Introduction to web development

HTML + CSS + Javascript

OPEN MINDS

Goals

Learn the basics of web technologies:

- HTML to create the document (web page) structure and content.
- CSS to control its visual aspect.
- Javascript for dynamic content and interactivity.

Before we go !

What do we need to get started ?

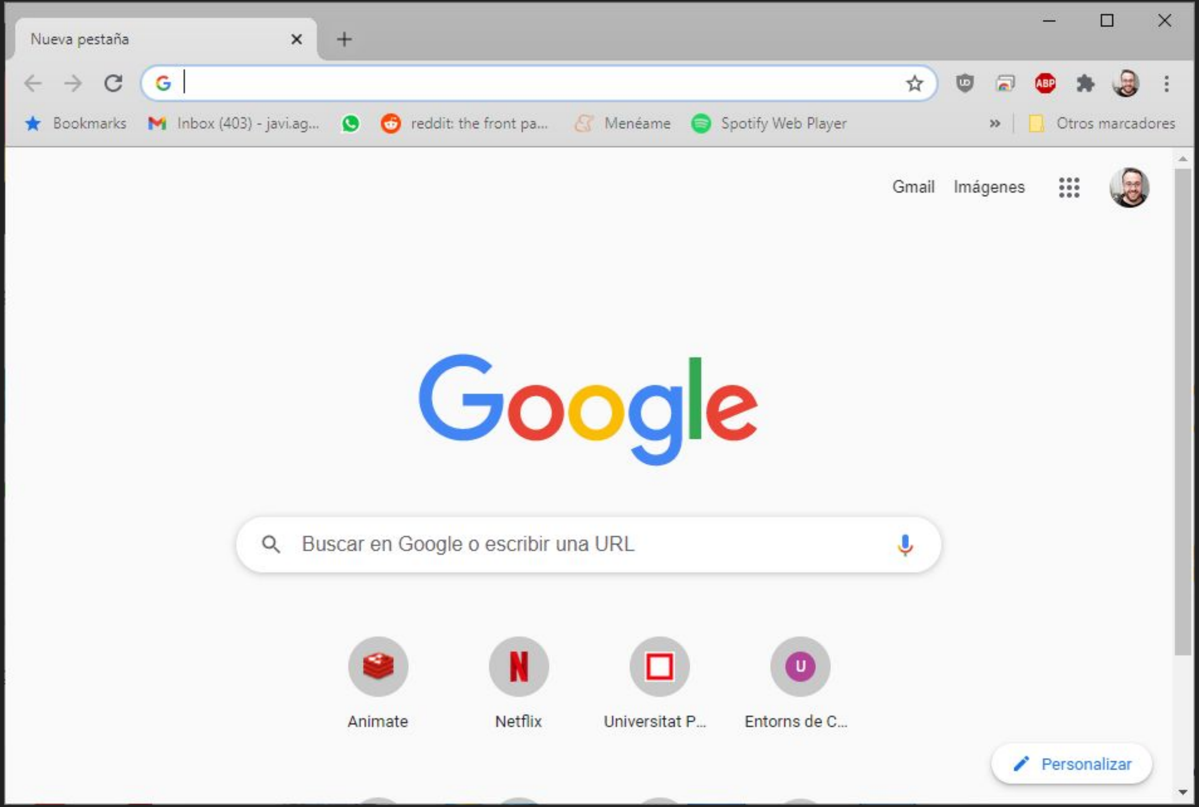
- a web-browser (Chrome or Firefox)
- a good text editor like **VSCode**, vim, atom, sublime, …

How can I test my code ?

Just open the index.html in your text editor and in your browser.
When you do any change to the code, check it in the browser by pressing F5 (to refresh the page).

To open the developer tools press:
Linux / Windows: `Control + Shift + I  or F12`
OSX: `Command + Opt + I`

Anatomy of a Browser



Inside a browser

Browsers have very differentiate parts.

We are interested in two of them:

- the Rendering Engine (in charge of transforming our HTML+CSS in a visual image).
- The Javascript Interpreter (also known as VM), in charge of executing the Javascript code.

Technologies

- HTML
- CSS
- Javascript

HTML

HTML means Hyper Text Markup Language.

The HTML allow us to construct the visible part of a website.

HTML is **NOT** a programming language, it’s a markup language, which means its purpose is to give structure to the content of the website.

