas **assíncronos** e **baseados em eventos** usando **sequências observáveis**.

Estende o padrão **Observer** para dar suporte à sequências de dados e/ou eventos e adiciona **operadores** que permitem unir diferentes sequências de forma declarativa, abstraindo preocupações com fatores como threads em baixo nível, sincronização, *thread-safety*, estruturas de dados concorrentes e I/O não bloqueante.

Rx

	um item	vários itens
síncrono	<code>T getData()</code>	<code>Iterable<T> getData()</code>
assíncrono	<code>Future<T> getData()</code>	<code>Observable<T> getData()</code>

Rx

O tipo Observable adiciona duas semânticas ao padrão Observer do Gang of Four para combinar com o tipo Iterable:

- A possibilidade do produtor sinalizar ao consumidor que não existem mais dados: `onCompleted()`
- A possibilidade do produtos sinalizar ao consumidor que ocorreu um erro: `onError()`

Rx

	Iterable (pull)	Observable (push)
pegar dado	T next()	onNext(T)
sinalizar erro	throws Exception()	onError(Throwable)
completar	!hasNext()	onComplete()

Tipos chave

- Observable
- Observer
- Subscription

Observable / Observer

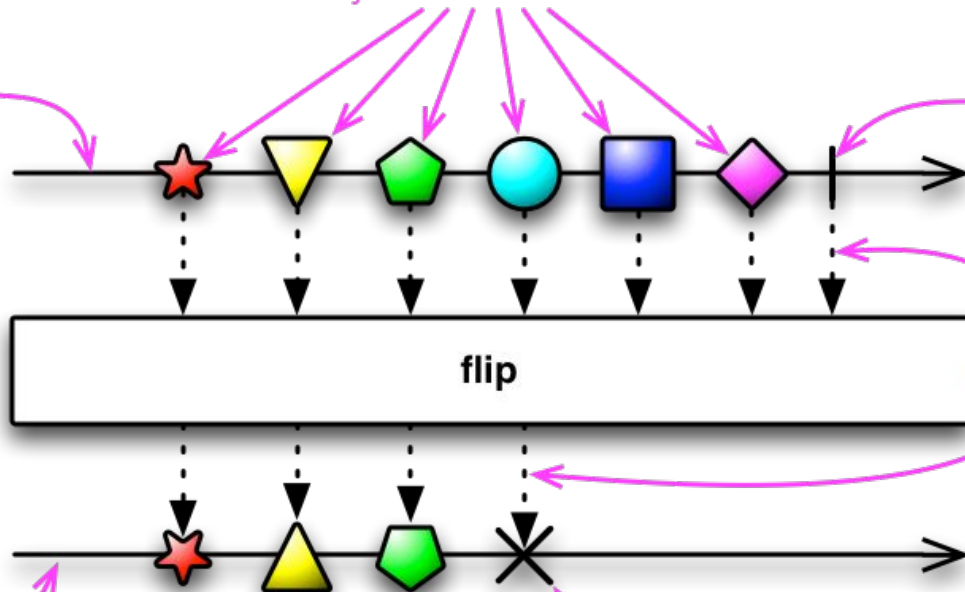
No Rx, um **Observer** se inscreve em um **Observable**. O Observer então reage aos itens que o Observable **emite**. Esse padrão facilita a criação de operações concorrentes porque ele **não precisa bloquear** enquanto espera que o Observable emita objetos.

Marble diagrams

This is the timeline of the Observable. Time flows from left to right.

These are items emitted by the Observable.

This vertical line indicates that the Observable has completed successfully.



These dotted lines and this box indicate that a transformation is being applied to the Observable. The text inside the box shows the nature of the transformation.

This Observable is the result of the transformation.

If for some reason the Observable terminates abnormally, with an error, the vertical line is replaced by an X.

RxJava: Criando observables

```
Observable<Integer> observable = Observable.just(0, 1, 2, 3, 4, 5, 6, 7, 8, 9);
observable.subscribe(new Action1<Integer>() {
    @Override
    public void call(Integer x) {
        System.out.println(x);
    }
});
```

RxJava: Criando observables

`create()`

`from()`

`just()`

`range()`

`fromCallable()`

...

Transformando observables

```
Observable.just(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
    .map(new Func1<Integer, Integer>() {
        @Override
        public Integer call(Integer integer) {
            return integer * 2;
        }
    })
    .subscribe(new Action1<Integer>() {
        @Override
        public void call(Integer x) {
            System.out.println(x);
        }
    });
```

Transformando Observables

`map()`

`flatMap()`

`groupBy()`

`buffer()`

...

Filtrando observables

```
Observable.just(0, 0, 1, 1, 2, 2, 3, 3, 4)
    .distinct()
    .subscribe(new Action1<Integer>() {
        @Override
        public void call(Integer x) {
            System.out.println(x);
        }
    });
```

Observer

Subscription subscribe()

Subscription subscribe(Action1<? super T> onNext)

Subscription subscribe(Action1<? super T> onNext, Action1<java.lang.Throwable>
onError)

Subscription subscribe(Action1<? super T> onNext, Action1<java.lang.Throwable>
onError, Action0 onComplete)

Subscription subscribe(Observer<? super T> observer)

Subscription subscribe(Subscriber<? super T> subscriber)

Subscription

Relação entre o **Observable** e o **Observer**. Quando o método `subscribe()` é chamado no `Observable`, um objeto **Subscription** é gerado.

RxJava no Android

RxJava no Android

- Tudo pode virar um Observable:
 - Cliques
 - Texto
 - Sensores
 - ...
- Trocar de threads
 - Acesso ao banco
 - Acesso à rede

Nosso projeto

Primeira tarefa: tirar trabalhos pesados da *main thread*

subscribeOn()

observeOn()

RxAndroid

Segunda tarefa: observar eventos de uma caixa de texto

Observable.OnSubscribe

Subscriptions





Terceira tarefa: combinando

`combineLatest()`

`concatMap()`

Referências

<http://reactivex.io/>

<http://techblog.netflix.com/2013/02/rxjava-netflix-api.html>

<https://github.com/Froussios/Intro-To-RxJava/>

<https://medium.com/swlh/party-tricks-with-rxjava-rxandroid-retrolambda-1b06ed7cd29c>