Fetch

A lot of promises

christoffer.wallenberg@zocom.se

ZoCom

Tänk att du är ett barn och du vill ha ett riktigt coolt lego. Du vet inte om du kommer få det eller när.

Din mamma kan antingen

Köpa det riktigt coola legot till dig då hon är glad

eller

inte köpa det riktigt coola legot då hon inte är glad

Detta är ett promise!

Promise

- Ett promise har tre tillstånd:
- **Pending** - Du vet inte om du får något lego
- **Fulfilled** - Din mamma köpte lego till dig
- **Rejected** - Din mamma köpte inte lego till dig

Promise

```
let askMom = new Promise(function(resolve, reject) {
    if(momIsHappy) {
        resolve('Lego köpt!');
    } else {
        reject('No lego for you!');
    }
  });

askMom.then(function(value) {
    console.log('Resolve:', value);
}).catch(function(value) {
    console.log('Reject:', value);
});
```

Konstruktor med en funktion som tar två funktioner

Mamma köper lego

Mamma köper inte lego

Om resolve gå in här

Om reject gå in här

AJAX-anrop

```
let ajax = new XMLHttpRequest();
ajax.open('get', url);
ajax.onreadystatechange = function() {
    if(ajax.status == 200 && ajax.readyState == 4) {
        console.log('Result: ' + ajax.responseText);
    }
}
ajax.send();
```

Fetch

```
fetch(url)     // Funktionen fetch körs direkt
.then(function(response) {
    // Den här koden körs när servern svarar
    return response.json();  // ett promise, gör om
svaret till ett objekt
})
.then(function(data) {  // objekt
    // Den här koden körs när svaret gjorts om till
ett objekt
    console.log(data)
});
// Koden här kommer att köras direkt efter fetch
och innan servern hinner svara
```

Fetch

```
fetch(url)     // Funktionen fetch körs direkt
.then(function(response) {
    // Den här koden körs när servern svarar
    return response.json();  // ett promise, gör om
svaret till ett objekt
})
.then(function(data) {  // objekt
    // Den här koden körs när svaret gjorts om till
ett objekt
    console.log(data)
}).catch(function(error) {
    console.error(error); // Denna kod körs om det
blir fel exempelvis att servern inte svarar
});
```

Synkront vs asynkront

- Vid synkront så görs saker i tur och ordning efter varandra.
- Exempel: en funktion som returnerar ett värde

- Vid asynkront så görs vissa saker parallellt.
- Exempel: Påbörjar ett Ajax-anrop och sedan fortsätter med koden och sedan återvänder till anropet när man fått ett svar.

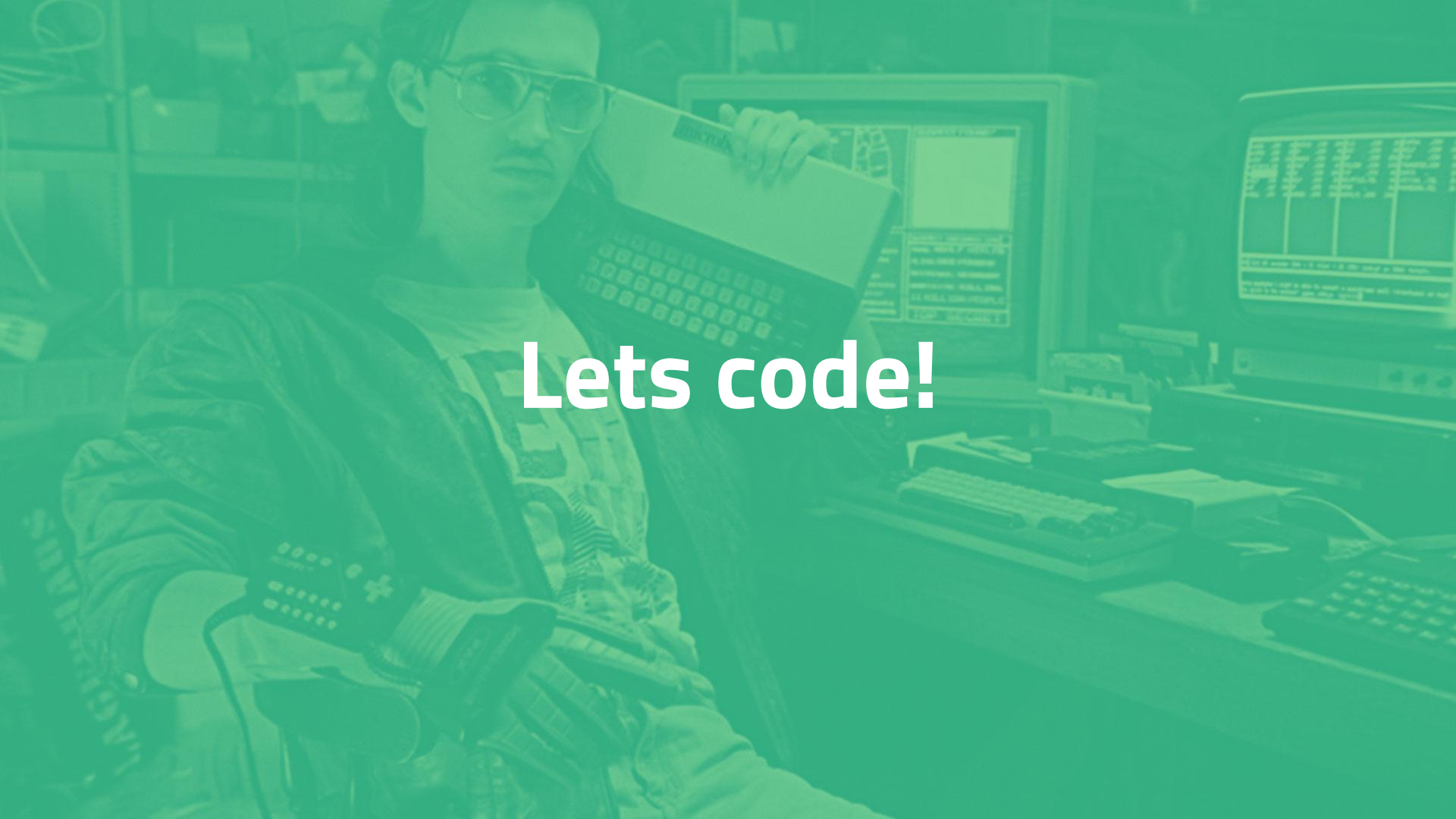
Mer läsning

Promises

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise

Fetch

https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch


Lets code!