M01 - Introduction

•••

My First Web Application

My First Web Application

1. The script tag
2. Comments
3. Logging and Interaction
4. Hello World!

My First Web Application

1. The script tag

My First Web Application

1. The script tag

- Let's talk about JavaScript, the language itself.
- First we need a working environment to run our scripts:
  - server-side environments (like Node.js), you execute the script in a command like "node my.js".
  - client-side environments (like a browser), we usually embed JS scripts within a HTML page

My First Web Application

1. The script tag

- The tag <script>
  - To add JavaScript code to a page

```
<body>
    <h1>This is an example of the script tag!</h1>
    <script>
        console.log('Hello JS!');
    </script>
</body>
```

  - The <script> tag contains JavaScript code which is automatically executed when the browser processes the tag

My First Web Application

1. The script tag

- The tag <script>
  - If we have a lot of JavaScript code, we can put it into a separate file
  - Script files are attached to HTML with the src attribute:

```
9   <body>
10      <h1>This is an example of the script tag!</h1>
11      <script src="index.js"></script>
12  </body>
```

Reference to an external script file
(relative path)

My First Web Application

1. The script tag

- Best practices
  - As a rule, only the simplest scripts are put into HTML. More complex ones reside in separate files.
  - The benefit of a separate file is the browser will download it and store it in its cache.
  - Other pages that reference the same script will take it from the cache instead of downloading it, so the file is actually downloaded only once.
  - That reduces traffic and makes pages faster.

My First Web Application

2. Comments

My First Web Application

2. Comments

- As time goes on, programs become more and more complex. It becomes necessary to add comments which describe what the code does and why.
- Comments can be put into any place of a script. They don't affect its execution because the engine simply ignores them.
- One-line comments start with two forward slash characters //

```
// This is a comment
console.log('Hi world!');

console.log('Hi JS!'); // This is another comment
```

My First Web Application

2. Comments

- Multiline comments
  - start with a forward slash and an asterisk /*
  - end with an asterisk and a forward slash */.

```
/*
This is a comment
in two lines
*/
console.log('Hi world!');
```

My First Web Application

2. Comments

- In VSC use hotkeys to comment your code
  - For one line: CTRL+ ~
  - For multiple lines: SHIFT+ALT+A

My First Web Application

2. Comments

- VS Code supports the specification JSDoc and uses annotations to deliver a more advance Intellisense
- You can create more quickly JSDoc comments to functions as:
  - type /** before the function declaration
  - select the suggestion of the snippet of the JSDoc comment

```
/**
 * Função para somar dois números
 * @param {number} a primeiro operando
 * @param {number} b segundo operando
 * @returns {number} resultado da soma dos dois operandos
 */
function add(a, b) {
  return a + b;
}
```

My First Web Application

2. Comments

```
/**
 * Função para somar dois números
 * @param {number} a primeiro operando
 * @param {number} b segundo op
 * @returns {number} resultado
 */
function add(a, b) {
  return a + b;
}
```

```
* Função para somar dois numeros
fu
}
```

add
addEventListener
addEventListener
AudioDestinationNode
Audio
AudioBuffer
AudioBufferSourceNode
AudioContext
AudioListener
AudioNode
AudioParam
AudioParamMap

ad

```
function add(a: number, b: number): nu
mber
```

Função para somar dois números

@param `a` — primeiro operando

@param `b` — segundo operando

@returns — resultado da soma dos dois operandos

Documentation

Intellisense

My First Web Application

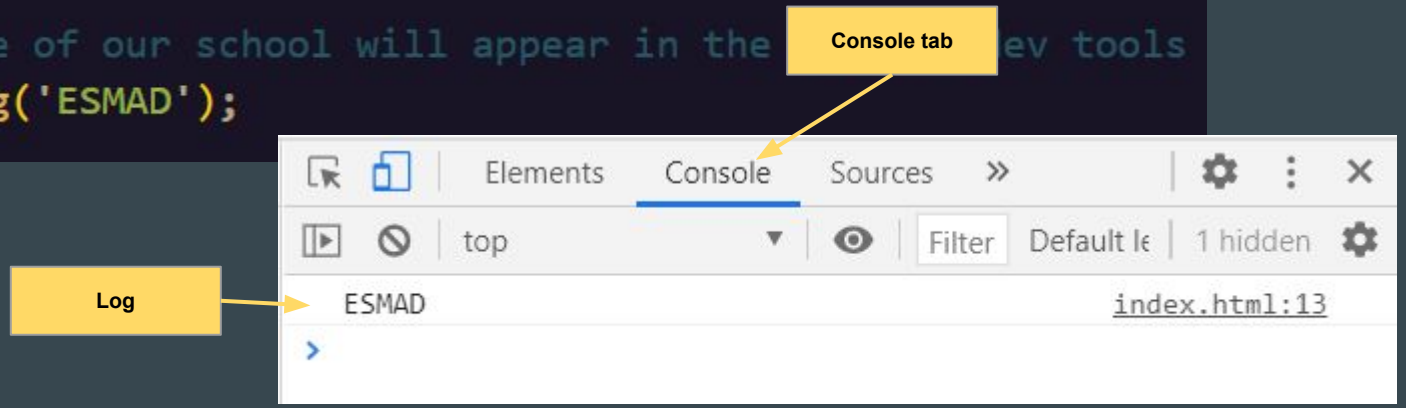
3. Logging and Interaction

My First Web Application

3. Logging and Interaction

- Logging
  - To output something to console from our code, there's console.log function.
  - Next example outputs the name ESMAD in the console tab of the Chrome DevTools:

```
// The name of our school will appear in the dev tools
console.log('ESMAD');
```



