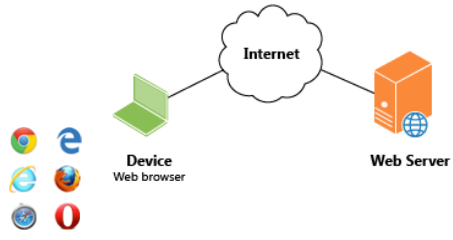
Client Side Web Technologies

Front End Development

Hands-on-session

Athens, May 2018
Simão Paredes *sparedes@isec.pt*

Client-Server model



Client Side Web Technologies

HyperText Markup Language (HTML)

HTML

- *Hyper Text Markup Language*
  - Markup language (tags)
    - HTML tags are **keywords** (tag names) surrounded by **angle brackets**

<tagname> content </tagname>

  - The global structure of an HTML document:

In the HTML5 standard, the <html> tag, the <body> tag, and the <head> tag can be omitted.

http://www.w3schools.com/html/html_head.asp

```
<html>
  <head>
    <title>Page title</title>
  </head>
  <body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
  </body>
</html>
```

Container for metadata
(data about data ; not displayed)

Content data (visible)

HTML

- Document Structure

```
<!DOCTYPE html>

<html lang="en">

<head>
   <meta charset="utf-8" />
   <title></title>
</head>

<body>

</body>

</html>
```

Version of HTML that is being used

Metadata (e.g.: **<title>**…</title>; **<style>** … </style>; **<link …. />**; **<meta … />**; **<script>** …. </script>; **<noscript>** …. </noscript>)

Visible content (e.g.: **<h1>**…</h1>; ** ** … ; ****; **<a>** …. , …)

http://www.w3schools.com/tags/

Task 1.1

Task 1

- Open Sublime Text 3
- Create a new file and save it as “exercise1.html”
- html:5 + tab

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
</head>
<body>

</body>
</html>
```


End

Task 1.1

HTML

- HTML element
  - Defined by an opening tag, a closing tag and the content
  - The content of an element is all that is defined between the opening tag and its closing tag.
    - Some elements have no content (empty elements) e.g.:
; <img >;…
- HTML Attribute
  - Attributes complement the HTML elements
  - May be required, i.e. without their information the tag is ignored by the browser
  - Attributes are always specified in the opening tag
  - Syntax of attributes is always performed as:
    - name = "value"
  - An element can have multiple attributes separated by spaces

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
</head>
<body>

</body>
</html>
```

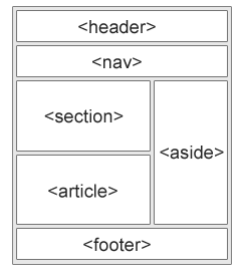

HTML

- HTML element
  - Defined by an opening tag, a closing tag and the content
  - The content of an element is all that is defined between the opening tag and its closing tag.
    - Some elements have no content (empty elements) e.g.:
; <img >;…
- HTML Attribute
  - Attributes complement the HTML elements
  - May be required, i.e. without their information the tag is ignored by the browser
  - Attributes are always specified in the opening tag
  - Syntax of attributes is always performed as:
    - name = "value"
  - An element can have multiple attributes separated by spaces

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
</head>
<body>

</body>
</html>
```

Structure Elements

- *Containers*
  - *** … ***
    - block-level element; container for other elements; …
  - *** … ***
    - inline element; container for text; …
- *Semantic tags*
  - *<header> … </header>*
  - *<nav> … </nav>*
  - *<section>…</section>*
  - *<article>…</article>*
  - *<aside>…</aside>*
  - *<footer>…</footer>*
  - *…*



Structure Elements

- *lists*
  - *** … : unordered list***
  - ***
… : list item***

```
<ul>
    <li>1</li>
    <li>2</li>
    <li>3</li>
</ul>
```

- 1
- 2
- 3

- *links*
  - *** link text***
  - *href attribute*
    - Absolute path: *href=" **http://**www.*** "*
    - *Relative path: href=" **/path ***"*

Task 1.2

Task 1.2

- Inside the <body> of the exercise1.html, create a header element with id="header“
- Inside the header element, create a nav element with id=“main-menu”
- Inside the nav element create an unordered list with four options, each one including a link:
  - destination: “index.html”; link text: “Home”
  - destination: “about.html”; link text: “About”
  - destination: “materials.html”; link text: “Materials”
  - destination: “contact.html”; link text: “Get in Touch”

Solution Task 1.2.

```
<body>
  <header id="header">
    <nav id="main-menu">
      <ul>
        <li><a href="index.html">Home</a></li>
        <li><a href="about.html">About</a></li>
        <li><a href="materials.html">Materials</a></li>
        <li><a href="contact.html">Get in Touch</a></li>
      </ul>
    </nav>
  </header>
```


End

Task 1.2

Structure Elements

- paragraph
  - *** … ***
    - different from carriage return **
**
- *headings*

```
<body>
    <h1>Heading 1</h1>
    <h2>Heading 2</h2>
    <h3>Heading 3</h3>
    <h4>Heading 4</h4>
    <h5>Heading 5</h5>
    <h6>Heading 6</h6>
</body>
```

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

- *buttons*

```
<button>Test it!!</button>
```

Test it!!

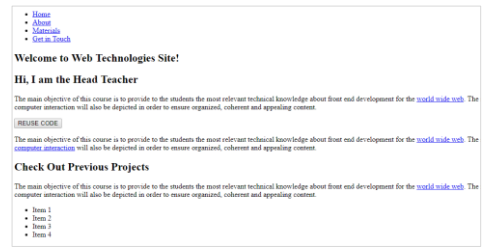
Task 1.3

Task 3

- copy the text from content.txt to the html file right after to the end of the header element

```
<div id="main">

  <article id="content">
    <h1>Welcome to Web Technologies Site!</h1>
    <section id="introduction">
      <h2>Hi, I am the Head Teacher</h2>
      <p>The main objective of this course is to provide to the students the most relevant technical knowledge about front end development for the
      Some concepts of human computer interaction will also be depicted in order to ensure organized, coherent and appealing content.</p>
      <button class="code-button">REUSE CODE</button>
      <p>The main objective of this course is to provide to the students the most relevant technical knowledge about front end development for the
      Some concepts of <a href="#">human computer interaction</a> will also be depicted in order to ensure organized, coherent and appealing content
    </section>

    <section id="promo">
      <h2>Check Out Previous Projects</h2>
      <p>The main objective of this course is to provide to the students the most relevant technical knowledge about front end development for the
      Some concepts of human computer interaction will also be depicted in order to ensure organized, coherent and appealing content.</p>
      </section>
  </article>

  <aside id="sidebar">
    <ul>
      <li>Item 1</li>
      <li>Item 2</li>
      <li>Item 3</li>
      <li>Item 4</li>
    </ul>
  </aside>

</div>

<footer id="footer"></footer>
```

- preview in the browser



End

Task 1.3

HTML

- There are two key attributes to provide access to specific content (formatting, layout definition, ...)
  - id = "value“
  - class = "value"
    - The ***id*** attribute is unique [identification]
    - The ***class*** attribute can be assigned to multiple elements [classification]

Outline

1. Structure Layer ✓
   - *Hyper Text Markup Language (HTML)*
2. Presentation Layer
   - *Cascading Style Sheets (CSS)*
3. Behaviour Layer
   - *JavaScript*

Cascading Style Sheets (CSS)

Cascading Style Sheets (CSS)

- Improvement of the graphical consistency of web sites:
  - CSS allows a clear separation between design and content
  - CSS defines all the formatting / layout (design)
  - CSS uses its own syntax
  - CSS Advantages:
    - Immediate application to multiple HTML documents
    - Ensures the same formatting rules (consistency)
    - Greater flexibility
    - Concentrated changes
    - Avoid repeating formats

Cascading Style Sheets (CSS)

- Link between HTML/CSS
  - Three different ways:
    - External Style Sheet (*.css)
      - There may be one or more HTML files for one or multiple CSS files
      - A link to the CSS file is included in the HTML file
    - Embedded Style Sheet
      - CSS code is directly incorporated in the HTML file
      - Less flexible than the previous option
    - Inline Style
      - Local formatting
      - This style **should be avoided!**

Cascading Style Sheets (CSS)

- ***External Style Sheet <link … />***
  - ***href*** attribute indicates the location of the (* .css) file
  - ***rel*** defines the relationship between the HTML document and the external file, that in this case is a stylesheet
  - ***type*** attribute is optional in HTML5. In previous versions of (X)HTML was required.

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8" />
<title> External Style Sheet</title>

<link href="style1.css" rel="stylesheet" type="text/css"/>

</head>

<body>
</body>
</html>
```

```
/* style1.css */

p { font-weight:bold;
    color:#F00;}
```

Cascading Style Sheets (CSS)

- ***External Style Sheet***
  - alternative way to link html and a external *.css
    - *@import + <style>*

```
<!DOCTYPE HTML>
<html>
<head>
    <style>
        @import url("style1.css");
        ...

    </style>
    <meta charset="utf-8" />
    <title> External Style Sheet</title>
</head>

<body>
</body>
</html>
```

```
/* style1.css */

p { font-weight:bold;
    color:#F00;}
```

    - the two ways are supported by most browsers producing similar results, however:
      - Several authors argue that the <link> tag is the most efficient way to make connections to a external CSS

http://www.stevesouders.com/

Cascading Style Sheets (CSS)

- ***Embedded Style Sheet***
  - <style> … </style>
    - Attribute type="text/css" is optional in HTML5.
    - Less flexible than the external style

```
<!DOCTYPE HTML>
<html>
<head>
    <style>
        p { font-weight:bold;
            color:#F00;}
    </style>
    <meta charset="utf-8" />
    <title> Embedded Style Sheet</title>
</head>

<body>
</body>
</html>
```

- ***Inline Style***
  - attribute ***style***
    - Local formatting, should be avoided!

CSS Syntax

- CSS Rule
  - Each rule selects an element and declares/defines the respective formatting:

selector { property:value;}

  - Selector
    - It allows the identification of the specific HTML elements that the CSS rule applies to

```
body{ background-color:#FFF;}
```

  - Statement / set of statements
    - An assignment of a value to a property. A rule can contain multiple statements (declaration block).

```
body{ background-color:#FFF;}
```

property *value*

Task 2.1

Task 2.1

- Apply the following formatting to the html file implemented in the previous exercise :
  - selector: body;
  - property: background-color;
  - value: orange
- Through an external file *.css (main.css);
- Remove the content of the external file but keep the link in the html file

End

Task 2.1

CSS Selectors

- Element Selector
  - based on the tag name

```
<style>
    p{ border: 1px solid blue;}
    em{border: 2px solid red;}
    h2{ color:#066;}
</style>
```

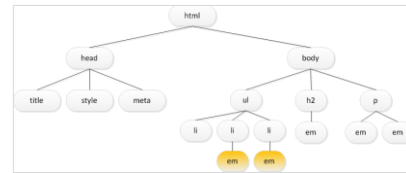
  - More than one element (separated by ,)

```
<style>
    h1,h2{ color: #900;}
</style>
```


CSS Selectors

- Contextual Selector
  - a white space to define the context

```
li em {color:orange;}
```



    - In this case only the **em** that are children of **li** are formated
  - *child selector* **(>)**
    - element_A > element_B
      - Selects all elements_B whose parent is a element_A
  - *adjacent sibling selector (+)*
    - element_A + element_B
      - Selects all elements_B that are placed immediately after element_A

http://www.w3schools.com/cssref/css_selectors.asp

CSS Selectors

- id Selector (#)
  - Selects the element with a specific id attribute

```
#slideshow1 {
            background-color: orange;
       }
```

- class Selector (.)
  - Selects elements with a specific class attribute

```
.image {
        background-color: orange;
    }
```

  - id and class selectors can be part of a contextual selector
  - id and class selectors can be applied to improve/develop an element selector

