JavaScript

Why learn pure JavaScript?

- Learning JavaScript is like learning how to build a house from scratch!

Vanilla JavaScript allows us to

- Actually use frameworks in the future
  - Many times we are going to use pure Js in our code, so we need to be able to do so
  - We must understand what we are writing and why

- create custom configurations independent to the framework/library we use

Why learn JavaScript?

Vanilla JavaScript allows us to:

- Have better quality in our code
- Fix probable issues in our code more easily

- We do not want to become tied down to frameworks/ libraries that won't last forever.

- Easily learn new frameworks/ libraries
  - We are able to actually understand what we see in a piece of code for example

To begin with…

- We are going to use simple HTML pages in order to see some basic concepts/usage of pure JavaScript

Html5

HTML5 introduction

- Hyper Text Markup Language, better known simply as HTML, is the standard language used for building websites.
- HTML describes the structure of a Web page and consists of a series of *elements*
- *Elements* determine the content displayed on the browser (i.e. image, link, icon etc.)

HTML Basics 1/2

- HTML documents must start with a document type declaration: **<!DOCTYPE html>:** helps browsers to display web pages correctly

- The HTML document itself is structured **<html> </html>**

- The visible part of the HTML document is between **<body> </body>**

HTML Basics 2/2

- An example of HTML structure for a simple "Hello World" message:

```
<!DOCTYPE html>
<html>
<body>
<p>Hello World</p>
</body>
</html>
```

HTML Elements

- An HTML **element** is an individual component of an HTML document.
- Represents semantics or meaning
  - i.e title element represents the title of the document.
- Most HTML elements start with an opening **tag** and end with the closing tag. Content is placed in between.
- Elements can also contain **attributes** that define their additional properties. For example, a paragraph, which is represented by the p element, would be written as:

HTML Elements

- i.e a paragraph with attribute would be written:

This is a paragraph with attribute

Elements example

- Below are some examples of elements in HTML

<h1> My First Heading </h1>

 My first paragraph

- Empty elements: do not have contents and end tag

i.e.

Nested HTML Elements Example

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Nested Elements </title>
  </head>
  <body>

    <h1>Hello <b>World</b> </h1>
    <p>Do not be afraid to <u>underline</u> what's important</p>
  </body>
</html>
```

Basic HTML Elements

Tag	Description
<!DOCTYPE>	Defines the document type
<html>	Defines an HTML document
<head>	Contains information regarding the document
<title>	Title of the document
<body>	Defines the document's body
<h1> to <h6>	Defines HTML headings
<p>	Paragraph

Basic HTML Elements

Tag	Description
 	Inserts single line break
<a>	Hyperlink
<button>	A clickable button
<img>	Defines an image
<map>	Defines a client-side image map
<link>	Link to an external resource (most used to link to style sheets)
	Defines a section in a document

Basic HTML Elements

Tag	Description
<header>	Defines a header for a document or section
<footer>	Defines a footer for a document or section
<script>	Defines a client-side script
	A section in a document

To begin with…

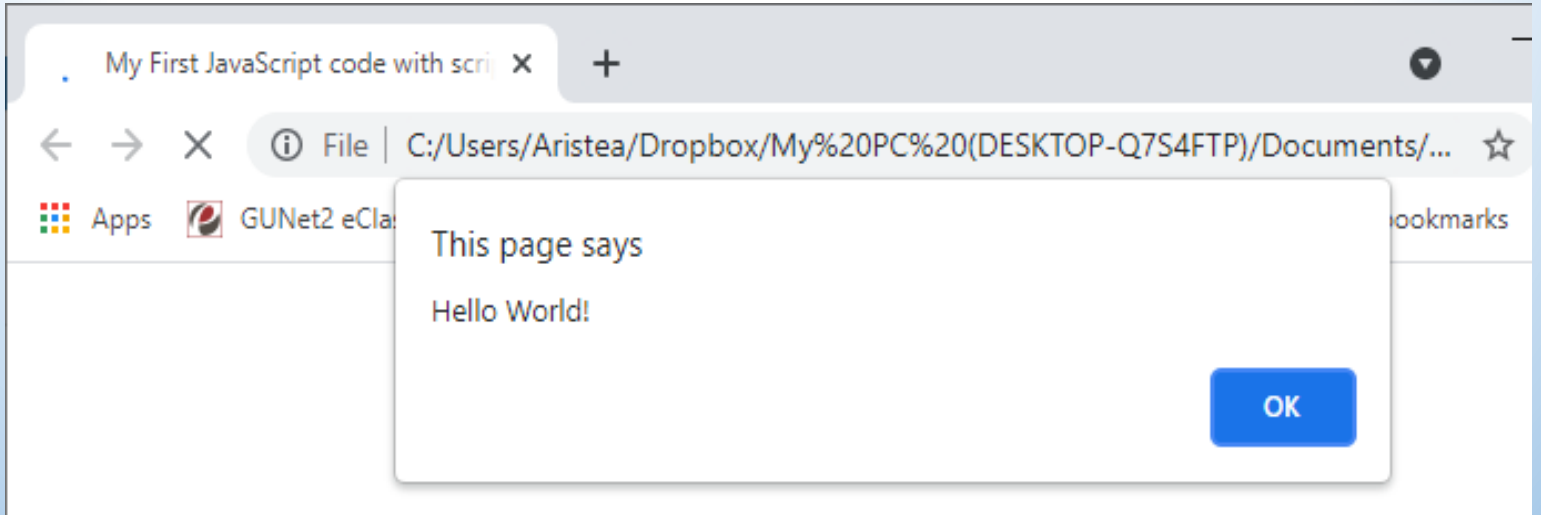
- HTML allows us to use Javascript using:
  - the <script> Tag
  - an External JavaScript

<script> Tag

- The <script> tag is used when placing the JavaScript code **within** the HTML document.
- <script> JavaScript code </script>
- Note: in HTML5 <script type="text/javascript"> is not required
- JavaScript is the default scripting language in HTML

<script> Tag Example

