academeez

Observables

Observables and their laziness

Observables

- Observables will wrap our async or sync code that will shout out to the listeners
- Observables will push the data to the listeners
- You will interact or get an Observable in one of the following ways
  - Create the Observable yourself (less common)
  - Create the Observables using RXJS creation functions - creation operators (a bit more common)
  - Use a method or library that returns an Observable

Create Observable Yourself

academeez

- You can create an Observable by creating the instance of the class
- The constructor will get a method that will push data to the listeners by calling next
- This Observable will send a message every second

```
import { Observable } from 'rxjs';

const intervalObservable: Observable<string> = new Observable((observer) => {
  setInterval(() => {
    observer.next('Hello from Observable');
  }, 1000)
});
```

Create Observable Operators

- Operators are functions
- There are operators that will help you to create an Observable
- This Observable will emit an incrementing number of second

```
import { Observable, interval } from 'rxjs';

const intervalObservable: Observable<number> = interval(1000);
```

Somebody created Observable for you

academeez

- The most common use case is going to be you using a library or method that will return an Observable
- The following is a common code in Angular
- We are using an instance with a method that returns an Observable
- That method sends an Ajax get request

```
// http: HttpClient
    const ajaxObservable: Observable<any> = http.get('https://some-url');
```

Promise VS Observable EX

- Create a Promise that wraps a setTimeout and will resolve with an hello message after one second
- Create an Observable that wraps a setInterval and will next an hello message after one second
- Try to answer the following questions:
    - Does the Promise wrapped async function runs sync or async?
    - What about the Observable async function? Does it run sync? async?
    - Does the running of the Observable async method depends on something?
    - How many times will the Promise async method will run if we have many listeners?
    - What about the Observable async method - how many times on multiple listeners?

Observable is lazy

academeez

- Observable async method will not run if there are no listeners
- Unlike Promise which runs the async method right away synchronously
- When you attach a listener it will call the async method right away (sync)

Observable async method called many times

academeez

- The Observable async method will be called for every listener
- Unlike Promise where the async method will be called once regardless of the amount of listeners

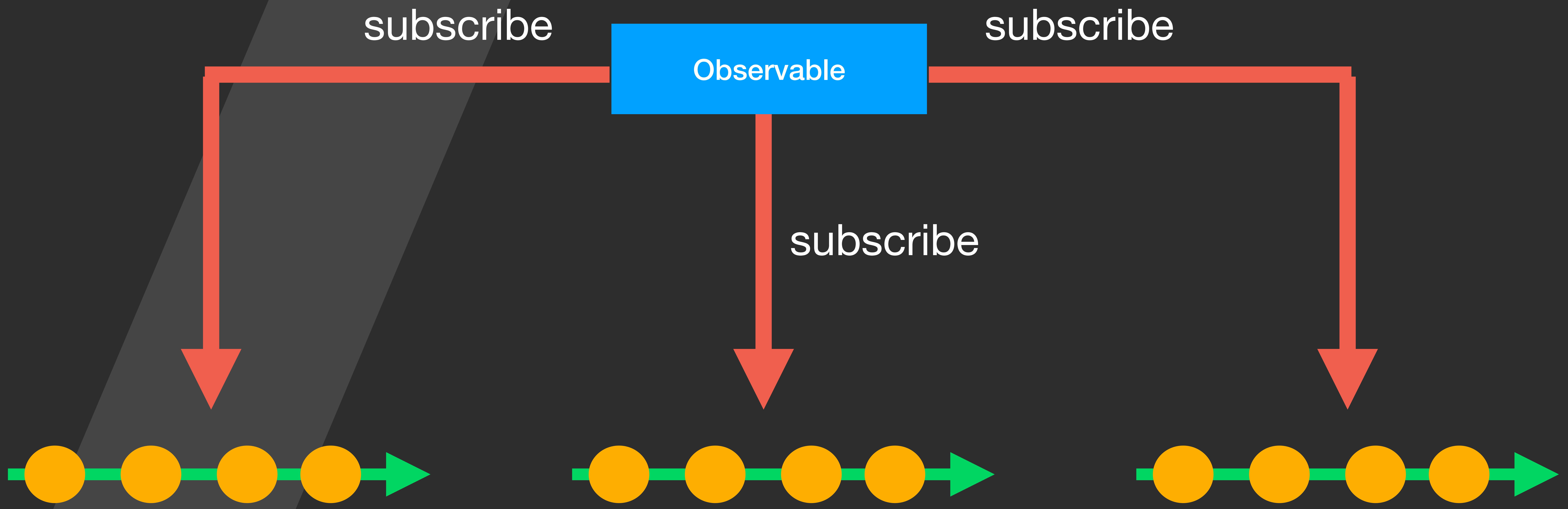
Observable data stream duplicates!

academeez

- You can think of the Observable like a function
- subscribe will call that function
- The function will return a data stream for every listener that called the function
- For 100 subscribe calls we will have 100 data streams
- In our example calling 100 subscribe will create 100 setIntervals

Observable data stream duplicates!

academeez



Summary

academeez

- We learned that the Observable async method will behave lazy and will not be called unless we attach a listener
- We learned that the number of times the async method is called will be equal to the number of listeners
- We learned that Observable duplicated the data stream for every listener
- When we attach a listeners the async method is called sync

academeez

Thank You

Next Lesson: Closing the Observable