Task 2.2

Task 2.2

```
<p class="cl1">Small Exercise: <span>CSS Selectors</span></p>
```

```
<style>
    p {color:red;}
    span {color:gray;}
    p.span {color:green;}
    p  span {color:brown;}
    p  .cl1 {color:orange;}
    p.cl1 {color:orange;}
</style>
```

Small Exercise: CSS Selectors

Small Exercise: CSS Selectors

Small Exercise: CSS Selectors

Small Exercise: CSS Selectors

Small Exercise: CSS Selectors

Small Exercise: CSS Selectors

End

Task 2.2

CSS[units]

- *Units*
  - Relative units are based on the dimension of a specific element

Unit	Description
em	Relative to the font-size of the element (2em means 2 times the size of the current font)
ex	Relative to the x-height of the current font (rarely used)
ch	Relative to width of the "0" (zero)
rem	Relative to font-size of the root element
vw	Relative to 1% of the width of the viewport*
vh	Relative to 1% of the height of the viewport*
vmin	Relative to 1% of viewport's* smaller dimension
vmax	Relative to 1% of viewport's* larger dimension
%	

- special case: px

http://www.w3schools.com/cssref/css_units.asp

  - *Device Pixel* Ratio (ratio of the resolution in physical pixels to the resolution in CSS pixels)

CSS[properties]

- CSS has hundreds of properties
  - *font properties*

Property	Example	Comments
font-family	body{font-family:Arial, sans-serif;}	
font-size	h1{font-size: 1.5em;}	
font-weight	h1 {font-weight: bold;}	Values : normal*, bold, bolder, lighter, 100…900, inherit
font-style	h1 {font-style: italic;}	Values : normal*, italic, oblique, inherit
font-variant	h1 {font-variant: small-caps;}	Values: normal*, small-caps, inherit
font	h1 {font: italic bold inherit 1.5em Arial, sans-serif;}	{font: style weight variant size font-family}

CSS[properties]

- *color properties*

Property	Example	Comments
color	h1 {color: red;} h1 {color: #FF0000;} h1 {color: #F00;} h1 {color: rgb(255,0,0);}	color value (RGB / hexadecimal); color name (140 names)  http://www.w3schools.com/colors/colors_names.asp
background-color	h1 {background-color: #F00;}	color value (RGB / hexadecimal); color name (140 names)

CSS[properties]

- *Box model*
  - All HTML elements (inline or block) are interpreted by the browser as being contained in rectangular boxes, to which some properties can be applied.

CSS[properties]

- *Box model*

Property	Example	Comments
margin *margin-top* *margin-right* *margin-left* *margin-bottom*	p {margin:2em;}	margin area. Values: length measurement, percentage, auto, inherit.
padding *padding-top* *padding-right* *padding-left* *padding-bottom*	p {padding:2em;}	padding area. Values: length measurement, percentage, auto, inherit.
overflow	p{overflow:scroll;}	Values: *visible, hidden, scroll, auto*

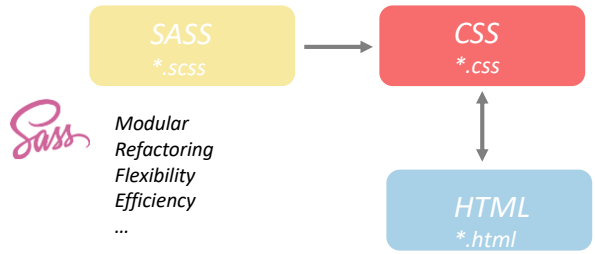
CSS[properties]

- *Box model*

Property	Example	Comments
border-style *border-top-style* *border-right-style* *border-left-style* *border-bottom-style*	*p {border-style: solid;}*	values: *none*, dotted, dashed, solid, double, inset, outset …*
border-width *border-top-width* *border-right-width* *border-left-width* *border-bottom-width*	*p {border-style: solid; border-width: 4px;}*	values: *length units, thin, medium*, thick, inherit*
border-color *border-top-color* *border-right-color* *border-left-color* *border-bottom-color*	*p {border-style: solid; border-width: 4px; border-color: red;}*	values: *color name/RGB value, transparent, inherit*
border-radius *border-top-left-radius:2em;* *border-top-right-radius:2em;* *border-bottom-right-radius:2em;* *border-bottom-left-radius:2em;*	*p {border-style: solid; border-width: 4px;} border-radius:2px;}*	values: *length units*
border *border-top* *border-right* *border-left* *border-bottom*	*p{border: 4px solid red; }*	(width, style, color)

Sass

- Sass is a pre-processor of CSS code



  - improves the refactoring, the code structure, …
  - it has some features **that are not available** in pure CSS, such as:
    - variables, inheritance, ….
  - *.scss files
    - *.css files are created based on *.scss
    - Important!! HTML **does not recognize** *.scss

Task 2.3

Task 2.3

- Open *Sublime Text 3*
  - Check for the following *Packages*.
    - *Sass*
    - *SCSS*
    - *SASS Buid*
    - *Emmet*
    - *SublimeOnSaveBuild*

      - *Preferences >> Package Control* (Ctrl + Shift + P). Install Packages.

  - Create a new file **test.scss** with the css code *body{background-color:red}*. Save it and build the respective **test.css**
    - *Tools >>Build (Ctrl + B)*.
  - Check if the test.css was created
  - Delete all the previous files.

End

Task 2.3

Sass

- Variables
  - allow variables declaration (CSS don't allow), all the variables start with **$**
    - Data types:
      - numbers
      - strings
      - colors
      - null
      - lists
      - maps

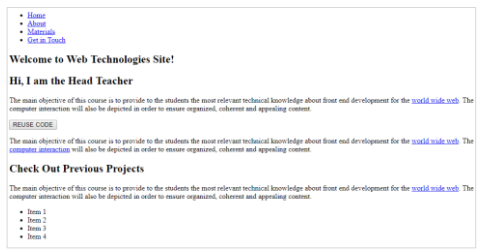
```
$margin-values: 1px 2px 3px 4px;
```

```
$map: (
    key: value,
    nextkey: nextvalue
);
```

  - The variables' names should be meaningful:
    - "header-***" ; "footer-***"; "section-***"; "****-color")

Task 2.4

Task 2.4



HTML
*.html

SASS
*.scss

CSS
*.css

HTML
*.html

Task 2.4

- Declare the variables related with ***colors***, assuming the following names:
  - $text-color: #222222;
  - $theme-color: #170a48;
  - $secondary-color: #f27731;
  - $ternary-color: #ccf962;
  - $menu-item-color: white;
  - $link-color: $secondary-color;
- Declare the ***font** variables*:
  - $text-font:Verdana, Arial, sans-serif;
  - $headline-font:Georgia, serif;
- Declare the variables that contain the ***dimensions*** of some sections:
  - $content-width:960px; $header-height:60px; $footer-height:60px;
- These variables should be declared before the first selector in *main.scss*. Identify the three groups of variables.
- Save and build the file main.css. Preview in the browser?

Solution

```
//    COLORS
$text-color:#222222;
$theme-color:#170a48;
$secondary-color:#f27731;
$ternary-color:#ccf962;
$menu-item-color:white;
$link-color:$secondary-color;

//    FONTS

$text-font:Verdana, Arial, sans-serif;
$headline-font:Georgia, serif;

//   SECTIONS

$content-width:960px;
$header-height:60px;
$footer-height:60px;
```

?

Preview in the Browser

Task 2.4

- The body assumes the values of variables **$text-color** and **$text-font** .
- All headings are defined with the **$headline-font**.
- The font of the paragraphs is the **$text-font**
- The color of links is the **$link-color.**

Solution

```
body {      color:$text-color;
            font-family: $text-font;}

h1, h2, h3, h4 {    font-family: $headline-font;}

p { font-family: $text-font;}

a { color:$link-color;}
```


Task 2.4

- The *id="header"* element:
  - background color ***$theme-color,***
  - height ***$header-height***,
  - padding of 10px
  - border radius of 5px

Solution:

```
#header {
  height:$header-height;
  background-color: $theme-color;
  padding:10px;
  border-radius:5px;
  }
```

End

Task 2.4

CSS Selectors

- *Pseudo-Class selector*
  - Based on the state of the element
  - Example <a> element:
    - *a:link*
      - link not visited
    - a:*visited*
      - link already visited
    - *a: hover*
      - the mouse is over the element
    - a:*active*
      - the element is being selected
    - …

```
/* unvisited link */
a:link {
    color: #FF0000;
}

/* visited link */
a:visited {
    color: #00FF00;
}
```

https://www.w3schools.com/css/css_pseudo_classes.asp

CSS[properties]

- display
  - The display property specifies the type of box used for display an HTML element

Property	Example	Comments
display	li {display:inline;}	values: none; inline; block; inline-block; ….
visibility	li {visibility:hidden;}	values: hidden, visible, ….

http://www.w3schools.com/cssref/pr_class_display.asp

CSS[properties]

- selector{*display:inline}*
  - sets the element as an inline element
  - useful for the implementation of horizontal menus (horizontal navigation bar)

```
<style>
    div {width:200px;height:30px;color:#FFF; font-weight:bold}

    #div1{ background-color:#F69140}
    #div2{ background-color:#EF415C;}
    #div3{ background-color:#8FCCBA;}

    div{display:inline;}
</style>
</head>

<body>
    <div id="div1">div1</div>
    <div id="div2">div2</div>
    <div id="div3">div3</div>
</body>
```

CSS[properties]

- selector{display: block}
  - Sets the element as a block element
  - Useful for the implementation of vertical menus / visibility of elements

```
<a href="#" class="class1">link 1</a>
<a href="#" class="class2">link 2</a>
```

```
a{display:block;
  width:10em;
  height:2em;
  color:white;}

a.class1{background-color:#F69140 }
a.class2{background-color:#EF415C }
```

link 1
link 2

- selector{display:inline-block}
  - The interior of the element is interpreted as a block-level element, however the element is defined as an inline-level element

```
a {display:inline-block;
  width:10em;
  height:2em;
  color:white;}

a.class1{background-color:#F69140 }
a.class2{background-color:#EF415C }
```

link 1 link 2

Sass: Nesting

- It allows the creation of a clear visual hierarchy, declaring the selectors directly inside the selectors that define the context
  - The extension of this selector dependence results in increasingly specific selectors
  - The **&** always refers to the parent selector when nesting

sass_1.scss

sass_1.scss

```
$primary-color: lightgray;
$width-element: 300px;

nav {
  ul {
    margin: 0;
    padding: 0;
    list-style: none;
  }

  li { display: inline-block; }

  a {
    display: block;
    padding: 6px 12px;
    text-decoration: none;
  }
}
```

sass_1.css

```
nav ul {
  margin: 0;
  padding: 0;
  list-style: none; }
nav li {
  display: inline-block; }
nav a {
  display: block;
  padding: 6px 12px;
  text-decoration: none; }
```

sass_1.css

Task 2.5

Task 2.5

- Links defined inside the *id="main-menu"* have the ***$menu-item-color,*** they aren't underlined. Their padding is 6px (*top/bottom)* and 8px (*left/right)*.
  - When there is a mouseover situation, these links are underlined with 2px white color.
  - List items defined inside the id="main-menu" should be interpreted as inline elements.
- *The width of id="main"* is ***$content-width***, while the left/right margins are set to *auto* (center this container).
- Assign to *footer* the ***$footer-height*** being the color the ***$ternary-color.*** This element has border radius of 5px.