It is a series of nested tags (it is a subset of XML) that contain all the website information (like texts, images and videos). Here is an example of tags:

```
<title>This is a title</title>
```

```
<html>
     <head>
     </head>
     <body>
          <div>
               <p>Hi</p>
          </div>
     </body>
</html>
```

HTML: basic rules

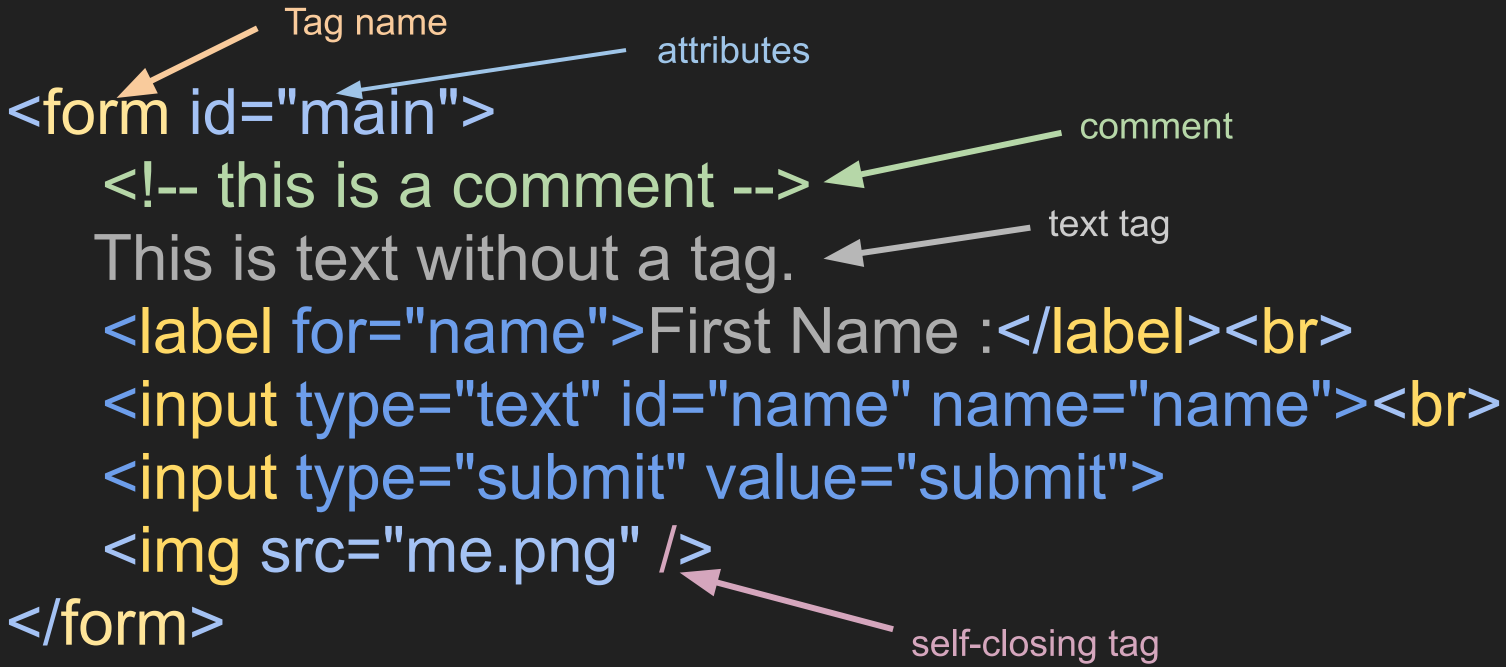
Some rules about HTML:

- It uses XML syntax (tags with attributes, can contain other tags).
  ```
  <tag_name attribute="value"> content </tag_name>
  ```
- It stores all the information that must be shown to the user.
- There are different HTML elements for different types of information and behaviour.
- The information is stored in a tree-like structure (nodes that contain nodes inside) called DOM (Document Object Model).
- It gives the document some semantic structure (pe. this is a title, this is a section, this is a form) which is helpful for computers to understand websites content.
- It must not contain information related to how it should be displayed (that information belongs to the CSS), so no color information, font size, position, etc.

