# Реактивное программирование



## Реактивное программирование

Reactive Programming (от слова "реакция") - программирование с асинхронными потоками данных

Получаете данные, вызывается метод - все это "потоки"

Поток - это контейнер для реализации принципов РП

Асинхронные вызовы (не зависает UI) - как только получаем результат - обновляем UI

Принципы "издатель-подписчик" - нет жесткой связи между объектами, каждый сам выбирает какие данные "слушать" (пример с Excel - в одном месте меняете значение, в других местах обновляется)

Принципы потоков ("все - потоки") - можно комбинировать, вкладывать друг в друга и пр. - собирать итоговый вариант как конструктор

Поддерживается во многих современных языках

Пока не начнете работать - до конца не поймете что такое РП

### **RxJS**

https://rxjs.dev

Реализация РП в JavaScript (TypeScript)

Angular изначально создавался под РП - частичное обновление страниц, быстрый отклик UI, автоматические Ајах запросы

Вручную не нужно выполнять Ajax запросы - все за вас делают Angular + RxJS

#### Объекты в RxJS

Готовые решения (контейнеры) для асинхронных вызовов с различным поведением

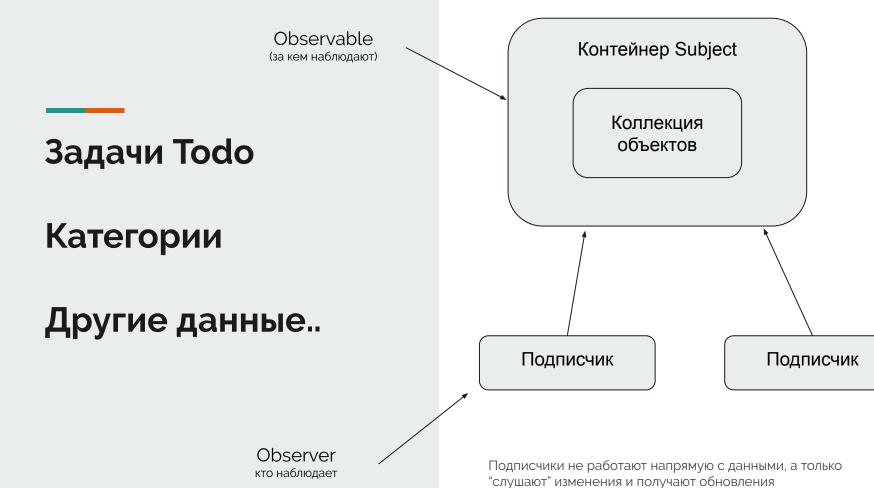
Нужно "обернуть" данные в контейнер и любой может подписаться на их изменение

**Observable** - объект, за которым наблюдем (базовый класс)

Subject - можно слушать изменения и реагировать на них (сам может подписывать на другие Observable)

**BehaviorSubject** - хранит последнее значение

**ReplaySubject** - хранит N последних значений



#### С чего начнем

https://rxjs.dev

Везде, где получаем данные - будем использовать асинхронные вызовы (издатель-подписчик)

### Домашнее задание

- 1. Почитать (здесь кроется решения практических заданий 3 и 4)
  - https://medium.com/@luukgruijs/understanding-rxjs-behaviorsubject-replaysubject-and-asyncsubject-8cc061f1cfc0
- 2. Начать изучать <u>https://rxjs.dev/guide/overview</u>
- 3. Реализовать отображение всех задач при запуске проекта
- 4. Реализовать отображение категорий в реактивном стиле (по аналогии с задачами)
  - а. Добавить **Subject** и подписчика