Task 2.5 - Solution

```
#header {
        height:$header-height;
        background-color: $theme-color;
        padding:10px;
        border-radius:5px;

        #main-menu {
                li {display:inline;}

                a {
                        color:$menu-item-color;
                        text-decoration: none;
                        padding: 6px 8px;
                        &:hover{border-bottom: 2px white solid;}
                }
        }
}
```


Task 2.5 - Solution

```
#main {
        width:$content-width;
        margin-right: auto;
        margin-left: auto;
        }


#footer {
        background-color: $ternary-color;
        height:$footer-height;
        border-radius:5px;
        }
```

End

Task 2.5

Sass: @mixin / @include

- Set of CSS declarations to be reused
  - This block of code is based on the @mixin directive
    - @mixin must have a name
    - It is possible to define parameters in the @mixin
    - @mixin is used based on the @include directive
    - @mixin can be grouped into a partial (code organization)

sass_1.scss

sass_1.scss

```
@mixin paragraph($radius) {
    background-color: orange;
    color:white;
    width: 200px;
    height:100px;
    text-align:center;
    padding:10px;
    border-radius: $radius;
}

p {@include paragraph(10px); }
```

sass_1.css

```
p {
  background-color: orange;
  color: white;
  width: 200px;
  height: 100px;
  text-align: center;
  padding: 10px;
  border-radius: 10px; }
```

sass_1.css

Sass: @import (partials)

- Sass allows the import of small blocks of * .scss code (partials)
  - Partial is a * .scss file whose name starts with underscore.
    - The underscore (_) indicates that it is a partial (to be imported by another * .scss file and not converted directly to * .css)
  - Sass partials are used with the @import directive (only the partial name is specified)
  - It makes the code more modular (reset, variables, functions, ...) and consequently easier to maintain.

_reset.scss

_reset.scss

```scss
html,
body,
ul,
ol {
   margin: 0;
  padding: 0;
}
```

sass_1.scss

sass_1.scss

```scss
@import 'reset';

body {
  font: 100% Helvetica, sans-serif;
  background-color: #efefef;
}
```

sass_1.css

sass_1.css

```css
html,
body,
ul,
ol {
  margin: 0;
  padding: 0; }

body {
  font: 100% Helvetica, sans-serif;
  background-color: #efefef; }
```


Task 2.6

Task 2.6

- Create a *mixin* designated as ***codereuse***. It is defined with background color of orange and text color white.

- Create a new class selector ***code-button.*** This selector should include the previous mixin as well as a padding of 8px.
  - Preview the final result.

- Redefine the *@mixin* allowing the parameters definition (background color). Modify the *@include* to define the background color green.
  - Preview this change.
  - Return to the original value (orange).

Task 2.6 Solution

```
@mixin codereuse {
        background-color: orange;
        color:white;
}
```

```
<button class="code-button">REUSE CODE</button>
```

```
.code-button{
        @include codereuse;
        padding: 8px;
}
```

Task 2.6 Solution

```
@mixin codereuse ($colorbg){
        background-color: $colorbg;
        color:white;
}

.code-button{
        @include codereuse(orange);
        padding: 8px;
}
```


End

Task 2.6

CSS[Responsive Web Design]

- Responsive Web Design
  - Adaptation of the page layout to the browser window size.
    - Improve the user interaction independently of the device used to access the web content

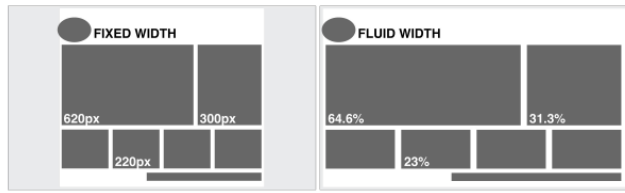
https://tech.co/top-5-reasons-use-responsive-website-design-2015-01

CSS[Responsive Web Design]

- Responsive Web Design
  - Three major elements:
    - Fluid layout
    - Flexible images
      - Images capable of suffering a scale effect according to the change of the layout.
    - CSS Media Queries
      - Method that allows the application of different styles according to the device in which the site is displayed.

CSS[Responsive Web Design]

- Fluid layout
  - The dimensions are specified based on %
    - Layouts for the main categories of devices (smartphones, tablets, desktops) are created and the fluid layout allows the adjustment to variations in the dimensions within each category.



https://www.sitepoint.com/premium/books/the-principles-of-beautiful-web-design-2nd-edition/online/ch01s11

CSS[Responsive Web Design]

- Flexible Images
  - The max-width property is used to set the maximum width of an element. This prevents the value of the width property from becoming larger than max-width.

```
img{max-width: 100%;}
```

  - Can also be applied to other types of media

```
img, object, video {max-width:100%;}
```

  - If max-width is defined as a %, the width of the element is directly indexed to the dimensions of the respective container
    - The element is automatically adjusted to the variation of the respective container width

CSS[Responsive Web Design]

- CSS media queries
  - The @media rule (media queries) is used to define different style rules for different media types/devices.
  - Media queries can be used to check different conditions, such as:
    - width and height of the viewport
    - width and height of the device
    - orientation (tablet/phone in landscape/portrait mode)
    - …

CSS[Responsive Web Design]

- *CSS media queries*
  - *media features*

Media Feature	Comments
width	viewport's width
height	viewport's height
device-width	device's width
device-height	device's height
orientation	portrait/landscape
aspect-ratio	width/height ratio
device-aspect-ratio	width/height ratio (device)
…	…

  - Some properties can be tested, using the prefixes:
    - ***min, max***

CSS[Responsive Web Design]

- *CSS media queries*
  - They can be directly implemented in the style sheet, e.g:

```
@media screen and (min-width:480px){

        /* CSS Code */

}

@media screen and (min-width:480px) and (orientation:landscape){

        /* CSS Code */

}
```

  - As an alternative, they can be embedded in html, incorporating the media queries in the media attribute of tag <link>

```
<link href="css_01.css" rel="stylesheet" />
<link href="css_02.css" rel="stylesheet"  media="screen and (min-width:480px)"/>
```

  - Usually the strategy applied to define media queries is based on the concept of mobile-first, i.e.:
    - It begins by defining the layouts for smaller devices; as the viewing space increases new styles are applied.

CSS[Responsive Web Design]

resolution

Pixel (CSS)
device width / device height

Common Smartphones values

name	phys. width	phys. height	CSS width	CSS height	pixel ratio
Apple iPhone X	1125	2436	375	812	3
Apple iPhone 6+, 6s+, 7+, 8+	1080	1920	414	736	3
Apple iPhone 7, iPhone 8	750	1334	375	667	2
Apple iPhone 6, 6s	750	1334	375	667	2
Apple iPhone 5	640	1136	320	568	2
Apple iPhone 4	640	960	320	480	2
Apple iPhone 3	320	480	320	480	1
Apple iPod Touch	640	1136	320	568	2
LG G5	1440	2560	360	640	4
LG G4	1440	2560	360	640	4

http://mydevice.io/devices/

CSS[Responsive Web Design]

- Viewport (visible area of a web page)
  - The viewport varies with the device, and of course will be smaller on a mobile phone than on a computer screen.
    - The browser must have the ability to adapt the visualization of the content to the dimensions of the viewport
  - *<meta name="viewport" … />*
    - Should be included in all HTML documents (responsive)
    - The viewport specification gives the browser instructions on how to control the page's dimensions and scaling.

CSS[Responsive Web Design]

- The viewport needs to be defined only once

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

  - content = "**width = device-width**, ..."
    - Sets that the width of the page has to follow the screen-width of the device (which will vary depending on the device).
  - content = "width = device-width, **initial-scale = 1**"
    - Sets the initial zoom level when the page is first loaded by the browser (default zoom level).

Sass: @media

- Sass allows the definition of CSS media queries
  - ***.scss** enables the nest of the CSS media queries
  - ***.css** code only enables the definition of the media queries at top level

*.scss

```
#main {
    width:$content-width;
    @media only screen and (max-width: 960px){
        width:auto;
        max-width:960px;
    }
```

*.css

```
@media only screen and (max-width: 960px) {
    #main {
      width: auto;
      max-width: 960px; } }
```


Task 2.7

Task 2.7

- Open the html file in the browser. Change resolution and check the respective behavior.
  - Conclusions?

- For the *id="main"* element, define a *media query* to be applied to a maximum of 960px wide. Apply the *@media directive,* defining the container's width as auto.

- Change the resolution in the browser.
  - Conclusions?

- Apply a new *media query* to the <*body*> element considering a maximum width of 960px. The font size should be 125%.

Task 2.7 Solution

```
#main {
        width:$content-width;
        margin-right: auto;
        margin-left: auto;

        @media only screen and (max-width: 960px){
          width:auto;
        }
...
}
```

Task 2.7 Solution

```
body {
        color:$text-color;
        font-family: $text-font;

        @media only screen and (max-width: 960px){
          font-size:125%;
        }
}
```


End

Task 2.7

Sass: Functions

- Two types of Functions: Built-in functions + Custom Functions
  - Built-in Functions:
    - Set of functions provided by SASS
      - List of main built-in functions: http://sass-lang.com/documentation/Sass/Script/Functions.html
    - Some of the most common:
      - *darken($color, amount)*
      - *lighten($color, amount)*
      - *transparentize($color; amount)*
      - *opacify($color;amount)*
      - …
  - Custom Functions

Task 2.8

Task 2.8

- Apply to *id="main"* a *buit-in function* ***darken*** with 15%. This darkened effect should be applied during mouseover.

Solution:

```
a {
    color:$link-color;
    &:hover{
            color:darken($link-color,15%);
    }
}
```


End

Task 2.8

CSS[properties]

- Float Positioning
  - The float property specifies whether or not an element should float

Property	Example	Comments
float	.par {float:left;}	values: left, right, none, inherit
clear	.par {clear:both;}	values: left, right, both, none, inherit

CSS[properties]

- *float property*
  - allows an element to be moved as far as possible to the left / right being surrounded by the content that follows it (right/left).
  - Positioning without float property (image + text)

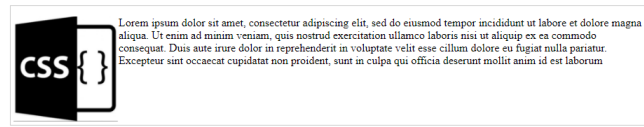
```
<img src="Imagens/css.png"/>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore
et dolore magna aliqua. Ut e.... </p>
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

CSS[properties]

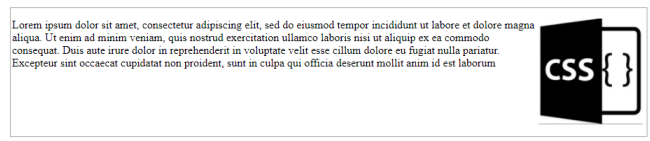
- seletor {float:**left**}

```
img{ width:150px;
     height:150px;
     float:left;}
```



- seletor {float:**right**}

```
img{  width:150px;
      height:150px;
      float:right;}
```



CSS[properties]

- *float positioning*
  - CSS allows to apply the float property to any element (block or inline)
  - All floated elements behave as block elements.
    - Whenever the float property is applied to text elements the respective width must be specified, otherwise all the available width is occupied (viewport's width).
    - The width value of the block to which the float positioning will be applied must be specified
    - A positioned element with float can not be positioned before its natural position in the source code.
      - Its position is conditioned by the previous block elements.
- clear
  - This property eliminates the float and returns to the normal position scheme
    - It ensures that the next element is displayed under the element to which the float property was applied.

CSS[properties]

- lists

Property	Example	Comments
list-style-type	ul {list-style-type:square;}	Values: none, disc*, circle, square, …
list-style-position	li {background-color: #666;} ul {list-style-position: outside;}	Values : inside, outside, inherit
list-style-image	ul {list-style-image: url(/image.png) list-style-position: outside;}	New bullets Values: url, none*, inherit

Sass: Functions

- @function
  - These functions allow multiple parameters as well as the specification of default values.
- @return
  - Returns the calculated value

```
@function simpleCalculation ($first_number, $second_number){
            @return $first_number + $second_number;
        }
```

- Function call:

```
.divMargin {
                margin: simpleCalculation(10px, 5px);
        }
```

Task 2.9

Task 2.9

- Create a function designated as *col-width* that has as parameters *(**$columns**:12, **$page-width**:100%, **$gap**:1%*) and returns the width of an individual column.

https://cssauthor.com/bootstrap-grid-system-psd-templates/

- **Solution:**