HTML: syntax example

```html
<div id="main">
     <!-- this is a comment -->
     This is text without a tag.
     <button class="mini">press me</button>
     <img src="me.png" />
</div>
```

HTML: syntax example

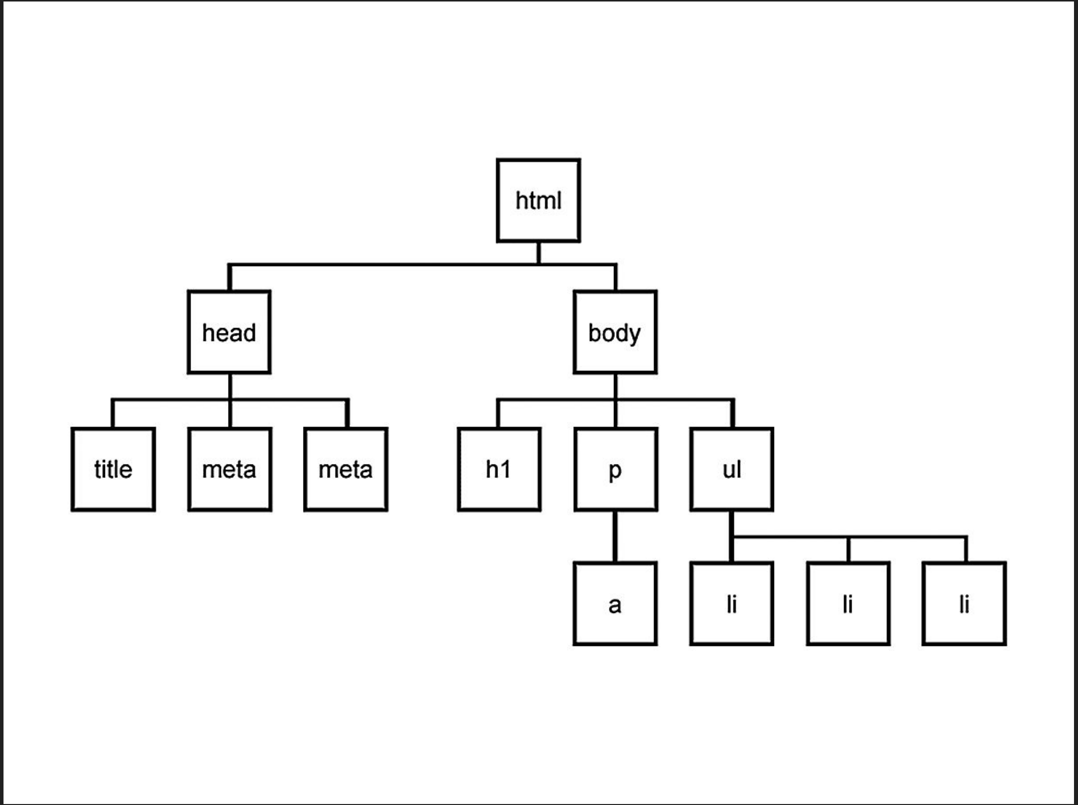


```html
<form id="main">
    <!-- this is a comment -->
    This is text without a tag.
    <label for="name">First Name :</label><br>
    <input type="text" id="name" name="name"><br>
    <input type="submit" value="submit">
    <img src="me.png" />
</form>
```

DOM is a tree

Every node can only have one parent, and every node can have several children, so the structure looks like a tree.

```
<html>
     <head>
     </head>
     <body>
          <div>
               <p>Hi</p>
               <p>Bye</p>
          </div>
     </body>
</html>
```



HTML: most common tags

Although there are lots of tags in the HTML specification, 99% of the webs use a subset of HTML tags with less that 10 tags, the most important are:

- <div>: a container, usually represents a rectangular area with information inside.
- <img/>: an image
- <a>: a clickable link to go to another URL
- <p>: a text paragraph
- <h1>: a title (h2,h3,h4 are titles of less importance)
- <style>: to insert CSS rules
- <script>: to execute Javascript
- <span>: a simple inline element

HTML: most common tags

Although there are lots of tags in the HTML specification, 99% of the webs use a subset of HTML tags with less that 10 tags, the most important are:

- <form>: used to create an HTML form to collect user input.
- <label>: defines a label for several form elements.
- <input>: a widget to let the user enter information.

HTML: other interesting tags

There are some tags that could be useful sometimes:

- <button>: defines a clickable button.
- <audio>: used to play an audio file on a web page
- <video>: used to show a video on a web page.
- <canvas>: used to draw graphics on web page
- <iframe>: used to embed another document within the current HTML document.

HTML good use

It is good to have all the information properly wrapped in tags that give it some semantics.

