UNIVERSIDADE FEDERAL DE ALAGOAS – CAMPUS ARAPIRACA

CIÊNCIA DA COMPUTAÇÃO – PARADIGMAS DE LINGUAGENS DE PROGRAMAÇÃO

PROF. THIAGO SALES

PROGRAMAÇÃO REATIVA

Solução de software utilizando o RxJS

SAVYO NASCIMENTO DE FARIAS - 17112844

PROGRAMAÇÃO REATIVA

1. O que é programação reativa?;
2. Os pilares da programação reativa;
3. O Framework ReactiveX;
4. RxJS;
5. O uso de operadores;
6. Operadores utilizados;
7. Projeto: Petiscaria;
8. Casos para testes;
9. Referências

1. O que é programação reativa?;

- Programação reativa é um modelo ou um paradigma de programação que é orientado a fluxo de dados e propagações de estados.
- Estes fluxos de dados são em grande parte assíncronos, ou seja, as operações são independentes umas das outras e não precisam ser executadas em uma sequência específica.
- Todas as ações quando falamos sobre programação reativa são transmitidas e detectadas por um fluxo de dados, logo, as aplicações reativas, são constituídas por reações a alterações nestes fluxos de dados.

Fonte: https://bit.ly/2SGbMEJ

2. Os pilares da programação reativa;

- A programação reativa é de grande importância devido o crescimento da internet e a demanda de aplicações e operações em tempo real.
- Ela está baseada em 4 pilares sendo eles:

Responsivo **Resiliente** **Elástico** **Orientado a mensagens**

Fonte: https://bit.ly/2SGbMEJ

3. O Framework ReactiveX;

- ReactiveX é uma biblioteca para composição assíncrona e programação baseada em eventos utilizando sequências observáveis.

Fonte: https://bit.ly/2SGbMEJ

4. RxJS;

- O RxJS é uma biblioteca para composição de programas em JavaScript que são assíncronos e baseados em eventos, usando sequências observáveis.

Fonte: https://bit.ly/3w3FRfl

4.1 RxJS;

Observer Pattern - um modelo de assinatura no qual os objetos se inscrevem em um evento e são notificados quando o evento ocorre;

Iterator Pattern - permite que os clientes realizem loop de forma efetiva em uma coleção de objetos.

Fonte: https://bit.ly/3w3FRfI

4.2 RxJS;

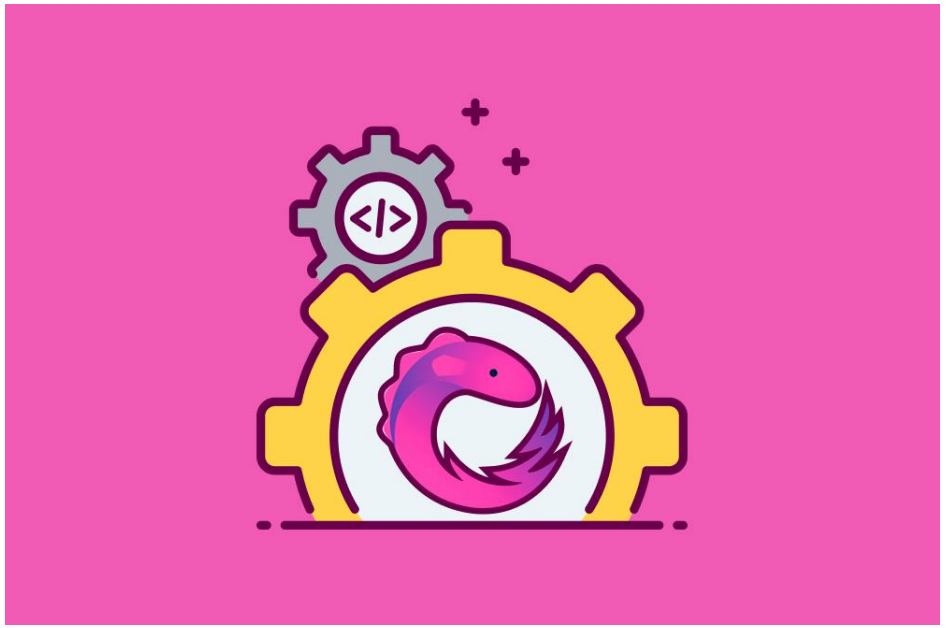
- Os conceitos presentes na biblioteca são essenciais para a resolução de eventos assíncronos, sendo eles:

Observable **Observer** **Subscription** **Operators** **Subject** **Schedulers**

Fonte: https://bit.ly/3w3FRfI

5. O uso de operadores;

- Os operadores são fundamentais dentro do nosso framework, eles são funções puras que permitem um estilo de programação funcional de lidar com coleções com operações como map, filter, concat, reduce, etc.



Fonte: https://bit.ly/3bln7At

6. Operadores utilizados;

- No presente trabalho, para o desenvolvimento do projeto foram utilizado os seguintes operadores:
  - merge: combina vários observáveis em um único;
  - skipWhile: discarta items dos observáveis seguindo uma condição especifica;
  - distinct: suprime itens duplicados de um observável;
  - take: apresenta apenas uma quantidade N de itens de um observável;
  - filter: apresenta apenas os itens de um observável que passam em um teste especifico.

7. Projeto: Petiscaria;

- Para o presente projeto, será simulada uma petiscaria onde será disposto um cardápio de itens e iremos trabalhar com estes itens (fluxo de dados), utilizando 3 clientes com demandas diferentes.

Fonte: https://bit.ly/3uG1Lpa

8. Casos para testes;

- Para testarmos os nossos operadores e o projeto iremos utilizar 3 testes, representados por 3 pessoas (clientes), cada um deles com uma especificidade, sendo elas:

Cliente 1: Este cliente só irá consumir do cardápio, 8 petiscos salgados, eles não podem ser o prato de calabresa, 3 sobremesas e nenhum petisco vegetariano. Ele não repetira nenhum prato.

Cliente 2: Este cliente não irá consumir petiscos salgados e nem vegetarianos, apenas sobremesas.

Cliente 3: Este cliente irá consumir um pouco de cada petiscos salgados, vegetarianos e sobremesas, mas não poderá repetir nenhum deles.

8. Referências

- O que é programação reativa? -https://bit.ly/3f5BgCU;
- Rxjs programação reativa em javascript. - https://bit.ly/3oaCjFP;
- Introdução ao ReactiveX - https://bit.ly/2Qr98C8.