



# Promise & Observable

Web programiranje

Jurica Maltar

# Promise



- *Asinkroni* tip podataka
- **Promise** (JavaScript) ili **Promise**<T> (TypeScript)
  - prihvaća callback s druga dva callback-a
  - **resolve**, ako sve prođe kako treba, a **reject** u suprotnome
- Nakon instanciranja

```
let promise = new Promise((resolve, reject) => { /* ... */ });
```

- slijedi `promise.then((value) => {  
 /*...*/  
}, (error) => {  
 /*...*/  
});` ili `promise.then((value) => {  
 /*...*/  
}).catch((error) => {  
 /*...*/  
});`

# Promise



- `Promise.all`
- `Promise.race`
- Umjesto `then/catch` u kombinaciji s callback-ovima, možemo koristiti:
  - `async` funkciju unutar koje *resolve-amo* promise pomoću ključne riječi `await`
  - Često u kombinaciji s `try/catch` blokom

# Primjeri loše i dobre prakse za UserDB iz copy.txt



```
const ms = 500;
const threshold = 0.1;

class UserDB {
  static is_from_EU(id, s_callb, e_callb) {
    setTimeout(() => {
      let prob = Math.random();
      if (prob >= threshold)
        s_callb(true);
      else e_callb("Something went wrong (EU)");
    }, ms);
  }

  /*
   is_legal_age, get_username
  */
}
```

# Promise u praksi



- U komunikaciji s bazom podataka

```
await collection.find({ tags: ["blank", "red"] }).toArray();
```

- Pri učitavanju datoteka na poslužitelju

```
await fs.readFile("./data.json");
```

- Za kriptografiju

```
await pbkdf2Async(password, salt, 1000, 64, "sha512");
```

# RxJS



- JavaScript biblioteka koja slijedi paradigmu reaktivnog programiranja
- ReactiveX: JavaScript, Java, Python, C++, ...
- Reaktivno programiranje
  - Tok podataka (data stream)
- Angular koristi RxJS za:
  - **EventEmitter**, **HttpClient**, **FormGroup**, signali

# Observable

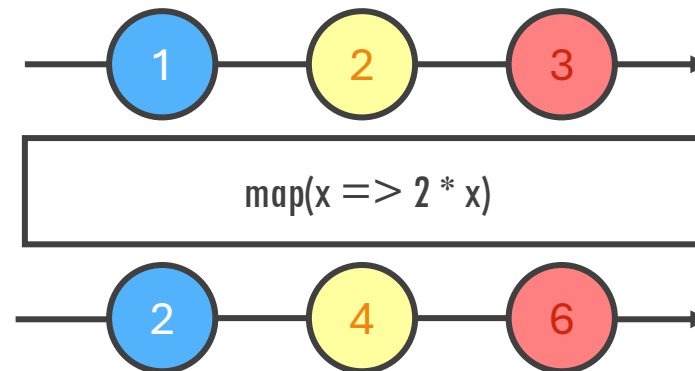


- Klasa koja predstavlja tok podataka
- Temeljena na obrascima oblikovanja:
  - Observer
  - Iterator
- `Observable<T>`
- ```
interface Observer<T> {  
    next: (value: T) => void;  
    error: (err: any) => void;  
    complete: () => void;  
}
```
- Drugi načini stvaranja: `from`, `bindCallback`, `fromEvent...`



# Lanac observable-a

- Više je tokova moguće ulančavati
- Funkcija **pipe**
- Operatori: **map**, **filter**, **reduce**...
- Marble diagram:





# Observable vs Promise



|      | Single   | Multiple   |
|------|----------|------------|
| Pull | Function | Iterator   |
| Push | Promise  | Observable |

- Observable → Promise:
  - Deprecated: `observable.toPromise()`;
  - `lastValueFrom`, `firstValueFrom`