# Reactive JS

# Reactive Programming

- Reactive programming is, effectively, programming with asynchronous data streams

- The asynchronous part is not new but the streams part implies that the data may be processed like an array

- A stream is a sequence of data ordered in time; data, events, errors, and termination signals may be emitted

- The stream may be transformed using operators

- A stream is AKA an Observable; the listeners are subscribers

# Getting Started with RxJS

- RxJS is one of many reactive JS libraries
- To install: `npm install rxjs`
- Then: `import {Observable} from 'rxjs';`
- Or CDN: https://cdnjs.com/libraries/rxjs
- Then: `const {Observable} = rxjs;`

# Observable

- An Observable pushes data to its subscribers

```
const ob$ = Observable.create(sub => {
  sub.next('Hello');
  sub.next('How are you?');
  sub.complete();
});
```

- There are several convenience functions that simplify the creation of the Observable, e.g. of, fromEvent, etc

# Subscriber

- A Subscriber subscribes to (listens for data from) an Observable

- A subscriber may be composed of three handler functions: one for data, one for errors, and one for completion messages

```
ob$.subscribe(
  data => console.log(data),
  error => console.err(error),
  () => console.log('Complete'));
```

# Sequencing

- Observables may be transformed in much the same way you would transform an array

- There exists a wide variety of pipeable operators that fall into various categories incl.: transformational, filters, combinational, conditional etc.

```
of(1, 2, 3, 4, 5)
  .pipe(
    filter(num => num % 2 == 0),
    map(num => num * num))
  .subscribe(console.log);
```

# Summary

- Reactive programming is programming with asynchronous data streams

- RxJS is probably the most popular reactive JS lib

- An Observable pushes data to its subscribers

- A Subscriber reacts to the data it receives from the Observable to which it's subscribed

- Pipeable operators may be used to transform Observables (the real power of reactive programming)