We also can extend the code semantics by adding extra attributes to the tags:

- id: tells a unique identifier for this tag
- class: tells a generic identifier for this tag

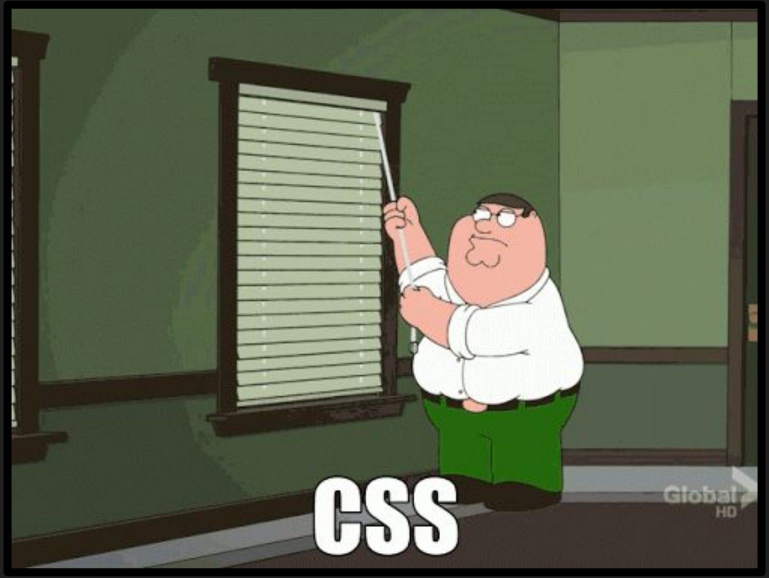
```
<div id="profile-picture" class="mini-image">...</div>
```

HTML references

HTML Reference: a description of all HTML tags.

Technologies

- HTML
- CSS
- Javascript

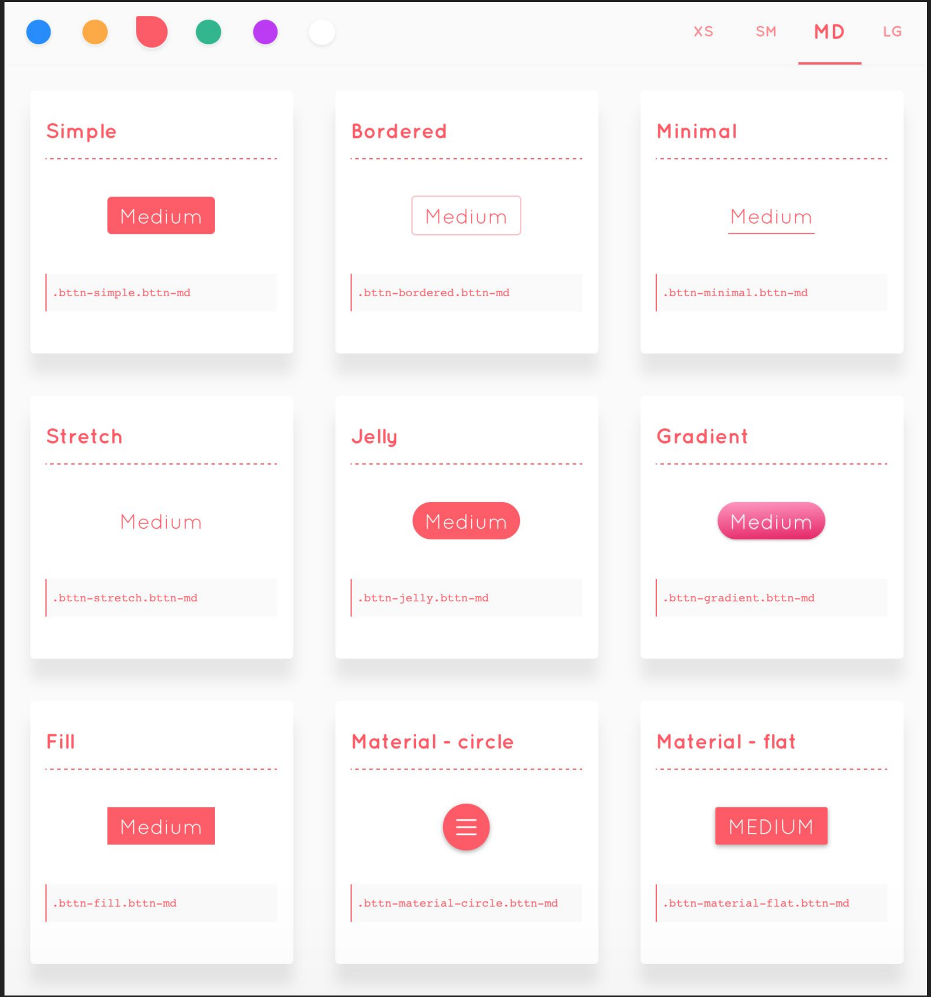


CSS

Css is a language we use to style an HTML document

Allows to controls all the aspects of the visualization and some other features:

- Colors: content, background, borders
- Margins: interior margin, exterior margin
- Position: where to place the element
- Sizes: control the width, height
- Behaviour: style element when a user mouses over it.



```
.bttn-simple.bttn-md
.bttn-bordered.bttn-md
.bttn-minimal.bttn-md
.bttn-stretch.bttn-md
.bttn-jelly.bttn-md
.bttn-gradient.bttn-md
.bttn-fill.bttn-md
.bttn-material-circle.bttn-md
.bttn-material-flat.bttn-md
```

CSS example

```css
* {
    color: blue; /*a comment */
    margin: 10px;
    font: 14px Tahoma;
}
```

This will change the style of all the HTML elements (tags) in our web page (‘*‘ means all)

This will set the text color to blue and the font family to Tahoma with a 14px font size, and set the margin to 10px around.

CSS fields

Here is a list of the most common CSS fields and an example:

- `color: #FF0000;    red;     rgba(255,00,100,1.0); //different ways to specify colors`
- `background-color: red;`
- `background-image: url('file.png');`
- `font: 18px 'Tahoma';`
- `border: 2px solid black;`
- `border-top: 2px solid red;`
- `border-radius: 2px; //to remove corners and make them more round`
- `margin: 10px; //distance from the border to the outer elements`
- `padding: 2px; //distance from the border to the inner elements`
- `width: 100%;    300px;    1.3em;  //many different ways to specify distances`
- `height: 200px;`
- `text-align: center;`
- `box-shadow: 3px 3px 5px black;`
- `cursor: pointer;`
- `display: inline-block;`
- `overflow: hidden;`

CSS how to add it

There are three ways to add CSS rules to your website:

- Inserting the CSS code inside a style (defined in the head tag)

```
<style>
    p { color: blue }
</style>
```

- Referencing an external CSS file

```
<link href="style.css" rel="stylesheet" />
```

- Using the attribute style on a tag

```
<p style="color: blue; margin: 10px">
```

CSS selectors

What if we want to change one specific tag (not all the tags of the same type).

We can specify more precise selectors besides the name of the tag. For instance, by class or id:

```
p.intro {
    color: red;
}
```

This will affect only the tag <p class="intro">

CSS Selectors

The main selectors are:

- tag name: just the name of the tag
  - `p { ... }   //affects to all <p> tags`
- dot (.): selects the HTML element with the class name mentioned after the dot
  - `p.highlight { ... } //affects all <p> tags with class="highlight"`
- sharp character (#): selects the HTML element with the id after the #.
  - `p#intro { ... } //affects to the <p> tag with the id="intro"`
- two dots (:): behaviour states (mouse on top)
  - `p:hover { ... } //affects to <p> tags with the mouse over`
- brackets ([attr='value']): tags with the attribute attr with the value 'value'
  - `input[type="text"] {...} // affects to the input tags of the type text`

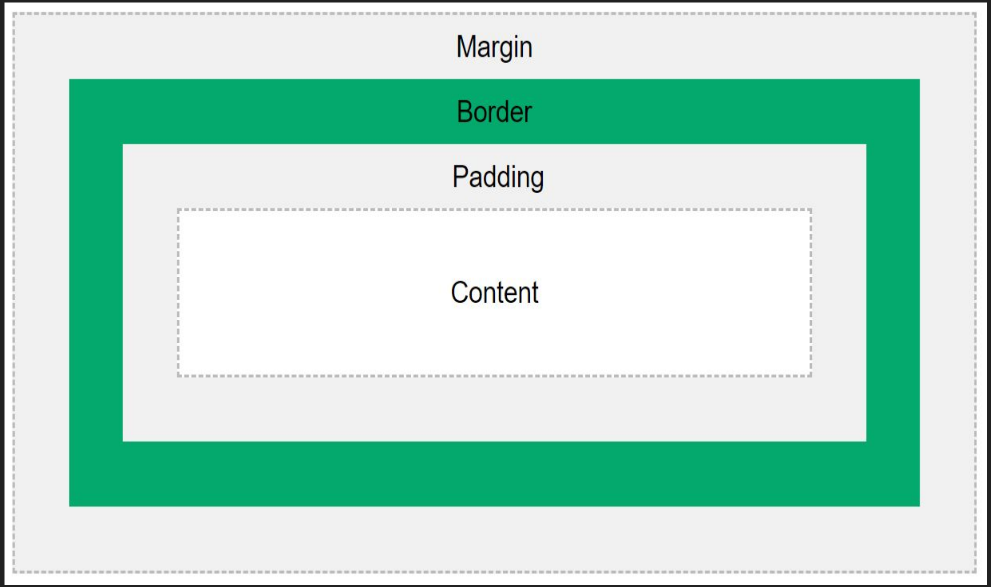
Box Model

It is important to note that by default any width and height specified to an element will not take into account its margin, so a div with width 100px and margin 10px will measure 120px on the screen, not 100px.

