FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

Desarrollo de Aplicaciones Multiplataforma

Brian Ducca

Índice

- Services con HTTP
- Observables
- Promesas
- Async-Await

Observables

- Nos permite pasar mensajes entre los denominados “publishers” y “subscribers”. Funciona de manera declarativa
- Permite manejar 0,1 o más eventos
- Puede ser cancelado

```
export class StopwatchComponent {

 stopwatchValue: number;
 stopwatchValue$: Observable<number>;

 start() {
   this.stopwatchValue$.subscribe(num =>
     this.stopwatchValue = num
   );
 }
}
```

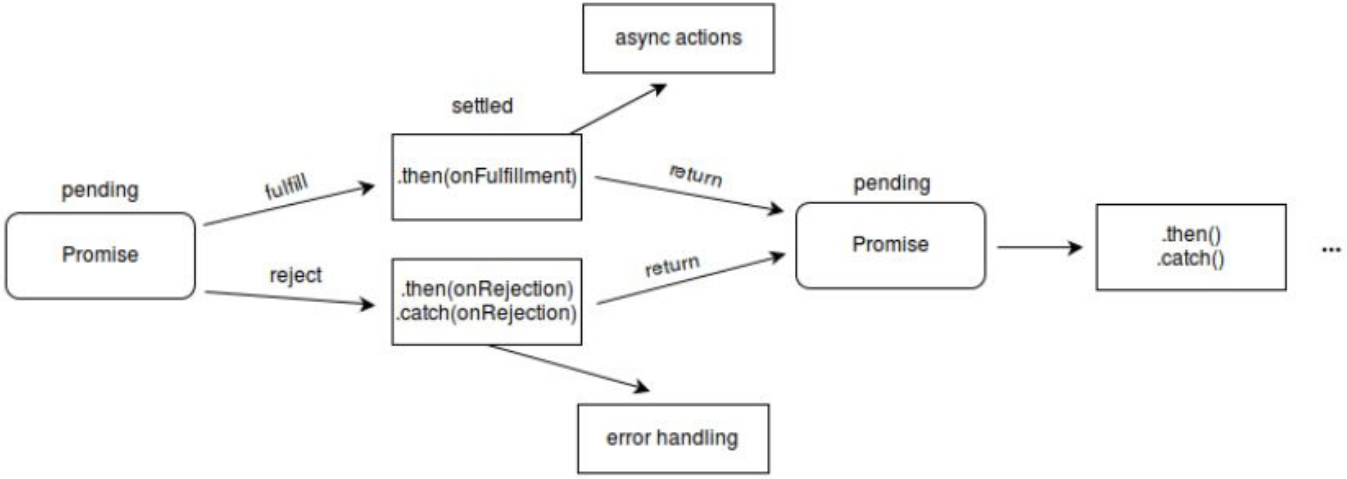
Promesas

- Maneja un solo evento cuando una operación asíncrona completa o falla
- Una promesa puede ser cumplida con un valor , o rechazada con una razón (error)
- No se puede cancelar

```
let error = true;
function doAsyncTask() {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      if (error) {
        reject('error'); // pass values
      } else {
        resolve('done'); // pass values
      }
    }, 1000);
  });
}

doAsyncTask().then(
  (val) => console.log(val),
  (err) => console.error(err)
);
```

Estados de una promesa



Async - Await

- Otra forma de presentar datos recibidos desde una API
- Una función “async” puede contener una expresión “await”.
- El “await” pausa la ejecución de la función async , espera a la resolución de la promesa , luego continua con la ejecución de la función async y retorna el valor.

Ejemplo Promesa vs Async-Await

```
resolveAfter2Seconds(x) {
    return new Promise(resolve => {
      setTimeout(() => {
        resolve(x);
      }, 2000);
    });
  }
 getValueWithPromise() {
    this.resolveAfter2Seconds(20).then(value => {
      console.log(`promise result: ${value}`);
    });
    console.log('I will not wait until promise is resolved');
 }
```

Async-Await

```
async getValueWithAsync() {
    const value = <number>await this.resolveAfter2Seconds(20);
    console.log(`async result: ${value}`);
}
```