```
1   <html>
2   <head>
3       <title>My First JavaScript code with script tag!!!</title>
4       <script>
5           alert("Hello World!");
6       </script>
7   </head>
8   <body>
9   </body>
10  </html>
```



Js in HTML document

- More than one script can be placed in an HTML document

- The aforementioned scripts can be placed
- in the <body>
- in the <head>
- or in both in an HTML page

- Best practice: placing scripts at the bottom of the <body> element improves the display speed

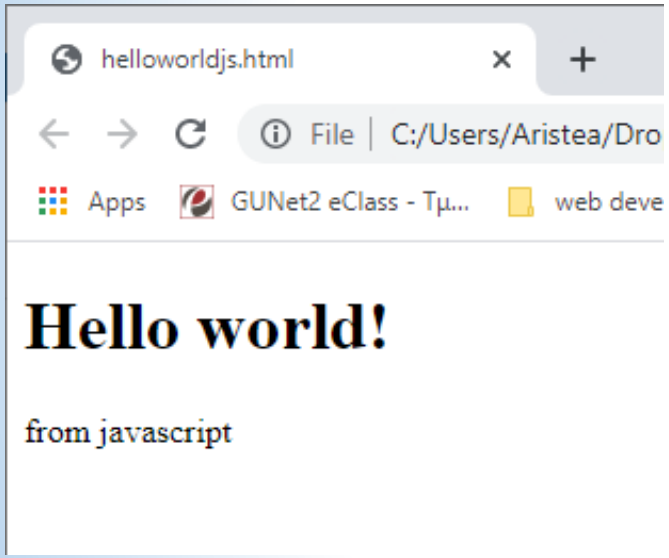
External JavaScript

- common practice-> place scripts in external files
- JavaScript files : .js extension

- External scripts are quite useful:
  - when we wish to reuse code
  - So as to keep HTML and JS separated
  - Enhances maintenance and understanding of scripts
  - speeds up page loads

Use JS to display data

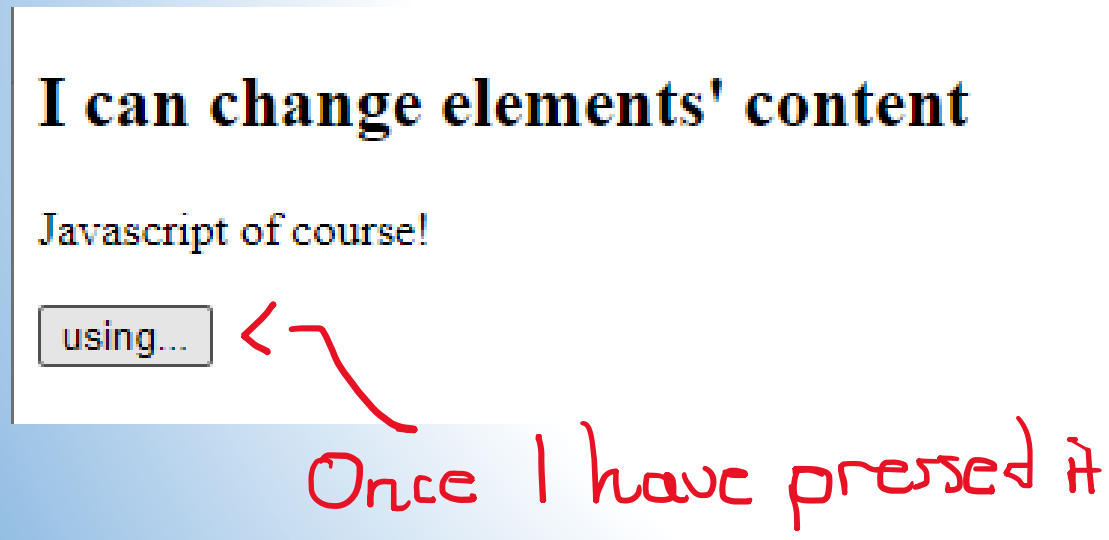
- document.write() : Writing into HTML displayed output