```
@function col-width($columns:12, $page-width:100%, $gap:1%){
        @return ($page-width - $gap*($columns - 1))/$columns;
}
```

Task 2.9

- This function should be called inside the *id="content"* and *id="sidebar" selectors*. It is called considering 8 columns.
  - The *id="content"* container has a width equal to 6 times the individual column width (returned by the *col-width function*) and it is positioned based on left floating.

- The *id="sidebar"* container has a width equal to 2 times the individual column width (returned by the *col-width function*) and it is positioned based on right floating.

- In the *id="footer"* the floating positioning should be removed.

- The <li> defined inside *id="sidebar"* don't have list bullets, have a padding of 16px (top/bottom) and 20px (left/right) and a *border bottom* dashed of 1px, color #ddd.

- Preview in the Browser!

Task 2.9 - Solution

```
#content {
        float:left;
        width:6*col-width(8);
}

#sidebar {
        float:right;
        width:2*col-width(8);
}

#footer {
        clear:both;
        background-color: $ternary-color;
        height:$footer-height;
        border-radius:5px;
}
```

Task 2.9 - Solution

```
#sidebar {
        float:right;
        width:2*col-width(8);

        li{
          list-style: none;
          padding:16px 20px;
          border-bottom: 1px dashed #ddd;
        }
}
```


End

Task 2.9

Outline

1. Structure Layer
   - *Hyper Text Markup Language (HTML)*
2. Presentation Layer
   - *Cascading Style Sheets (CSS)* ✓
3. Behaviour Layer
   - *JavaScript*

JavaScript

JavaScript

- Main Features
  - Server Side (Node.js) / Client Side
  - Client Side:
    - Script language directly interpreted by web browsers (on the fly)
    - Code may be directly included in the HTML
    - Reduced learning curve
    - Improvement of user interaction
    - Event management
    - Browser Control
    - …

JavaScript

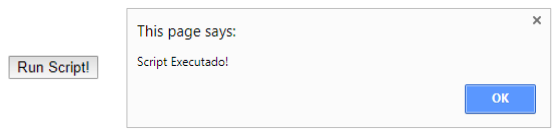
- Embedded Script
  - <script> ... </ script>
    - The script can be placed on the head or in the body;
    - To maximize performance, the script should be placed at the end of the body as it doesn't influence the download of the remaining HTML elements
    - The script can be executed when the page downloads or triggered by an event

```
<h1>New Date</h1>
<script>
    var newDate = new Date();
    document.write(newDate.getDate());
    document.write("/");
    document.write(newDate.getMonth()+1);
    document.write("/");
    document.write(newDate.getFullYear());
</script>
```

JavaScript

- Embedded Script
  - ***<script> … </script>***
    - triggered by an event, e.g. *onclick*

```
<button onclick="scriptFunction()">Run Script!</button>

<script>
    function scriptFunction(){
        alert('Script Executado!');
    }
</script>
```



JavaScript

- External Script (*.js)
  - ***<script src=" *.js "> … </script>***

*The function **scriptFunction()** declared in the file **external.js** is executed*

```
<body>

    <button onclick="scriptFunction()">Run Script!</button>

    <script src="external.js"></script>

</body>
```

```
function scriptFunction(){
        alert('Script Executado!');
  }
```

external.js

JavaScript

- Scripts
  - Embedded in HTML
  - External files (*.js)
  - Execution:
    - After download
    - Triggered by an event

Task 3.1

Task 3.1

- Open Sublime Text 3 or Brackets
- Open exercise2.html
- Create a new file app.js
- Copy the content date.txt to app.js
- Create the connection to app.js in the exercise2.html
- Preview in the browser
- Keep app.js but delete/comment its content

End

Task 3.1

Javascript

- Syntax
  - It is case sensitive.
  - // Comment symbol
  - / * Comment for multiple lines * /
  - A script consists of a set of statements/expressions:

```
<h1>New Date</h1>
<script>
    var newDate = new Date();
    document.write(newDate.getDate());
    document.write("/");
    document.write(newDate.getMonth()+1);
    document.write("/");
    document.write(newDate.getFullYear());
</script>
```


Javascript

- Declaration of variables
  - loosely typed, it is not necessary to define the type of a variable since it is automatically assumed according to the respective declaration (initial value);
  - Keyword: **var**
    - Names must begin with a letter or underscore "_"
    - Should not contain spaces or special characters (., / \ * + = ...)

```
var x=10;       //numeric
var x="ten"     //string
var x;
```

  - global variables declared outside any function are visible everywhere (global execution context)

Javascript

- JavaScript allows the declaration of variables based on two major types:
  - Primitive Type
    - The variable contains directly the value assigned to it
  - Reference Types
    - Objects
      - The variable doesn't actually contain the value, it contains the reference (pointer) to the object's location in memory.
  - However, in JS, some primitive types can be treated as reference types (*Primitive Wrapper Types)*
    - They have properties/methods (except null and undefined types)
    - More consistent language

Javascript Objects

- **Objects** are the central concept of JS
  - An object is an unordered list of **properties**, with the respective values. Whenever the value is a function, a method is created.
  - Create an object:
    - Literal Notation

```
<script>
    var hotel = {

        name: 'Coimbra',
        rooms: 20,
        booked: 15,
        gym: true,
        roomTypes:['single', 'double','suite'],

        checkAvailability: function () {
            return this.rooms - this.booked;
        }

    }
</script>
```

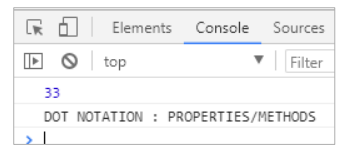
    - *Constructor Object ()*
    - Constructor Function ≠ Object()

```
function Hotel(name, rooms, booked) {
    this.name = name;
    this.rooms = rooms;
    this.booked = booked;

    this.checkAvailability = function () {
        return this.rooms - this.booked;
    }
}
var coimbraHotel = new Hotel('Coimbra', 20, 15);
```

(.)dot notation

- Access to properties
  - *objectName.**propertyName***
- Access to methods
  - *objectName.**methodName( )***

```
<script>
    var txt="dot notation : properties/methods";

    console.log(txt.length);

    console.log(txt.toUpperCase());
</script>
```



Task 3.2

Task 3.2

- Copy the following files:
  - exercise2.html
  - style.css
  - back.png
  - dice-1.png
  - dice-2.png
  - dice-3.png
  - dice-4.png
  - dice-5.png
  - dice-6.png

Task 3.2

- Check índex.html in the browser

https://www.udemy.com/

Task 3.2

- The objetive is to implement a simple game:
  - Rules:
    - The game has 2 players, playing in rounds
    - In each turn, a player rolls a dice as many times as he wishes. Each result gets added to his ROUND score
      - **BUT**, if the player rolls a 1, all his ROUND score gets lost. After that, it's the next player's turn
    - The player can choose to 'Hold', which means that his ROUND score gets added to his GLOBAL score. After that, it's the next player's turn
    - The first player to reach 100 points on GLOBAL score wins the game

Task 3.2

- In app.js, declare the global variables:
  - *scores*
  - *roundScore*
  - *activePlayer*
  - *gamePlaying*

  - Solution:

```
var scores, roundScore, activePlayer, gamePlaying;
```

End

Task 3.2

Javascript

- *Array*
  - It is a Reference Type (Object), has properties / methods
  - It holds a set of related values
  - It doesn't have to be declared with a specific dimension
  - It may include different data types
  - Index starts with value 0
    - Literal Notation:

```
var values=[20, "array Literal", 5, true];

document.write(values[0]+"<br/>");
document.write(values[1]+"<br/>");
document.write(values[2]+"<br/>");
document.write(values[3]+"<br/>");
```

```
20
array Literal
5
true
```

    - Based on a Constructor:

```
var values=new Array(20, "array Literal", 5, true);
```

Javascript - Function

- Set of statements to perform a specific task:
  - In JS a Function is also an object
  - Function's Declaration (Literal notation):

```
function  firstFunction(){
          document.write("hello");
          }
```

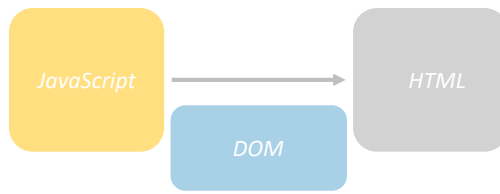
  - Function's call:

```
firstFunction ();
```

  - The browser runs over the entire script before the execution of each statement

HTML DOM

- Document Object Model (HTML DOM)
  - W3C standard
  - HTML DOM provides a set of methods / properties (API) to allow the access through a programming language (Javascript, PHP, Ruby, …) to HTML elements
  - Accessing HTML elements by name or attributes and perform several operations:
    - Add, modify, delete elements and content



Document Object Model

*The **HTML DOM** is a standard for how to get, change, add, or delete HTML elements.*

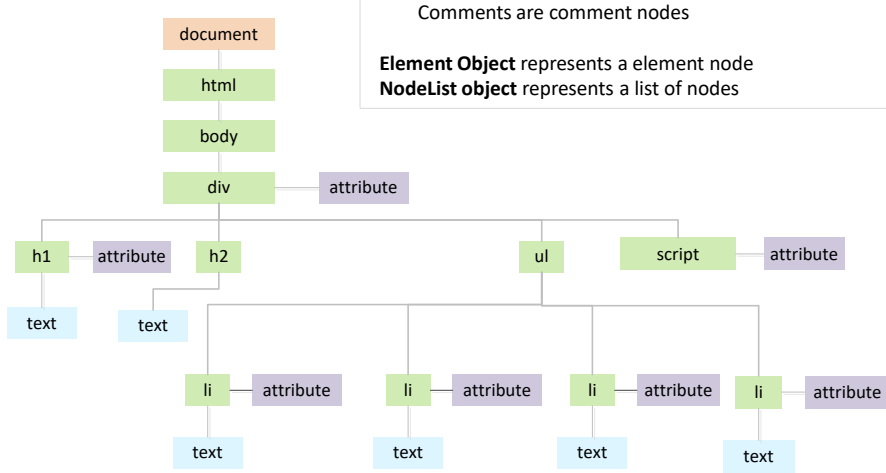
http://www.w3schools.com/js/js_htmldom.asp

Document Object Model (DOM)

- DOM Tree

HTML DOM considers everything as a node (node)
- The document (root element) is a document node
- Any element is an element node
- The text contained in the HTML elements is a text node
- Any HTML attribute is an attribute node
- Comments are comment nodes

Element Object represents a element node
NodeList object represents a list of nodes



HTML DOM

- *Document Object*
  - When an HTML document is loaded into a web browser, it becomes a **document object**.
  - Methods:

Method	Description
getElementById()	Returns the element that has an ID attribute with the a value
getElementsByTagName()	Returns a node list (collection/array of nodes) containing all elements with a specified tag name
getElementsByClassName()	Returns a node list containing all elements with a specified class
appendChild()	Adds a new child node to a specified node
removeChild()	Removes a child node
replaceChild()	Replaces a child node
insertBefore()	Inserts a new child node before a specified child node
createAttribute()	Creates an Attribute node
createElement()	Creates an Element node
createTextNode()	Creates a Text node
getAttribute()	Returns the specified attribute value
setAttribute()	Sets or changes the specified attribute, to the specified value

http://www.w3schools.com/htmldom/dom_methods.asp

HTML DOM

- Element Object
  - In the HTML DOM, the **Element object** represents an HTML element, like p, div, a, table or any other HTML element.
  - Some Properties:

Properties	Comments
innerHTML	access / change the HTML content of an element
textContent	Access / change text content
className	Returns / changes the value of the class attribute of an element

HTML DOM – Access Nodes

- *document.getElementById()*
  - The fastest and the most effective way to access a single element
  - Returns the element object whose element has the id specified in the DOM query

```
<ul>
    <li id="one" class="fruit">apples</li>
    <li id="two" class="fruit">oranges</li>
    <li id="three" class="fruit">grapes</li>
</ul>

<script>
    var el = document.getElementById('one');
    el.className = 'promo';
</script>
```