This could be a problem breaking your layout.

You can change this behaviour changing the box model of the element so the width uses the outmost border:

```
div { box-sizing: border; }
```

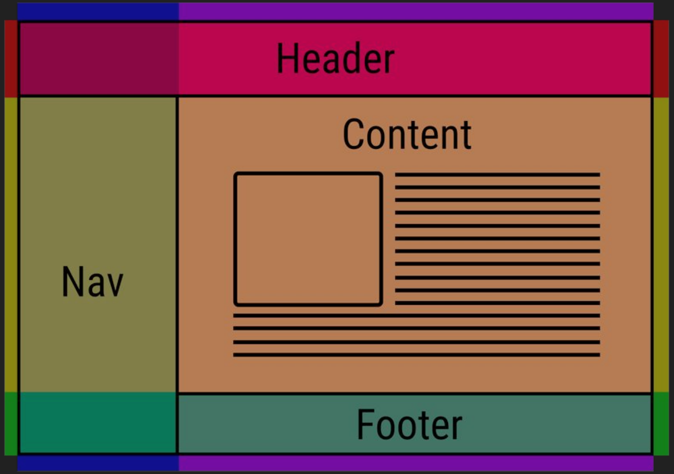


Layout

One of the hardest parts of CSS is construing the layout of your website (the structure inside the window) .

By default HTML tends to put everything in one column, which is not ideal.

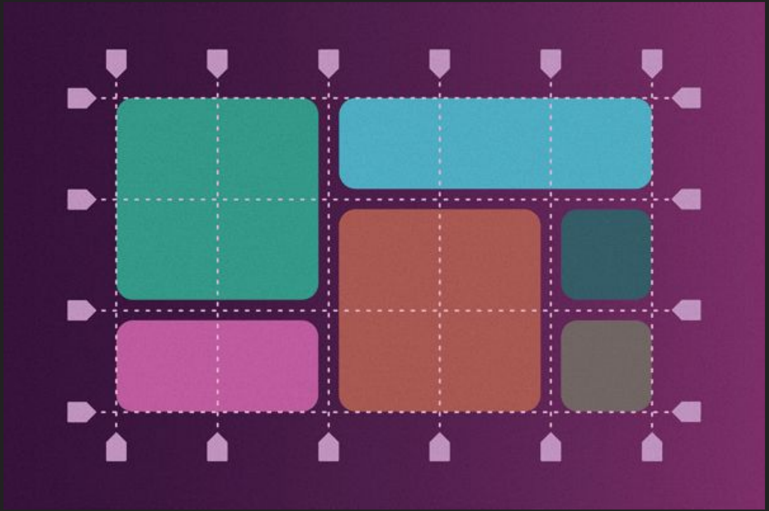
There has been many proposals in CSS to address this issue (tables, fixed divs, flex, grid, …).



Grid system

Because most sites are structured in a grid, I recommend to use the CSS Grid system.

Check this tutorial to create the site structure easily



HTML

```
<div class="grid-container">
  <div class="grid-item1">1</div>
  <div class="grid-item2">2</div>
</div>
```

CSS

```
.grid-container {
  display: grid;
  grid-template-rows: 100px; 100px;
  grid-template-columns: 100px; 100px; 100px;
  grid-gap: 5px;
}

.grid-item1 {
  background: blue;
  border: black 5px solid;
  grid-column-start: 1;
  grid-column-end: 5;
  grid-row-start: 1;
  grid-row-end: 3;
}
```

Fullscreen divs

Sometimes we want to have a div that covers the whole screen (to make a webapp), instead of a scrolling website (more like regular documents).

In that case remember to use percentages to define the size of elements, but keep in mind that percentages are relative to the element's parent size, so you must set the size to the <body> element to use 100%.