```
<body>

<script>

document.write("<h1>Hello world!</h1>");

document.write("<p>from javascript</p>");

</script>

</body>
```

Use JS to display data

- select an element by id
- innerHTML: write into an HTML element



print.html | 22_GetByid.html

```
<!DOCTYPE html>
<html>
<body>

<h2>I can change elements' content</h2>

<p id="example">Lets start doing some magic.</p>

<button type="button" onclick='document.
    getElementById("example").innerHTML = "
    Javascript of course!"'>using...</button>

</body>
</html>
```

Use JS to display data

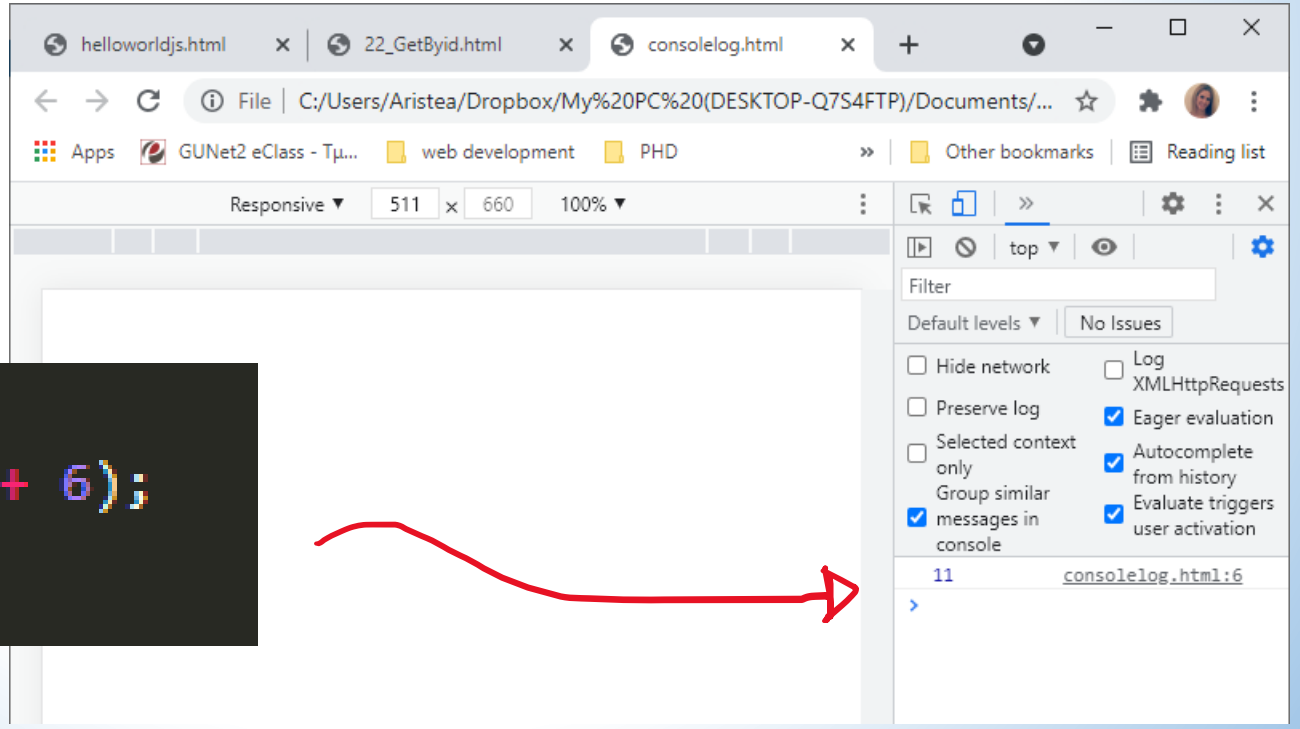
- window.alert() : Writing into an alert box

```
    <title>My First JavaScript code with script tag!!!</title>
    <script>
        alert("Hello World!");
    </script>
</head>
```

Use JS to display data

- console.log(): write into the browser console

```
<script>
console.log(5 + 6);
</script>
```



JavaScript Print

- window.print() method : print the content of the current window.

```
<button onclick="window.print()">Print this page</button>
```

JS Comments

- Comments are useful in Javascript code in order to make it more readable or just to prevent execution while testing

- Single line comments: //
- Multiple line comments

```
/*
These are
Comments of course!
*/
```

To be continued…