My First Web Application

3. Logging and Interaction

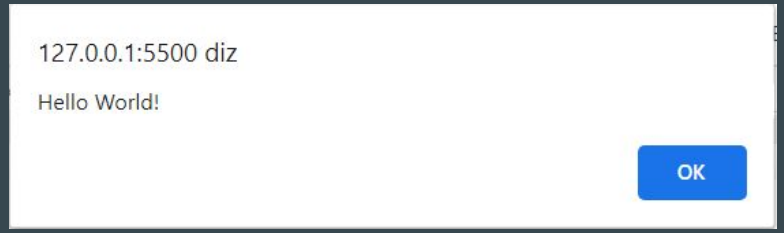
- Interaction
  - Important browser functions for user interaction:
  - Functions alert, prompt and confirm
  - Alert
    - shows a message
  - Prompt
    - shows a message asking the user to enter text
    - returns the text or, if the Cancel or Esc buttons are clicked, returns null
  - Confirm
    - shows a message and expects the user to press Ok or Cancel
    - returns true for OK and false for Cancel or Esc

My First Web Application

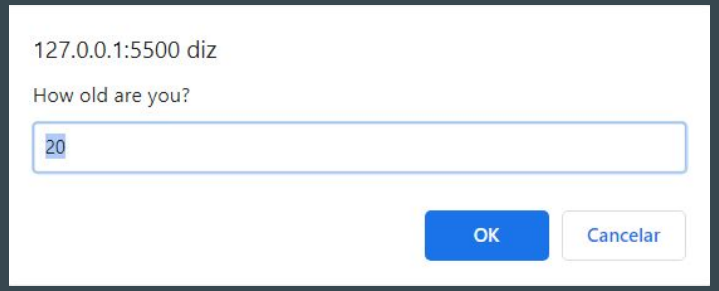
3. Logging and Interaction

- alert(message)
  - shows a message (message)
  - pauses script execution until the user presses OK

```
alert('Hello World!');
```



My First Web Application



3. Logging and Interaction

- prompt(title[,default])
  - Shows a modal window with text message, input field and OK/Cancel buttons
  - Two parameters:
    - title - the text to show to the user
    - default - an optional second parameter, the initial value of the input field
  - The user can:
    - type something in the input field and press OK
    - cancel the entry by pressing the Cancel button or the ESC key
  - The function returns the text of the entry field or null if the entry was canceled

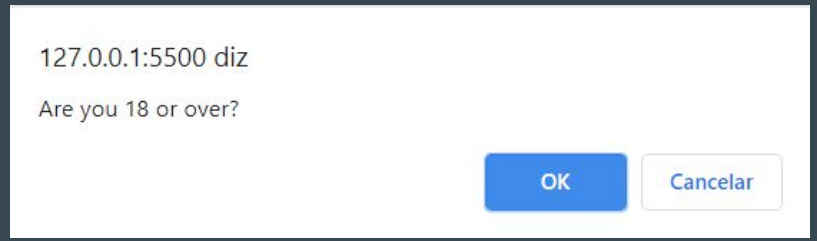
```
let age = prompt("How old are you?", 20);
alert(`You are ${age} years!`); // You are 20 years!
```

My First Web Application

3. Logging and Interaction

- confirm(question)
  - Shows a modal window with a question and two buttons: OK and Cancel
  - The result is true if OK is pressed and false otherwise

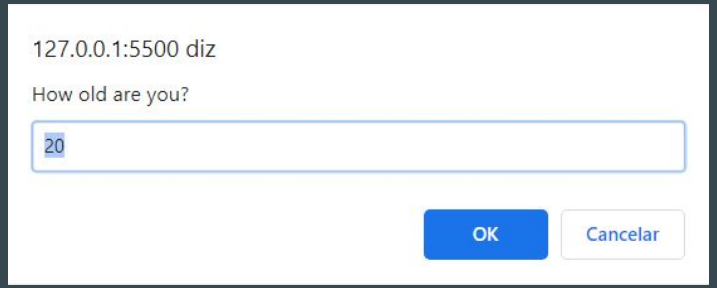
```
let over18 = confirm("Are you 18 or over?");
alert(over18); // true if OK button is pressed, false otherwise
```



My First Web Application

3. Logging and Interaction

- Interaction
  - All of these functions display modal windows: they stop the script from running and do not allow the visitor to interact with the page until the window is discarded
  - There are two limitations shared by all of the above functions:
    - The exact location of the window is defined by the browser. Usually, in the center.
    - The exact appearance also depends on the browser. We cannot change it.



My First Web Application

4. Hello World!

My First Web Application

4. Hello World

- Challenge
  - Create an app Hello World using HTML and JavaScript
  - The message should appear in the console of the browser
- Tasks:
  - Create a folder myFirstApp
  - Open VSC
  - Open folder created previously
  - Create a file index.html
  - Put a tag <script> and inside, a console.log statement that logs “Hello World!”
  - Open the file in a browser through Live Server