CSS

```css
html, body {
	width: 100%;
	height: 100%;
}

div {
	margin: 0;
	padding: 0;
}

#main {
	width: 100%;
	height: 100%;
}
```

Trick to center

Centering divs can be hard sometimes, use this trick:

```css
.horizontal-and-vertical-centering {
  display: flex;
  justify-content: center;
  align-items: center;
}
```

CSS further reading

There are many more rules for selectors.

Check some of the links to understand them better.

Understanding the Box Model: a good explanation of how to position the information on your document.

All CSS Selectors: the CSS selectors specification page.

CSS Transition: how to make animations just using CSS

Technologies

- HTML
- CSS
- Javascript

Javascript

A regular programming language, easy to start.

Allows to give some interactivity to the elements of the web page.

Javascript has no variable types.

You can change the content of the HTML or the CSS applied to an element.

You can even send or retrieve information from the internet to update the content of the web without reloading the page.

Javascript: where to start ?

There is three ways to add a javascript code into a web page:

- Embed the code in the HTML using the <script> tag.

```
<script> /* some code */ </script>
```

- Import an external Javascript file using the <script> tag:

```
<script src="file.js" />
```

- Inject the code on an event inside a tag:

```
<button onclick="javascript: /*code*/">press me</button>
```

Javascript: Syntax

```
var my_number = 10;

// this is a comment
var my_string = "hello";
var my_array = [10,20,"name",true];
var my_object = {

    name:"Open Minds Club",

    teams: ["IT","IT","IT","IT","IT","IT"] };

function sayHello( name )
{
      console.log(" Hello, " + name );
}
```

Javascript example

```
<html>
    <body>
        <h1>This is a title</h1>
        <script>
            var title ="Welcome to our website";
            alert(title);
        </script>
    </body>
</html>
```

Javascript API

Javascript comes with a rich API to do many things like:

- Access the DOM (HTML nodes)
- Do HTTP Requests
- Play videos and sounds
- Detect user actions (mouse move, key pressed)
- Launch Threads
- Access the GPU, get the Webcam image, ...

And the API keeps growing with every new update of the standard.

Check the WEB API reference to know more

Javascript: retrieving element

You can get elements from the DOM (HTML tree) using different approaches.

- Crawling the HTML tree (starting from the body, and traversing its children)
- Using a selector (like in CSS)
- Attaching events listeners (calling functions when some actions are performed)

Javascript: crawling the DOM

There are different global variables that you can access to get information about the web page:

- document: contains the DOM information (all the HTML elements)
- window: the browser window

The document variable allows to crawl the tree:

```
document.body.children[0] // returns the first node inside body tag
```

Javascript: using selectors

You can retrieve elements using selectors:

```
var nodes = document.querySelectorAll("p.intro");
```

will return an array with all `<p class="intro">` nodes.

Or if we have already a node and we want to search inside it:

```
var node = mynode.querySelectorAll("p.intro")
```

Javascript: modify nodes

changing attributes

```
mynode.id = "intro"; //sets an id
mynode.className = "important"; //set class
mynode.classList.add("good"); //to add a class to the element
```

Change the content

```
mynode.innerHTML = "<p>text to show</p>"; //change content
```

Modify the style (CSS)

```
mynode.style.color = "red"; //change any css properties
```

or add the behaviour of a node

```
mynode.addEventListener("click", function(e) {
     //do something
});
```

Javascript: create nodes

Create elements:

```
var element = document.createElement("div");
```

And attach them to the DOM:

```
document.querySelector("#main").appendChild( element );
```

Or remove it from its parent:

```
var element = document.querySelector("foo");
element.parentNode.removeChild( element );
```

Javascript: hide and show elements

Sometimes it may be useful to hide one element or show another.

You can change an element CSS directly by accessing its property style.

To avoid being displayed on the web change display to "none"

```
element.style.display = "none"; //hides elements from being rendered
element.style.display = ""; //displays it again
```

Example of a website

HTML in index.html

```
<link href="style.css" rel="stylesheet"/>
<h1>Welcome</h1>
<p>
     <button>Click me</button>
</p>
<script src=”code.js”/>
```

CSS in style.css

```
h1 { color: #333; }
button {
     border: 2px solid #AAA;
     background-color: #555;
}
```