```
<style>
    .promo {color:white;
            background-color:lightgreen;}
</style>
```

apples
- oranges
- grapes

- *document.querySelector()*
  - based on CSS selectors
  - returns the first element that is identified by the CSS selector

HTML DOM

- *Style Object*
  - The Style object represents an individual style statement, allow the implementation of the CSS properties

Style Object Properties

The "CSS" column indicates in which CSS version the property is defined (CSS1, CSS2, or CSS3).

Property	Description	CSS
alignContent	Sets or returns the alignment between the lines inside a flexible container when the items do not use all available space	3
alignItems	Sets or returns the alignment for items inside a flexible container	3
alignSelf	Sets or returns the alignment for selected items inside a flexible container	3
animation	A shorthand property for all the animation properties below, except the animationPlayState property	3
animationDelay	Sets or returns when the animation will start	3
animationDirection	Sets or returns whether or not the animation should play in reverse on alternate cycles	3
animationDuration	Sets or returns how many seconds or milliseconds an animation takes to complete one cycle	3
animationFillMode	Sets or returns what values are applied by the animation outside the time it is executing	3
animationIterationCount	Sets or returns the number of times an animation should be played	3
animationName	Sets or returns a name for the @keyframes animation	3
animationTimingFunction	Sets or returns the speed curve of the animation	3
animationPlayState	Sets or returns whether the animation is running or paused	3
background	Sets or returns all the background properties in one declaration	1

https://www.w3schools.com/jsref/dom_obj_style.asp

CSS[properties]

- display
  - The display property specifies the type of box used for display an HTML element

Property	Example	Comments
display	li {display:inline;}	values: none; inline; block; inline-block; ….
visibility	li {visibility:hidden;}	values: hidden, visible, ….

http://www.w3schools.com/cssref/pr_class_display.asp

Task 3.3

Task 3.2

- Declare a *init* function with no parameters
  - This function must:
    - set the global variables to zero:
      - scores must hold the scores of the two players;
      - gamePlaying must be set to true;
    - set to zero all the values of the UI;
    - set to “player 1” and “player 2” the names of the players;
  - Call this function right after the global variable declaration
    - test the function

Task 3.2 Solution

```
init();

function init(){
    scores=[0,0];
    roundScore=0;
    activPlayer=0;
    gamePlaying=true;

            // Initial Configuration

    document.querySelector('#score-0').textContent='0';
    document.querySelector('#score-1').textContent='0';

    document.querySelector('#current-0').textContent='0';
    document.querySelector('#current-1').textContent='0';

    document.querySelector('#name-0').textContent='Player 1';
    document.querySelector('#name-1').textContent='Player 2';

}
```


Task 3.2

- Still in the init() function, hide the dice
- Solution:

```
    document.querySelector('#name-0').textContent='Player 1';
    document.querySelector('#name-1').textContent='Player 2';

    document.querySelector('.dice').style.display='none';

}
```

End

Task 3.3

JavaScript: Functions

- **Anonymous Function**
  - The declaration of a function can be embedded in an expression
  - The function name is not specified after the keyword function (**anonymous function**)
  - It is treated as an expression, i.e. the function is interpreted only after processing the expression where it is integrated

```
<script>
    var hotel = {

        name: 'Coimbra',
        rooms: 20,
        booked: 15,
        gym: true,
        roomTypes:['single', 'double','suite'],

        checkAvailability: function () {
            return this.rooms - this.booked;
        }

    }
</script>
```

Javascript

- Events
  - Actions that can be detected by JavaScript and originate a specific behavior: call a function.
    - The function is executed only after the occurrence of the event.
  - Examples:
    - Mouse click
    - Load a page
    - Image mouse over
    - …

Javascript

- Three distinct ways to declare an event:
  - *HTML Event Handlers*
    - Directly defined in the HTML element.
    - Considered bad practice since this approach associates directly the HTML and JavaScript. There should be a clear separation in the application of the two technologies

```
<a onclick="hide()"> … </a>
```

  - *DOM Event Handlers*
    - Considered better than the previous option since they allow a clear separation between Javascript and HTML
    -

```
element.onevent=functionName;
```

  - *DOM Event Listeners*
    - Different syntax
    - The most powerful way to handle events, allow the association of multiple functions to the same event, and many events to the same element

```
element.addEventListener('event',functionName,[Boolean]);
```

Task 3.4

Task 3.4

- An event (click) must be associated to the button with class “btn-roll“
  - All the operations inside this function (anonymous function) must depend on the variable gamePlaying (true)

- Solution:

```
document.querySelector('.btn-roll').addEventListener('click',function(){
    if (gamePlaying)
    {


    };

});
```

End

Task 3.4

Global Javascript Objects

- Math
  - Mathematical operations, this object provides advanced mathematical functions (trigonometry, statistics, etc.)
  - The Math properties / methods are called without explicitly creating an object of type Math (example: Math.round (x))

Propriedade	Descrição
Math.PI	returns 3.14159265359

Método	Descrição
Math.round()	rounds the number to nearest integer
Math.sqrt()	square root
Math.ceil()	rounds the number to next integer
Math.floor()	rounds the number to the previous integer
Math.random()	generates a random number between 0 and 1

Task 3.5

Task 3.4

- The local variable “dice” should hold a random number between 1 and 6 (Math.floor; Math.random)

- **Solution:**

```
document.querySelector('.btn-roll').addEventListener('click',function(){
    if (gamePlaying)
    {
            var dice=Math.floor(Math.random()*6 + 1);
            console.log(dice);

    };
});
```

- Check if it is working.

End

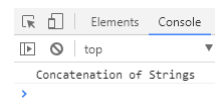
Task 3.5

Operators

- Strings
  - Concatenation (+)
    - Very frequently applied operator

```
<script>
  var strC = "Concatenation " + "of" + " Strings";
  console.log(strC);
</script>
```

Concatenation



CSS[properties]

- display
  - The display property specifies the type of box used for display an HTML element

Property	Example	Comments
display	li {display:inline;}	values: none; inline; block; inline-block; ….
visibility	li {visibility:hidden;}	values: hidden, visible, ….

http://www.w3schools.com/cssref/pr_class_display.asp

Task 3.6

Task 3.6

- Display the png image that corresponds to the random value (dice-number.png)
  - Remember:
    - dice-1.png
    - dice-2.png
    - dice-3.png
    - dice-4.png
    - dice-5.png
    - dice-6.png

Solution:

```
document.querySelector('.btn-roll').addEventListener('click',function(){
    if (gamePlaying)
    {
            var dice=Math.floor(Math.random()*6 + 1);


            var domDice=document.querySelector('.dice');
            domDice.src='dice-' + dice + '.png';
            domDice.style.display='block';

    };

});
```


End

Task 3.6

Operators

Operator	Description	Example	Result of x	Result of y
+	Addition	x=y+2	7	5
-	Subtraction	x=y-2	3	5
*	Multiplication	x=y*2	10	5
/	Division	x=y/2	2.5	5
%	Modulus (division remainder)	x=y%2	1	5
++	Increment	x=++y	6	6
		x=y++	5	6
--	Decrement	x=--y	4	4
		x=y--	5	4

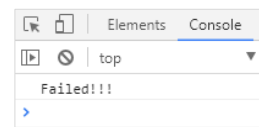
Operator	Example	Same As	Result
=	x=y		x=5
+=	x+=y	x=x+y	x=15
-=	x-=y	x=x-y	x=5
*=	x*=y	x=x*y	x=50
/=	x/=y	x=x/y	x=2
%=	x%=y	x=x%y	x=0

Operator	Description	Comparing	Returns
==	is equal to	x==8	*false*
		x==5	*true*
===	is exactly equal to (value and type)	x==="5"	*false*
		x===5	*true*
!=	is not equal	x!=8	*true*
!==	is not equal (neither value nor type)	x!=="5"	*true*
		x!==5	*false*
>	is greater than	x>8	*false*
<	is less than	x<8	*true*
>=	is greater than or equal to	x>=8	*false*
<=	is less than or equal to	x<=8	*true*

Selection

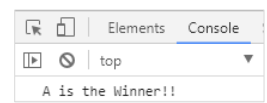
- if …else

```
<script>
    var threshold=50;
    var grade=40;

    if (grade>=threshold)
        {console.log("Aproved!!");}
    else
        {console.log("Failed!!");}
</script>
```



- Chained conditions

```
<script>
    var scoreA=60;
    var scoreB=50;

    if (scoreA>scoreB)
        {console.log("A is the Winner!!");}
    else if (scoreA<scoreB)
        {console.log("B is the Winner!!");}
    else
        {console.log("Draw !!")}
</script>
```



Task 3.7

Task 3.7

- If the dice is not 1:
  - The current field of the active player (id="current-0"/id="current-1") should be updated with the sum of the values

Solution:

```
        ...
        domDice.style.display='block';

        if (dice!== 1)
        {
            roundScore += dice;
            document.querySelector('#current-'+ activePlayer).textContent=roundScore;
        }
        else{}
};
```

Task 3.7

- Otherwise:
  - The player must be switched

Solution:

```
    document.querySelector('.btn-roll').addEventListener('click',function(){
    if (gamePlaying)
    {
            var dice=Math.floor(Math.random()*6 + 1);

            var domDice=document.querySelector('.dice');
            domDice.src='dice-' + dice + '.png';
            domDice.style.display='block';

            if (dice!== 1)
            {
                roundScore += dice;
                document.querySelector('#current-'+ activePlayer).textContent=roundScore;
            }
            else{
                nextPlayer();
            }
    };
});
```


End

Task 3.7

Task 3.8

Task 3.8

- An event (click) must be associated to the button with class “btn-hold“
  - All the operations inside this function (anonymous function) must depend on the variable gamePlaying (true)

Solution:

```
document.querySelector('.btn-hold').addEventListener('click', function(){
    if (gamePlaying)
    {


    };
});
```

Task 3.8

- Update the global score with the value stored in current score (considering the respective activePlayer)

Solution:

```
document.querySelector('.btn-hold').addEventListener('click', function(){
    if (gamePlaying)
    {
        scores[activePlayer] += roundScore;
        document.querySelector('#score-'+ activePlayer).textContent=scores[activePlayer];

    };
});
```


End

Task 3.8

JS *element.classList*

- The JavaScript **element.classList object** exposes one property and a number of methods that allow to work with CSS classes that are assigned to an HTML element.

Property	Description
length	Returns the number of classes in the list. This property is read-only

Methods

Method	Description
add(*class1, class2, ...*)	Adds one or more class names to an element. If the specified class already exist, the class will not be added
contains(*class*)	Returns a Boolean value, indicating whether an element has the specified class name. Possible values: • true - the element contains the specified class name • false - the element does not contain the specified class name
item(*index*)	Returns the class name with a specified index number from an element. Index starts at 0. Returns *null* if the index is out of range
remove(*class1, class2, ...*)	Removes one or more class names from an element. **Note:** Removing a class that does not exist, does NOT throw an error
toggle(*class*, true|false)	Toggles between a class name for an element. The first parameter removes the specified class from an element, and returns false. If the class does not exist, it is added to the element, and the return value is true. The optional second parameter is a Boolean value that forces the class to be added or removed, regardless of whether or not it already existed. For example:

https://www.w3schools.com/jsref/prop_element_classlist.asp

Task 3.9

Task 3.9

- Check if the player is the winner of the game
  - If so:
    - replace the identification of the player by ‘winner!’
    - hide the dice
    - add the class ‘winner’
    - remove the class ‘active’
    - set the variable gamePlaying to ‘false’
  - Otherwise:
    - switch player

Task 3.9 - Solution

