# THE COMPLETE JAVASCRIPT COURSE

FROM ZERO TO EXPERT!

**SECTION**

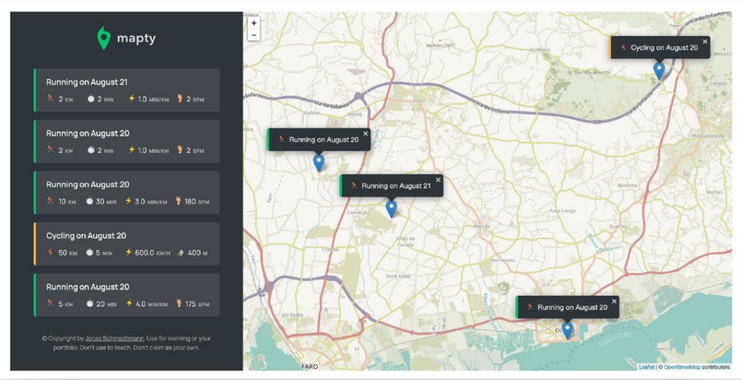
MAPTY APP: OOP, GEOLOCATION, EXTERNAL LIBRARIES, AND MORE!

**LECTURE**

FINAL CONSIDERATIONS

# 10ADDITIONAL FEATURE IDEAS: CHALLENGES

- 👉 Ability to **edit** a workout;
- 👉 Ability to **delete** a workout;
- 👉 Ability to **delete all** workouts;
- 👉 Ability to **sort** workouts by a certain field (e.g. distance);
- 👉 **Re-build** `Running` and `Cycling` objects coming from Local Storage;
- 👉 More realistic error and confirmation **messages**;
- 👉 Ability to position the map to **show all workouts** [very hard];
- 👉 Ability to **draw lines and shapes** instead of just points [very hard];
- 👉 **Geocode location** from coordinates (“Run in Faro, Portugal”) [only after asynchronous JavaScript section];
- 👉 **Display weather** data for workout time and place [only after asynchronous JavaScript section].



# ASYNCHRONOUS JAVASCRIPT: PROMISES, ASYNC/AWAIT AND AJAX

# THE COMPLETE JAVASCRIPT COURSE

FROM ZERO TO EXPERT!

**SECTION**

ASYNCHRONOUS JAVASCRIPT: PROMISES, ASYNC/AWAIT AND AJAX

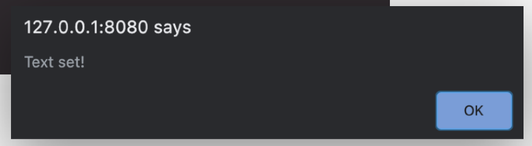
**LECTURE**

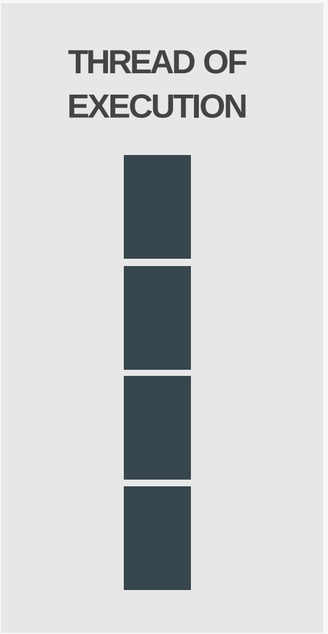
ASYNCHRONOUS JAVASCRIPT, AJAX AND APIS

# SYNCHRONOUS CODE

BLOCKING

```
const p = document.querySelector('.p');
p.textContent = 'My name is Jonas!';
alert('Text set!');
p.style.color = 'red';
```




Part of execution context that actually executes the code in computer's CPU

## SYNCHRONOUS

- 👉 Most code is **synchronous**;
- 👉 Synchronous code is **executed line by line**;
- 👉 Each line of code **waits** for previous line to finish;
- 👎 Long-running operations **block** code execution.

# ASYNCHRONOUS CODE

Asynchronous

CALLBACK WILL RUN AFTER TIMER

```
const p = document.querySelector('.p');
setTimeout(function () {
  p.textContent = 'My name is Jonas!';
}, 5000);
p.style.color = 'red';
```

👉 **Example:** Timer with callback

```
[1, 2, 3].map(v => v * 2);
```

Callback does NOT automatically make code asynchronous!





## ASYNCHRONOUS

Coordinating behavior of a program over a period of time

- 👉 Asynchronous code is executed **after a task that runs in the “background” finishes**;
- 👍 Asynchronous code is **non-blocking**;
- 👉 Execution doesn’t wait for an asynchronous task to finish its work;
- 👉 Callback functions alone do **NOT** make code asynchronous!