Javascript in code.js

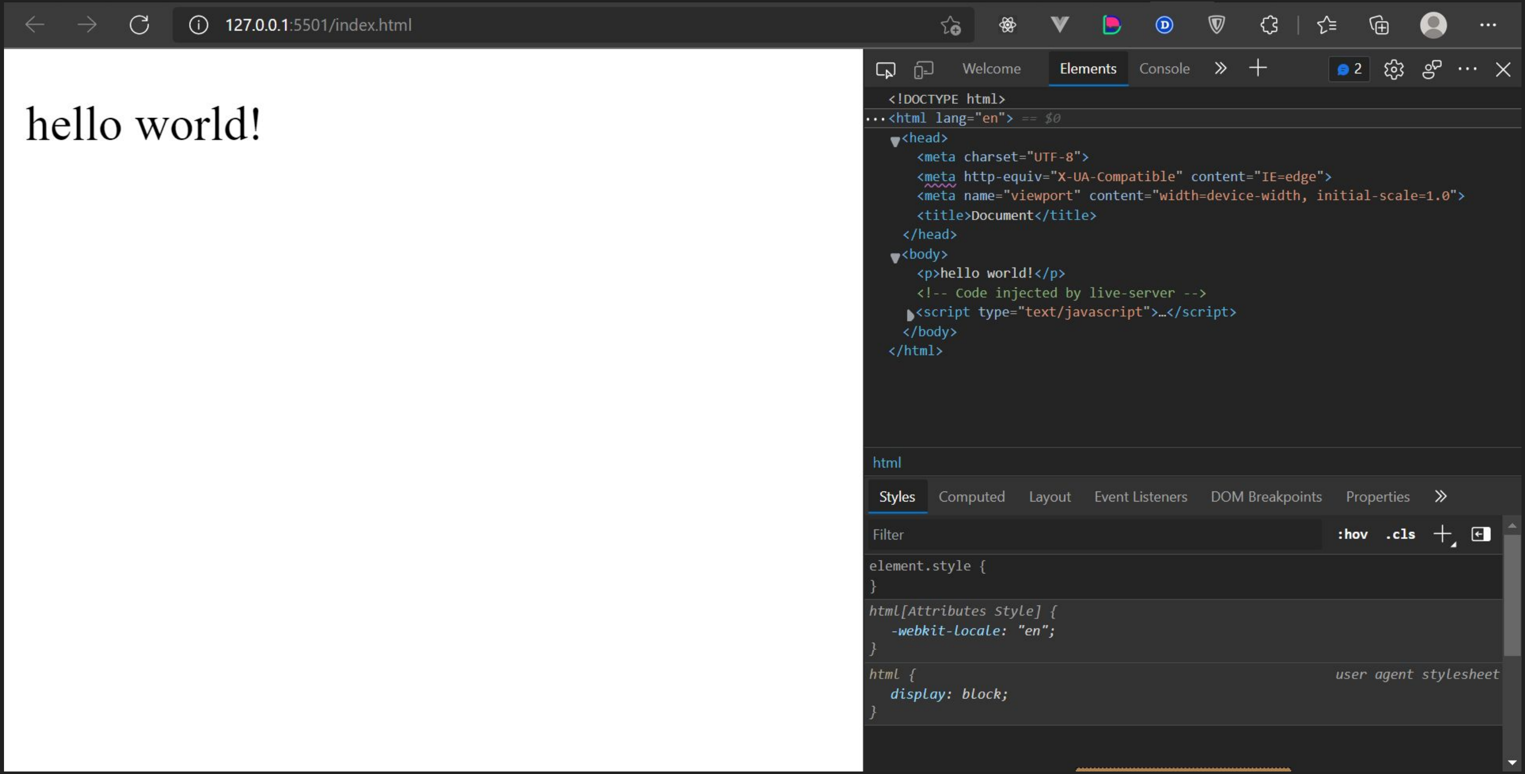
```
//fetch the button from the DOM
var button = document.querySelector(“button”);

//attach and event when the user clicks it
button.addEventListener(“click”, myfunction);

//create the function that will be called when the
button is pressed
function myfunction()
{
     //this shows a popup window
     alert(“button clicked!”);
}
```

Using the Dev Tools

Press Control + Shift + I (or F12) to open DevTools



hello world!

```
<!DOCTYPE html>
<html lang="en"> == $0
  <head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
  </head>
  <body>
    <p>hello world!</p>
    <!-- Code injected by live-server -->
    <script type="text/javascript">…</script>
  </body>
</html>
```

html

```
element.style {
}
html[Attributes Style] {
   -webkit-locale: "en";
}
html {                                    user agent stylesheet
   display: block;
}
```

Further info

HTML, CSS and JavaScript:

http://www.w3schools.com/

Aya c bon

Let’s Code !