```
document.querySelector('.btn-hold').addEventListener('click', function(){
    if (gamePlaying)
    {
        scores[activePlayer] += roundScore;
        document.querySelector('#score-'+ activePlayer).textContent=scores[activePlayer];

        if (scores[activePlayer]>=10)
        {
            document.querySelector('#name-' + activePlayer).textContent='Winner!!!';

            document.querySelector('.dice').style.display="none";
            document.querySelector('.player-' + activePlayer +'-panel').classList.add('winner');
            document.querySelector('.player-' + activePlayer +'-panel').classList.remove('active');

            gamePlaying=false;
        }
        else
        { nextPlayer();}
    };
});
```

End

Task 3.9

Task 3.10

Task 3.10

- switch player:
  - change the activePlayer
  - set the current score to zero
  - change the class of the panels (if “not active” change to “active”; if “active” change to “not active”)
  - hide the dice.

Solution:

```
function nextPlayer(){

    if (activePlayer===1)
        activePlayer=0;
    else
        activePlayer=1;

    roundScore=0;

    document.querySelector('#current-0').textContent='0';
    document.querySelector('#current-1').textContent='0';

    document.querySelector('.player-0-panel').classList.toggle('active');
    document.querySelector('.player-1-panel').classList.toggle('active');

    document.querySelector('.dice').style.display='none';
}
```


Task 3.10

- An event (click) must be associated to the button with class “btn-new“
  - This buttton creates a new game, so the initial configuration must be applied (init() function)

Solution:

```
document.querySelector('.btn-new').addEventListener('click', init);
```

Task 3.10

- update init()
  - remove the class winner
  - make the player 1 the active player in the beginning of the game

Solution:

```
...
    document.querySelector('#name-0').textContent='Player 1';
    document.querySelector('#name-1').textContent='Player 2';

    document.querySelector('.dice').style.display='none';

    document.querySelector('.player-0-panel').classList.add('active');
    document.querySelector('.player-0-panel').classList.remove('winner');

    document.querySelector('.player-1-panel').classList.remove('winner');
    document.querySelector('.player-1-panel').classList.remove('active');
}
```


End

Task 3.10

Outline

1. Structure Layer
   - *Hyper Text Markup Language (HTML)*
2. Presentation Layer
   - *Cascading Style Sheets (CSS)*
3. Behaviour Layer
   - *JavaScript* ✓
   - *jQuery*

jQuery
write less, do more.

jQuery

- jQuery is a framework JavaScript
  - Created to facilitate the use of Javascript
    - There are other JavaScript frameworks (Angular, React, ...)
  - It allows the:
    - selection and handling of HTML (DOM) elements
    - selection and manipulation of elements CSS
    - creation of events in HTML
    - creation of effects and animations
    - improvement of the portability between browsers
    - ...

jQuery

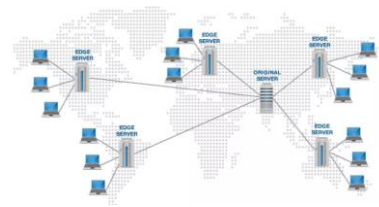
- jQuery is included in a HTML file based on the <script> tag and can be done in two different ways:
  - Directly through the download of the most recent version of jQuery:

```
<script src="jquery-3.2.1.js"></script>
```

```
<script>
    /* jQuery Code */
</script>
```

jQuery

- From a *Content Delivery Network (CDN)*



- Potentially faster but has the disadvantage of possible unavailability of the file

```
<script src="//ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
```

- which can be circumvented through the following code:

```
<script src="//ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

<script>
        window.jQuery|| document.write('<script src=" jquery-3.2.1.js"></script>')
</script>
```


jQuery

- jQuery syntax

$ (Selector) .method (parameters)

- $
  - It indicates that it is a jQuery command
- Selector
  - It is based (mainly) on CSS selectors
  - Identifies the HTML element(s) modified by the method defined on the right (after the period)
  - Eases selection since they do not need to use the methods provided by the DOM (document.getElementById(), getElementsByTagName(), ...)
- Method (parameters)
  - Sets the operation to carry out in the elements.

jQuery *[selection]*

- Selection of HTML elements
  - The use of CSS based selectors facilitates the selection of HTML elements
  - It allows to apply the same action to a set of elements avoiding the use of cycles

```
<div id="d1">
    <p>JavaScript</p>
    <p class="pgh">script language</p>
</div>
<div id="d2">
    <p>JavaScript</p>
    <p class="pgh">script language</p>
</div>

<script src="jquery-1.12.1.js"></script>
<script>
    $(document).ready(function(){
        $('p').html('jQuery');
        $('.pgh').html('write less do more');
    });
</script>
```

jQuery

write less do more

jQuery

write less do more

jQuery *[selection]*

- In addition to CSS selectors jQuery Filters can be applied
  - *:first*
    - allows the selection of the first element of a sample group (e.g. *$('p:first'))*
  - *:last*
  - *:even*
    - *index (0, 2, 4, …)*
  - *:odd*
    - *index (1, 3, 5, …)*
  - *:has*
    - allows the selection of an element that contains another element (e.g. *$('li:has(a);)*
  - …

jQuery

- Every time that multiple selections of the same HTML element have to be performed, it is more efficient to declare a variable that stores that selection

```
var $varname = $('div');

var $varname = $('p');
```

  - The use of a variable with the desired selection makes the interpretation by the browser faster when compared to the multiple selection of the same HTML element.
    - By convention variable names begin with the $ symbol
  - jQuery methods can be applied to that variable (jQuery Objet) in the same way as to the initial HTML element selection.

jQuery

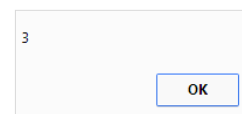
- jQuery Object

```
$('div');

var  $nomeVar = $('div');

var  $nomeVar=$('p').next()
```

  - It has properties and methods
    - length
      - the length property returns the number of elements of a jQuery object

```
<ul>
<li>Coffee</li>
<li>Milk</li>
<li>Soda</li>
</ul>
```

```
alert($("li").length);
```



jQuery [method(parameters)]

- $(selector).method(parameters);
  - Methods:
    - Element Selection
    - Change Content
    - Change Structure
    - Change Attributes
    - Effects
    - Animation
    - ...

jQuery [method(parameters)]

- Chaining
  - JQuery allows the chaining of multiple methods
  - If one method does not work, the other methods that are linked with it will also not work

$('#ex1').css ('font-size','200%').css('border','2px solid #FF0');

```
$('#ex1')
   .hide()
   .fadeIn(1400)
   .css('border','2px solid #FF0');
```

jQuery [selection]

- The access to an HTML element can also be made through the *jQuery DOM Traversing Methods*:
  - *next()*
    - provides access to the next sibling of a given element

```
$( "li.third-item" ).next().css( "background-color", "red" );
```

```
<ul>
 <li>list item 1</li>
 <li>list item 2</li>
 <li class="third-item">list item 3</li>
 <li>list item 4</li>
 <li>list item 5</li>
</ul>
```

Ref.

  - *prev ()*
    - provides access to the preceding sibling of a given element
  - ...

jQuery [method(parameters)]

- Change Content
  - html ()
    - Without parameters allows to obtain the HTML content inside the element specified in selector;
    - With parameters allows to change the HTML inside the element specified in the selector

```
<div id="d1">
    <p>JavaScript</p>
    <p class="pgh">script language</p>
</div>
<div id="d2">
    <p>JavaScript</p>
    <p class="pgh">script language</p>
</div>

<script src="jquery-1.12.1.js"></script>
<script>
    $(document).ready(function(){
        $('p').html('jQuery');
        $('.pgh').html('write less do more');
    });
</script>
```

jQuery

write less do more

jQuery

write less do more

  - text()
    - Similar to html() but restricted to text

jQuery [method(parameters)]

- Change Attributes
  - attr ();
  - addClass();
    - Add the class attribute
  - removeClass();
    - Remove the class attribute
  - css ()
    - Sets Single/multiple properties(s)

```
$( "li.third-item" ).next().css( "background-color", "red" );
```

- Change Structure
  - append(); remove()

jQuery [method(parameters)]

- Effects/Animations
  - show() / hide() / toggle()
    - Visibility control
  - slideDown() / slideUp() / slideToggle()
    - Slider effect
  - animate()
    - Definition/parametrization of animations
      - animate (CSS Property, duration (ms), function)
        - CSS property
        - Duration of animation (ms)
        - Function to run after the completed animation

```
$( "#book" ).animate({opacity: 0.25},5000, function() {
                        // Animation complete.
                });
```

jQuery [method(parameters)]

- Event
  - Action that can be detected and triggers the execution of a specific processing (function)
    - working with events, three main steps:
      - HTML element selection
      - Event definition
      - The set of actions that occur when the specified event is triggered (function)

jQuery [method(parameters)]

- Events
  - Three steps:
    - HTML elements selection
      - ***$('#menu')***
    - Event definition
      - ***$('#menu').mouseover();***
    - Anonymous function
      - Function is executed only after the ocurrence of the respective event

```
$('#menu').mouseover(function( ){
        $('submenu').show();
});
```

jQuery [method(parameters)]

Category: Mouse Events

.click()
Bind an event handler to the "click" JavaScript event, or trigger that event on an element.

.dblclick()
Bind an event handler to the "dblclick" JavaScript event, or trigger that event on an element.

.focusout() Also in: Events > Keyboard Events
Bind an event handler to the "focusout" JavaScript event.

.hover()
Bind one or two handlers to the matched elements, to be executed when the mouse pointer enters and leaves the elements.

.mousedown()
Bind an event handler to the "mousedown" JavaScript event, or trigger that event on an element.

.mouseenter()
Bind an event handler to be fired when the mouse enters an element, or trigger that handler on an element.

http://api.jquery.com/category/events/mouse-events/

jQuery

- *$(document).ready(function(){…})*
  - Ensures that the jQuery code is executed only after the complete download of the HTML document
  - This function should always be included in the jQuery script as it avoids any potential problem caused by the script location
  - The script with jQuery code (if included in the head) must be defined after the CSS

```
$(document).ready(function(){
    // jQuery Code
});
```

JavaScript

- *Important to the following exercise:*
  - variable declaration
    - *var*

```
var x=10;  // numeric
var x="ten"; // string
var x;
```

  - *Selection statetement*
    - *if (condição){ …} else{…}*
  - *Function declaration*

```
function functionName(var1,var2)
{
     // code
}
```


CSS[properties]

- *position*
  - one of the most important CSS properties

Property	Example	Comments
position	div{position:absolute}	Values: static*, relative, absolute, fixed, inherit
z-index	div {z-index:5}	highest integer value (bring to front)

CSS[properties]

- Position
  - Static (default)
    - The elements are displayed in the order they are defined (top-> bottom, left-> right);
    - Block elements are arranged sequentially vertically (from the top) and fill the horizontal space available in the browser window or defined by its container
    - Inline elements are arranged aligned to fill the block elements.