# ASYNCHRONOUS CODE

Asynchronous

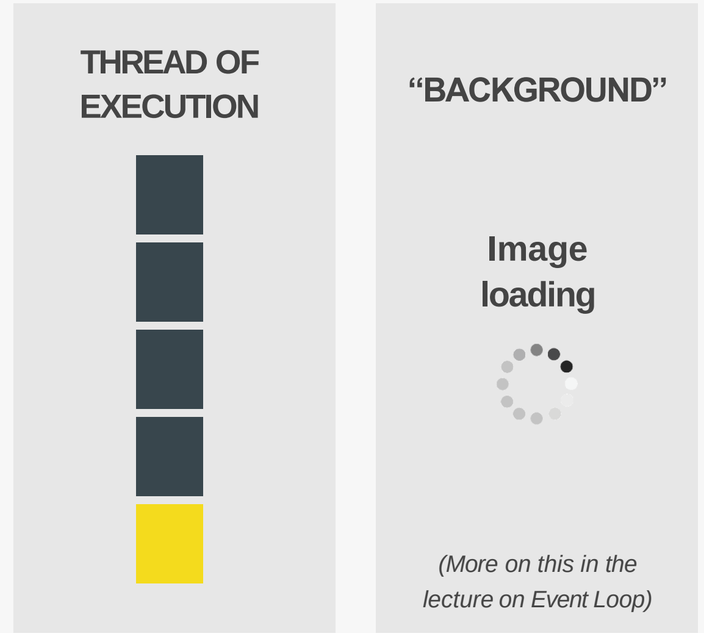
```
const img = document.querySelector('.dog');
img.src = 'dog.jpg';
img.addEventListener('load', function () {
  img.classList.add('fadeIn');
});
p.style.width = '300px';
```

CALLBACK WILL RUN AFTER IMAGE LOADS

👉 **Example:** Asynchronous image loading with event and callback

👉 **Other examples:** Geolocation API or **AJAX calls**

`addEventListener` does NOT automatically make code asynchronous!




## ASYNCHRONOUS

Coordinating behavior of a program over a period of time

👉 Asynchronous code is executed **after a task that runs in the “background” finishes**;

👍 Asynchronous code is **non-blocking**;

👉 Execution doesn’t wait for an asynchronous task to finish its work;

👉 Callback functions alone do **NOT** make code asynchronous!

# WHAT ARE AJAX CALLS?

**AJAX**

**A**synchronous **J**avaScript **A**nd **X**ML: Allows us to communicate with remote web servers in an **asynchronous way**. With AJAX calls, we can **request data** from web servers dynamically.

# WHAT IS AN API?

## API

- 👉 **A**pplication **P**rogramming **I**nterface: Piece of software that can be used by another piece of software, in order to allow **applications to talk to each other**;
- 👉 There are be many types of APIs in web development:

DOM API | Geolocation API | Own Class API | **"Online" API**

Just "API"

- 👉 **"Online" API:** Application running on a server, that receives requests for data, and sends data back as response;
- 👉 We can build **our own** web APIs (requires back-end development, e.g. with node.js) or use **3rd-party** APIs.


There is an API for *everything*

AJAX

XML

XML data format

JSON data format


Most popular API data format

- 👉 Weather data
- 👉 Data about countries
- 👉 Flights data ✈️ 🌍
- 👉 Currency conversion data
- 👉 APIs for sending email or SMS
- 👉 Google Maps
- 👉 *Millions of possibilities...*

# THE COMPLETE JAVASCRIPT COURSE

FROM ZERO TO EXPERT!

**SECTION**

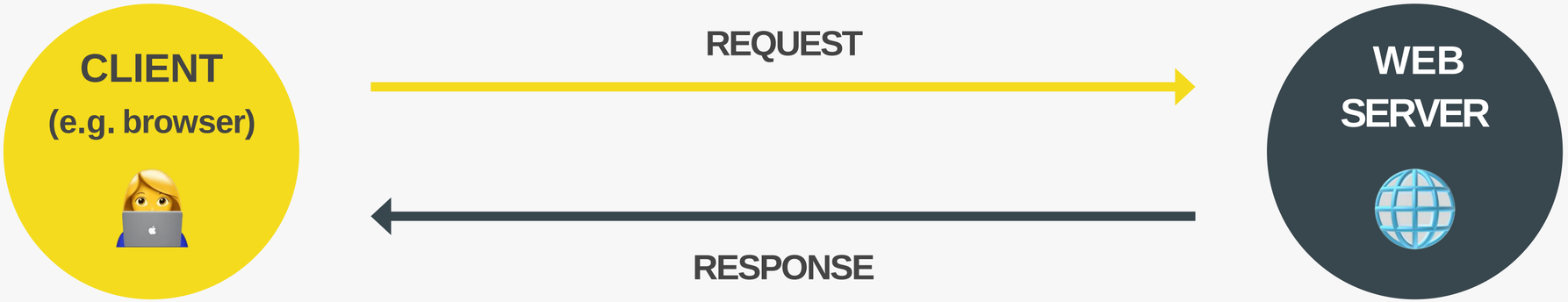
ASYNCHRONOUS JAVASCRIPT: PROMISES, ASYNC/AWAIT AND AJAX

**LECTURE**

HOW THE WEB WORKS: REQUESTS AND RESPONSES

# WHAT HAPPENS WHEN WE ACCESS A WEB SERVER

👉 **Request-response model** or **Client-server architecture**



# WHAT HAPPENS WHEN WE ACCESS A WEB SERVER