
Globant
we are ready

UI Engineering Studio. Day 6

Bootcamp: JS Asynchronous

Globant

Asynchronous

not happening or done at the same time or speed

-Cambridge

UI Boot Camp: JS

JS Asynchronous Code

JavaScript is a blocking, single-threaded language

- Eventloop
- Next tick
- Callbacks: just the name of a convention for using JavaScript functions. There isn't a special thing called a 'callback' in the JavaScript language, it's just a convention

-MDN

Function declaration

```
function sumar(a, b) {
 var c = a + b;
 setTimeout(()=> {c = c*2})
 return c;
}
```

Function expression

```
var request = require('request');
request('http://sohamkamani.com', function (error, response, body)
{
  console.log(body);
})
console.log('I come after the request');
```

UI Boot Camp: JS

Callback and Callback Hell

Asynchronous JavaScript, or JavaScript that uses callbacks, is hard to get right intuitively. A lot of code ends up looking like this:

- source

```
fs.readdir(source, function (err, files) {
  if (err) {
    console.log('Error finding files: ' + err)
  } else {
    files.forEach(function (filename, fileIndex) {
      console.log(filename)
      gm(source + filename).size(function (err, values) {
        if (err) {
          console.log('Error identifying file size: ' + err)
        } else {
          console.log(filename + ' : ' + values)
          aspect = (values.width / values.height)
          widths.forEach(function (width, widthIndex) {
            height = Math.round(width / aspect)
            console.log('resizing ' + filename + 'to ' + height + 'x' + height)
            this.resize(width, height).write(dest + 'w' + width + '_' + filename, function(err) {
              if (err) console.log('Error writing file: ' + err)
            })
          }.bind(this))
        }
      })
    })
  }
})
```

UI Boot Camp: JS Functions

Promises

This is why JavaScript Promise libraries like Bluebird and Q got so much traction.

This method did not remove the use of callbacks, but it made the chaining of functions straightforward and simplified the code, making it much easier to read.

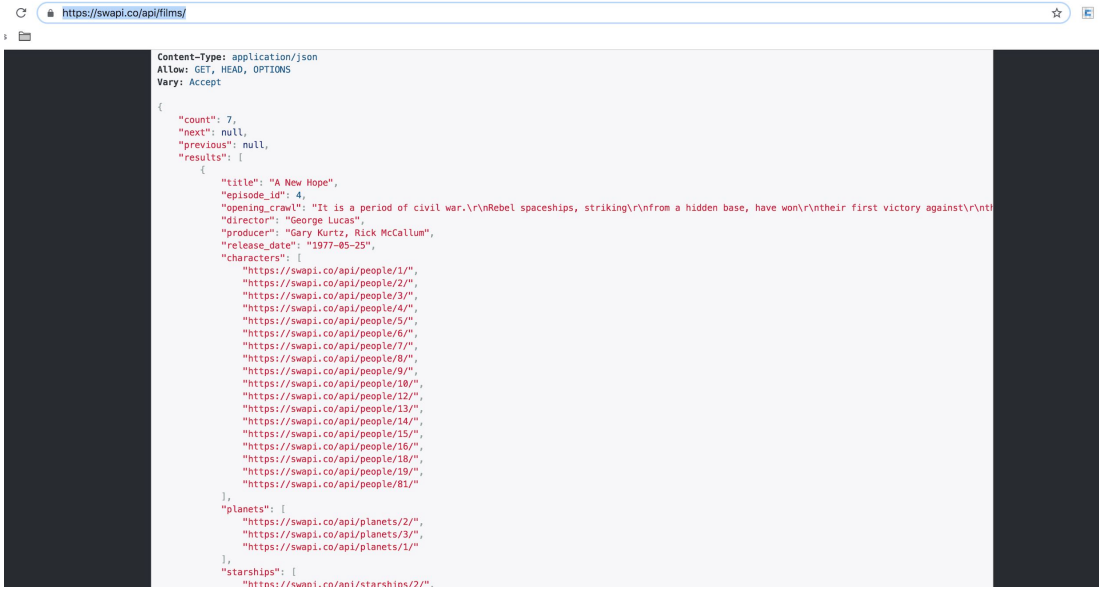
MDN

```
var promise1 = new Promise(function(resolve, reject) {
  setTimeout(function() {
    resolve('foo');
  }, 300);
});

promise1.then(function(value) {
  console.log(value);
  // expected output: "foo"
});

console.log(promise1);
// expected output: [object Promise]
```

UI Boot Camp: Basics JS Asyncronous

Homework: Async Await

https://swapi.co/api/



DIY: you have to study what are this features of javascript about, and bring some exercise done for next meeting

You might mdn as your reference, but the only one.

Scenario: given a list of films, create a <select> element that list them all.
When pickup one you should display details about the first of each attribute

UI Boot Camp: Basics JS Asyncronous

Homework: Stars wars APIs

https://swapi.co/api/

Styles colors and aspects should just be the one the browser use by default.

UI Boot Camp: Basics JS Asyncronous

Homework: Stars wars APIs | Fetch API

So, to accomplish this challenge you need a way of request to a backend services

We propouse you to use Fetch API, which provides an interface for fetching resources (including across the network). It will seem familiar to anyone who has used XMLHttpRequest, but the new API provides a more powerful and flexible feature set.

- Fetch API
- Using Fetch
- Response object: here you can find what methods you have to get information from the response

https://swapi.co/api/

Sample

```
fetch("https://swapi.co/api/films/").then((resp) => {
   console.log(resp.json())
});
```


Globant
we are ready