```
<style>
        div {
            width: 200px;
            height: 50px;
        }

        .c1 {
            background-color: lightgray;
            color: white;
        }

        .c2 {
            background-color: lightcoral;
            color: white;
        }
    </style>
</head>
<body>
    <div class="c1">STATIC (default)</div>
    <div class="c2">STATIC (default)</div>
</body>
```

STATIC (default)

STATIC (default)

CSS[properties]

- Positioning
  - The relative and absolute positions are based on displacement from references: top, left, right, bottom
  - Thus, in these two cases the property position requires the specification of additional properties:

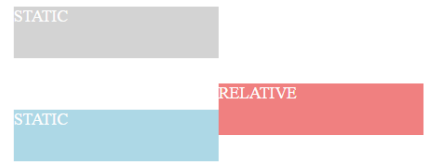
Property	Example	Comments
top ↓ right ← bottom ↑ left →	div{top:30px;left:50px;}	Values: length measurement, % , auto, inherit

CSS[properties]

- Positioning
  - Relative

```
<div class="c1">STATIC </div>
<div class="c2">RELATIVE </div>
<div class="c3">STATIC </div>
```

```
<style>
    div {width:200px;height:50px;
    }
    .c1 {background-color:lightgray;
         color:white;
    }
    .c2 {background-color:lightcoral;
         color:white;
         position:relative;
         left:200px;
         top:25px;
    }
       .c3 {background-color:lightblue;
         color:white;
    }
</style>
```



  - The reference for displacement is the original position of the displaced element

CSS[properties]

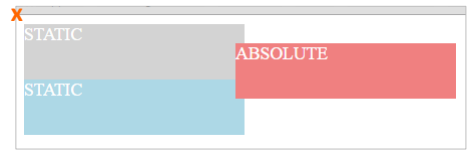
- Positioning
  - Absolute
    - The original position of the element is not retained, thus the original space occupied by the displaced element is filled by the following elements in the HTML flow.
    - Absolute positioning is performed in relation to the:
      - Browser Window
        - If the element to move is not contained in another element with position: relative, absolute or fixed, it will be positioned relatively to the browser window (initial containing block)
      - Container ≠ browser window
        - If the element is contained in another element with position: relative, absolute or fixed it will be positioned in relation to that container

CSS[properties]

- Positioning
  - Absolute

```
<div class="c1">STATIC </div>
<div class="c2">ABSOLUTE</div>
<div class="c3">STATIC</div>
```

```
<style>
  div {width:200px;height:50px;
  }
  .c1 {background-color:lightgray;
       color:white;
  }
  .c2 {background-color:lightcoral;
       color:white;
       position:absolute;
       left:200px;
       top:25px;
  }
     .c3 {background-color:lightblue;
       color:white;
  }
 </style>
```



  - The **reference for displacement** is the upper left corner of the browser window

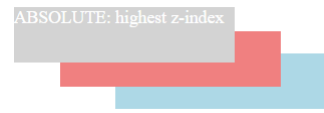
CSS[properties]

- Positioning (overlapping)
  - z-índex

Property	Example	Comments
z-index	div {z-index:5}	Values: number, auto*, inherit

```
<div class="c1">ABSOLUTE: highest z-index </div>
<div class="c2">ABSOLUTE</div>
<div class="c3">ABSOLUTE</div>
```

```
div {width:200px;height:50px;}
.c1 {background-color:lightgray;
     color:white;
     position:absolute;
     z-index:10;}
.c2 {background-color:lightcoral;
     color:white;
     position:absolute;
     left:50px;
     top:30px;
     z-index:5; }
.c3 {background-color:lightblue;
     color:white;
     position:absolute;
     left:100px;
     top:50px;
     z-index:1;}
```



Task 4.1

Task 4.1

- Slideshow without Repetition
  - Create an HTML file having the following structure:

```
<div id="slideshow">
    <img src="img1.png" alt="" class="active" />
    <img src="img2.png" alt="" />
    <img src="img3.png" alt="" />
    <img src="img4.png" alt="" />
    <img src="img5.png" alt="" />
</div>
```

  - Implement a CSS so that:
    - The div element acts **as a reference for the positioning** of images. This element should be centered with a margin of 100 px from top and must have a width of 500px.
    - All images are positioned according to the absolute positioning with zero displacement relative to the reference point defined.
    - The images will overlap, it is necessary to assure that an image without class attribute does not overlap the image whose attribute class = "active".
      - The image whose attribute class = "active" should be visible (to occupy the top position) in the overlap situation with the remaining images.

Task 4.1 - Solution

- Slideshow without Repetition
  - HTML Structure

```
<div id="slideshow">
    <img src="img1.png" alt="" class="active" />
    <img src="img2.png" alt="" />
    <img src="img3.png" alt="" />
    <img src="img4.png" alt="" />
    <img src="img5.png" alt="" />
</div>
```

  - CSS Code

```
<style>
    body {
        margin: 0;
    }

    #slideshow {
        position: relative;
        width: 500px;
        margin: 100px auto;
    }
```

```
#slideshow img {
    position: absolute;
    top: 0;
    left: 0;
    z-index: 8;
}
```

```
#slideshow img.active {
    z-index: 10;
}
```


End

Task 4.1

Task 4.2

Task 4.2

- Include the jQuery in your project.
- Create a new script
  - Set the function that allows you to ensure that the script execution occurs after downloading all the elements.
  - Declare a function called *slideSwitch*.
    - Declare two variables:
      - ***$activeImg*** variable that stores the element with class = "active“
      - ***$nextImg*** that stores the next image (next sibling of the $activeImg).

Task 4.2 - Solution

```
<script src="jquery-3.2.1.js"></script>
<script>
        $(document).ready(function(){
            function slideSwitch(){

                var $activeImg = $('img.active');
                var $nextImg = $activeImg.next();

            }
        });
```


Task 4.2

- The implementation of this slideshow is based on switching (on/off) of the class "active" once the image whose class is "active" prevails (occupies the top position) in the overlapping situation.
  - Thus, based on the methods to change attributes add the class "active" to **$nextImg** and delete the class "active" on the **$activeImg**.
    - End the function

Solution:

```
$(document).ready(function(){
    function slideSwitch(){

        var $activeImg = $('img.active');
        var $nextImg = $activeImg.next();

        $nextImg.addClass('active');
        $activeImg.removeClass('active');

    };

});
```

Preview in the browser!

Conclusions?

Task 4.2

- Call the function at regular intervals of 2s.
  - To do this, apply the global function:
    - *setInterval(arg1, arg2)*
      - arg1 is the function name
      - arg2 is the time interval set in ms

```
<script src="jquery-3.2.1.js"></script>
<script>
    $(document).ready(function(){
        function slideSwitch(){

            var $activeImg = $('img.active');
            var $nextImg = $activeImg.next();

            $nextImg.addClass('active');
            $activeImg.removeClass('active');

        };

        setInterval(slideSwitch,2000);      2 seconds
    });
</script>
```

Check the final result!

Can be improved?
Comments?

End

Task 4.2

Task 4.3

Task 4.3 (Loop)

- Slideshow with repetition (loop)
  - The incorporation of a loop operation is based on the identification of the next image to display.
  - Two possible situations:
    - The next image belongs to the sequence as the current image is in an intermediate position of the sequence of images;
    - The next image is the first image as the current image is the last one of the sequence of images.
  - Thus, this decision (implemented through and if (condition) {...} else {...} structure) is based on the existence of the next image.

Task 4.3 - Solution

```
<script src="jquery-3.2.1.js"></script>
<script>
    $(document).ready(function(){
        function slideSwitch(){
            var $activeImg = $('img.active');
            var $nextImg = $activeImg.next();

            if ($nextImg.length==0)
                $nextImg=$('img:first');

            $nextImg.addClass('active');
            $activeImg.removeClass('active');
        };
    setInterval(slideSwitch,2000);
    });
</script>
```

Loop!
Identification of the next image

Check the final result!

Can be improved?
Comments?

End

Task 4.3

Task 4.4

Task 4.4 (Progressive)

- ***Progressive*** slideshow with repetition (loop)
  - The slideshow implemented has no control on the image replacement speed, which doesn't assure a fluid display. Usually, to make the slideshow more pleasant, the speed of the image replacement must be controlled making it progressive.
    - animate(*CSS Property, duration (ms), function*) method
      - Add the class "active" to the $nextImg
      - set the CSS property opacity to 0.0;
      - Use the animate() method to apply an animation effect to the opacity property (final value of 1.0); with a duration of 2s (2nd argument).
      - The elimination of class "active" from the $activeImg must be carried out once the animation is finished (3rd argument).

Task 4.4 - Solution

```
<script src="jquery-3.2.1.js"></script>
<script>
        $(document).ready(function(){
            function slideSwitch(){

                var $activeImg = $('img.active');
                var $nextImg = $activeImg.next();

                if ($nextImg.length==0)
                    $nextImg=$('img:first');


                $nextImg.addClass('active').css({opacity:0})
                                           .animate({opacity:1.0},2000,function(){
                                                  $activeImg.removeClass('active');
                                            });
                };

            setInterval(slideSwitch,3000);
        });
</script>
```

progressive

Increase the interval to 3s
Check the final result!

Comments?

End

Task 4.4

Task 4.5

Task 4.5

- ***Progressive*** slideshow with repetition (loop)
  - Remain some problems in the transition of the images. These problems can be overcome through a unique/correct identification of the last viewed image.
    - Create a new CSS rule that applies to an image whose attribute class = "last-active" and whose value of the z-index property relies in the interval between the two defined values (e.g. 9).
    - Add the class "last-active" to $activeImg
    - Remove the class “active” from $activeImg
    - In the callback function of the animate() method, remove the class "last-active" from the $activeImg

Task 4.5

```
<script src="jquery-3.2.1.js"></script>
<script>
        $(document).ready(function(){

            function slideSwitch(){

                var $activeImg=$('#slideshow img.active');
                var $nextImg=$activeImg.next();

                if ($activeImg.next().length==0)
                    $nextImg=$('#slideshow img:first');

                $activeImg.addClass('last-active');
                $activeImg.removeClass('active');

                $nextImg.addClass('active')
                    .css({opacity: 0.0})
                    .animate({opacity: 1.0}, 2000, function() {
                        $activeImg.removeClass('last-active');
                });
            };

        setInterval(slideSwitch, 3000 );
        });
</script>
```


End

Task 4.5

Task 4.6

Task 4.6

- Events
  - The slideshow should only starts after a mouse click on the div "slideshow".

```
<script src="jquery-3.2.1.js"></script>
   <script>
       $(document).ready(function(){

       $('#slideshow').click(function(){

           function slideSwitch(){
               var $activeImg=$('#slideshow img.active');
               var $nextImg=$activeImg.next();

               if ($activeImg.next().length==0)
                   $nextImg=$('#slideshow img:first');

               $activeImg.addClass('last-active');
               $activeImg.removeClass('active');

               $nextImg.addClass('active')
                   .css({opacity: 0.0})
                   .animate({opacity: 1.0}, 2000, function() {
                       $activeImg.removeClass('last-active');
               });
           };

           setInterval(slideSwitch, 3000 );
           });
       });
   </script>
```

End

Task 4.6

Outline

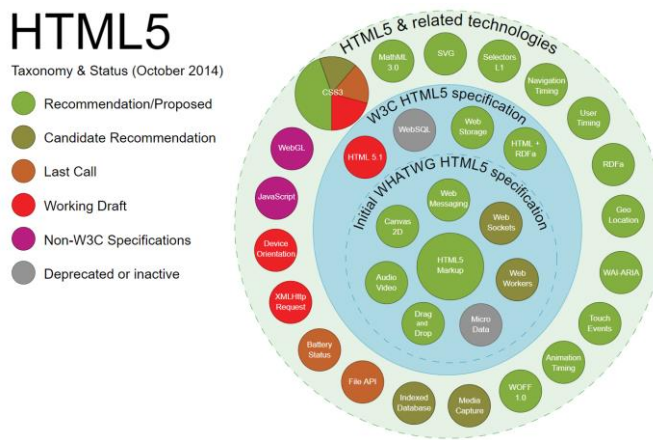
1. Structure Layer
   - *Hyper Text Markup Language (HTML)*
2. Presentation Layer
   - *Cascading Style Sheets (CSS)*
3. Behaviour Layer
   - *JavaScript*
   - *jQuery* ✓

APIs

Application Programming Interfaces

API

- *Application Programming Interfaces*
  - What the API can do? How to acess it ? sintax?



Geolocation API

- navigator.geolocation
  - The navigator object contains information about the browser.
  - *navigator.geolocation* returns a geolocation object that can be used to locate the user's position
    - Latitude; Longitude

method	Description
getCurrentPosition(success, fail)	If the user allows this method returns the user's position (latitude / longitude). **success (usually)** is the name of the function to call if the coordinates of the user's location are obtained. ***fail (usually)*** is the name of the function to call if the location coordinates are not obtained

Geolocation API

- *geolocation*
  - if the location of the user (success) is obtained, the Position object is passed to the callback function
    - coords is a child object of the Position object

Property	Description
Position.coords.latitude	Latitude (degrees)
Position.coords.longitude	Longitude (degrees)
Position.coords.accuracy	Accuracy (meters)
Position.coords.altitude	Height (meters)
Position.coords.altitudeAccuracy	Height's accuracy (meters)
Position.coords.speed	speed (meters/second)
…	

Geolocation API

- *geolocation*
  - if the user's location is not obtained (fail) the *PositionError object* is passed to the *callback function*

Property	Description
PositionError.code	Error code: 1 Permission denied 2 Unavailable 3 Timeout
PositionError.message	Message associated with the error (not available for the user)

Task 5.1

Geolocation API

- Obtain the location of the user (your location here in the university)
  - Replace the image and get the latitude/longitude of our location

Geolocation API

- 1. Verify if the browser supports the geolocation API

