NG MEETUP

TASTE OF RXJS

MD. ZIAUL HAQ

JavaScript Developer
@Upwork Global

Author @Packt

Follow me:
@jquerygeek

WHAT IS RXJS?

RXJS

- Reactive Extensions for JavaScript, A Library
- Combination of Observer pattern and Iterator pattern with functional approach
- Handle asynchronous events as collection

RXJS

- RxJS is a library for composing asynchronous and event-based programs by using observable sequences

RXJS

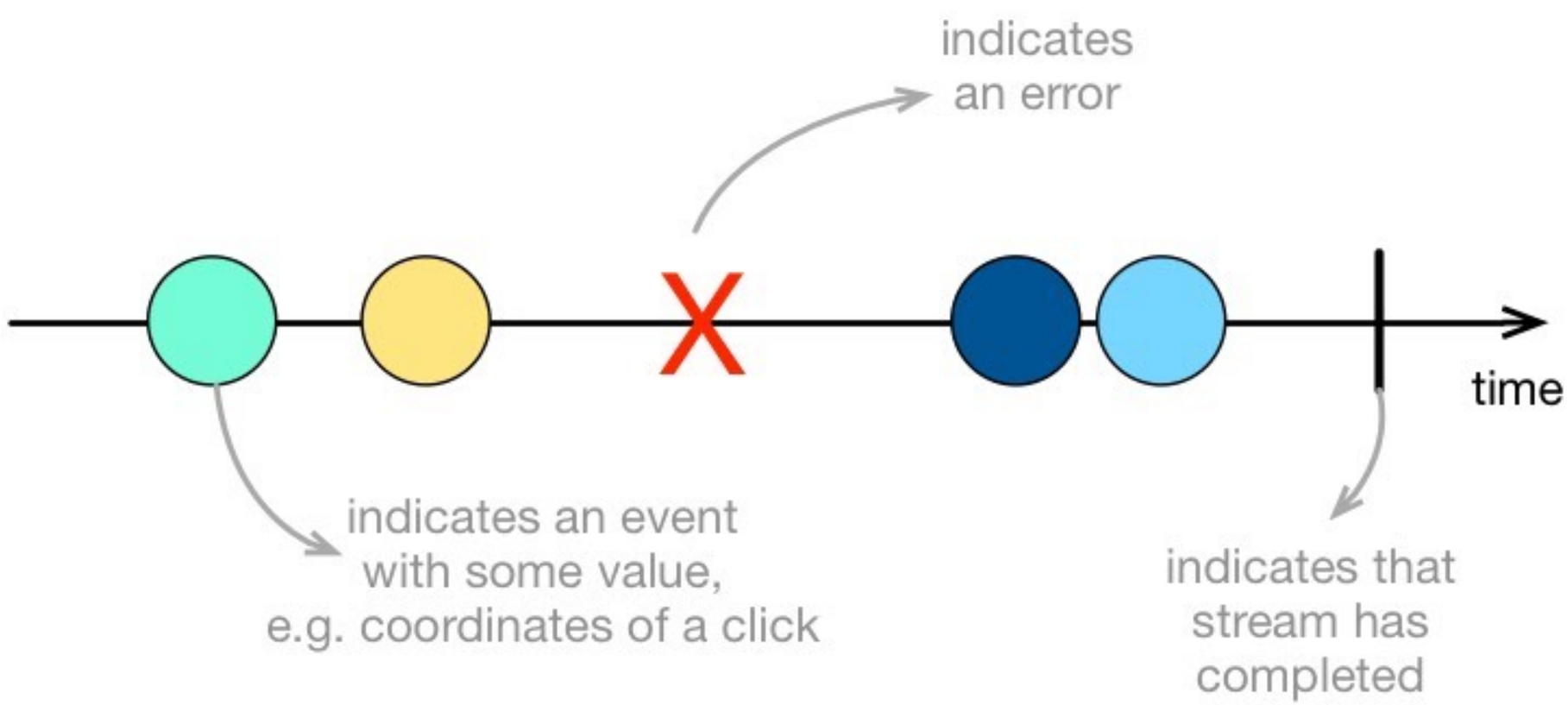
- A library for composing asynchronous program by using observable sequences
- Also provide a long list of operator for select, filter,, transform, combine, etc

THINK OF RXJS AS LODASH FOR EVENTS

reactivex.io

STREAM

- A sequence of ongoing events order by time
- We can get **value**, error and complete signal from stream



SOURCE OF DATA STREAM

- UI Events
- Http request
- Array like object
- Memory / cache

RXJS IS NOT ALONE

- **RxJS**
- RxJava
- RxPHP
- Rx.Net
- RxScala
- RxSwift

WHAT RXJS CONTAINS

- Observable
- Observer
- Operators

OBSERVABLE

- Observable are used to watch these stream and emit functions when a value, error or complete signal return
- Observable can be subscribed by an observer or many observer
- Observable will constantly watch stream and will update accordingly
- We can interact with as regular array and can apply all the operator on these data.

OBSERVABLE IS JUST A FUNCTION THAT TAKES AN OBSERVER AND RETURNS A FUNCTION

Ben Lesh

```
const node = document.querySelector('input[type=text]');

const input$ = Rx.Observable.fromEvent(node, 'input');

input$.subscribe({
  next: (event) => console.log(`You just typed ${event.target.value}!`)
  error: (err) => console.log(`Oops... ${err}`),
  complete: () => console.log(`Complete!`)
});
```

COLD VS HOT OBSERVABLE

- Cold: When subscribe
- Hot: From begging

OBSERVER

- Observer watch the Observable
- Use .subscribe() for invoke the Observable

OPERATORS

- Creating operators: fromEvent
- Transformation operators: map, scan
- Filtering operators: filter, first
- Combination operators: concat
- Utility operators: toPromise

OPERATORS

```
const input$ = Rx.Observable.fromEvent(node, 'input')
  .map(event => event.target.value)
  .filter(value => value.length >= 2)
  .subscribe(value => {
    // use the `value`
  });
```

WHY RXJS

WHO USING

- Angular return Observable on asynchronous actions
- Goes with every framework

OBSERVABLE

VS

PROMISE

OBSERVABLE VS PROMISE

- All that Promise provide
- Observable can handle multiple events
- Cancelable
- Can retry with retry operator

RESOURCE

- http://reactivex.io/rxjs/
- https://medium.com/@benlesh/learning-observable-by-building-observable-d5da57405d87

QUESTIONS?