```
<h1> Where are you?</h1>
<div><img src="iseCoimbra.png"></div>
<div id="location">Trying to find the location!!</div>

<script>
    var elMap= document.getElementById('location');
    var msg="Sorry! Unable to find your location!";

    if (navigator.geolocation)
        {

        }
    else
        {

        }
</script>
```

Geolocation API

- 2. Get position
  - Get the current user position
  - Show message meanwhile the position is not obtained
  - Show message if geolocation does not exist

```
<h1> Where are you?</h1>
<div><img src="iseCoimbra.png"></div>
<div id="location">Trying to find the location!!</div>

<script>
    var elMap= document.getElementById('location');
    var msg="Sorry! Unable to find your location!";

    if (navigator.geolocation)
        {
            navigator.geolocation.getCurrentPosition(success,fail);
            elMap.textContent="Checking Position...";}
    else
        {elMap.textContent=msg;}

    ...
```


Geolocation API

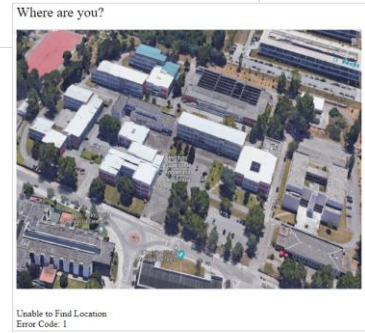
- 3. If *getcurrentPosition()* is well succeed, build a message to show the latitude and longitude

```
<h1> Where are you?</h1>
<div><img src="iseCoimbra.png"></div>
<div id="location">Trying to find the location!!</div>

<script>
    var elMap= document.getElementById('location');
    var msg="Sorry! Unable to find your location!";

    if (navigator.geolocation)
        {
            navigator.geolocation.getCurrentPosition(success,fail);
            elMap.textContent="Checking Position...";}
    else
        {elMap.textContent=msg;}

    function success (position){

        msg="<h3> Position </h3> <br>";
        msg +="Latitude: " + position.coords.latitude.toPrecision(5) + "<br>";
        msg +="Longitude: " + position.coords.longitude.toPrecision(5) + "<br>";

        elMap.innerHTML=msg;
    }
    ...
```

Geolocation API

- 4. Else, if location cannot be obtained, an error code must be provided

```
    ...
        function success (position){

        msg="<h3> Position </h3> <br>";
        msg +="Latitude: " + position.coords.latitude.toPrecision(5) + "<br>";
        msg +="Longitude: " + position.coords.longitude.toPrecision(5) + "<br>";

        elMap.innerHTML=msg;
    }

    function fail(err){
        elMap.innerHTML="Unable to Find Location <br> Error Code: " + err.code;
    }
</script>
```



End

Task 5.1

Web Storage

- Two diferent types:
  - ***Local Storage***

    The **localStorage** object stores the data with no expiration date. The data will not be deleted when the browser is closed
  - ***Session Storage***

    The **sessionStorage** object is equal to the **localStorage** object, except that it stores the data for only one session. The data is deleted when the user closes the specific browser tab.

    http://www.w3schools.com/html/html5_webstorage.asp

Storage	Local	Session
is the data stored when you close a window/tab?		
Can all open window/tabs access the data?		

Web Storage

- HTML5 introduces a storage object (local, session) that present several advantages when compared with the traditional cookies:
  - store more data
  - the storage is local, no need to communicate with the server
  - more secure.

https://www.nczonline.net/blog/2009/05/12/cookies-and-security/

methods	description
setItem(key,value)	New pair key/value
getItem(key)	Get a key's value
removeItem(key)	Removes the pair key/value
clear()	Clears all the information of a storage object

properties	description
length	Number of keys

Task 5.2

Web Storage API

- 1. Verify if the browser supports localStorage

```
<h2> Web Storage</h2>
  <form id="mainForm">
      Username: <input type="text"  id="username" class="textinput"><br>
      Password: <input type="text"  id="password" class="textinput"><br>
      <input type="submit"  id="submit" value="submit">
  </form>

  <script>
      if (window.localStorage)
          {



          }
  </script>
```

Web Storage API

- 2. Declare two variables to store the element objetcs with id="username" and id="password"

```
<script>
    if (window.localStorage)
        {
            var txtUser=document.getElementById('username');
            var txtPass=document.getElementById('password');


        }
</script>
```


Web Storage API

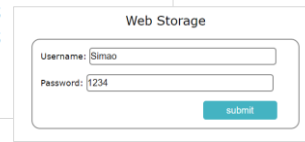
- 3. Each time the input event is triggered (input values) in one of the form fields, these values are saved automatically in the *localStorage* object (*setItem()*). Save the data that are inputted by the user.

```
<script>
    if (window.localStorage)
    {
        var txtUser=document.getElementById('username');
        var txtPass=document.getElementById('password');

        txtUser.addEventListener('input',function(){
                    localStorage.setItem('user',txtUser.value);
                }, false);

         txtPass.addEventListener('input',function(){
                    localStorage.setItem('pass',txtPass.value);
                }, false);
    }
</script>
```

Web Storage API

- 4. The values of the fields are obtained based on the getItem() method, whenever a refresh of the page is made and/or closing and opening of the browser window. Obtain the previous inputted data.

```
<script>
    if (window.localStorage)
        {
            var txtUser=document.getElementById('username');
            var txtPass=document.getElementById('password');

            txtUser.addEventListener('input',function(){
                        localStorage.setItem('user',txtUser.value);
                    }, false);

             txtPass.addEventListener('input',function(){
                        localStorage.setItem('pass',txtPass.value);
                    }, false);

            txtUser.value=localStorage.getItem('user');
            txtPass.value=localStorage.getItem('pass');
        }
</script>
```



Web Storage API

- 5. Repeat the process considering the sessionStorage

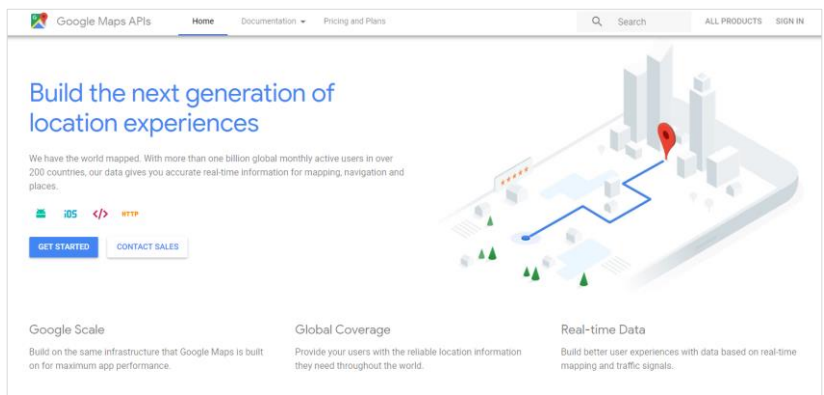
```
<script>
    if (window.sessionStorage)
        {
            var txtUser=document.getElementById('username');
            var txtPass=document.getElementById('password');

            txtUser.addEventListener('input',function(){
                        sessionStorage.setItem('user',txtUser.value);
                    }, false);

             txtPass.addEventListener('input',function(){
                        sessionStorage.setItem('pass',txtPass.value);
                    }, false);

            txtUser.value=sessionStorage.getItem('user');
            txtPass.value=sessionStorage.getItem('pass');
        }
</script>
```

End

Task 5.2

Platform API's

- Google API
  - It allows to display and define how maps (form, type of information, ...) of Google maps in a given project will be displayed



https://developers.google.com/maps/
http://maps.googleapis.com/maps/api/js?callback=init

Google API

- The creation of a map is done based on the Map() constructor that accepts two parameters:
  - The HTML element on which the map will be made available
  - A set of map options that define how the map will be displayed:
    - **center**
      - can be defined literally or through an object https://developers.google.com/maps/documentation/javascript/examples/map-latlng-literal

```
center: {lat: -34.397, lng: 150.644}
```

    - **mapTypeId**
      - allows the definition of four map types https://developers.google.com/maps/documentation/javascript/maptypes?hl=pt-es#MapTypes
    - **zoom**
      - Zoom level between 0 e 20

- 1: World
- 5: Landmass/continent
- 10: City
- 15: Streets
- 20: Buildings

Task 5.3

Task 5.3

- Create an html structure, with a title <h2> that contains the name of your university and with a
  - <h2> must have font-size:3em; font-weight:500; color:#1287A8; text-align:center
  - must have height:400px; width:100%

- To access the Google API the source of the script must be:

  http://maps.googleapis.com/maps/api/js? callback=init

- Before this script insertion, the init() function must be defined
  - Create other script

Task 5.3 - Solution

```
    <h2>Coimbra Engineering Institute</h2>
    <div id="map"></div>

    <script>
       function init(){


       }
    </script>
     <script src='http://maps.googleapis.com/maps/api/js?callback=init'></script>
</body>
```

Task 5.3

- init() function
  - A mapOptions object is created with the following properties:
    - center: {lat: ***latitudeOfYourUniversity***, lng: ***longitudeOfYourUniversity***},
    - mapTypeId: ***try 'roadmap'/'satellite'/'terrain'***,
    - zoom: 13 ***(modify between 0 and 20)***,
  - The map is created based on the Map () constructor, which takes two parameters:
    - The HTML element where the map will be created
    - The mapOptions object.

Task 5.3 - Solution

```
    <h2>Coimbra Engineering Institute</h2>
    <div id="map"></div>

    <script>
       function init(){
            var mapOptions={
               center: {lat: 40.192, lng:-8.412},
               mapTypeId: 'roadmap',
               zoom:13,
            };

            var isecMap=new google.maps.Map(document.getElementById('map'),mapOptions);
       }
    </script>
     <script src='http://maps.googleapis.com/maps/api/js?callback=init'></script>
</body>
```

End

Task 5.3

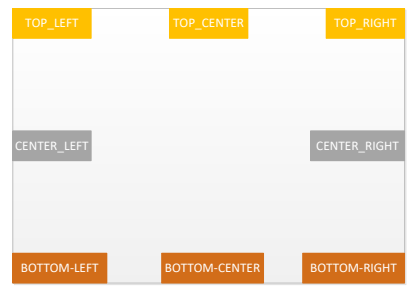
Google API

- Additional Visualization Properties (controls):

Properties	Description
zoomControl	Zoom level (slider for bigger maps; buttons for smaller maps)
scaleControl	Show the scale on the map
mapTypeControl	Change the map type (roadmap, satellite)
streetViewControl	Contains a Pegman icon, which can be dragged to the map to activate Street View
…	

https://developers.google.com/maps/documentation/javascript/controls#ControlOptions

- Controls position in the map:



Task 5.4

Google API

Show the controls

(slide 260)

```
function init(){
        var mapOptions={
            center: {lat: 40.192, lng:-8.412},
            mapTypeId: 'roadmap',
            zoom:13,

            zoomControl:true,
            mapTypeControl:true,
            scaleControl:true,
            streetViewControl:true,
        };
```

Controls show/hide

Coimbra Engineering Institute

Google API

- View Control
  - Each control has its own **options object** to control the style and position.

https://developers.google.com/maps/documentation/javascript/reference/3.exp/control

```
function init(){
     var mapOptions={
     center: {lat: 40.192, lng:-8.412},
     mapTypeId: 'roadmap',
     zoom:13,

     zoomControl:true,
     zoomControlOptions:{
         style:google.maps.ZoomControlStyle.SMALL,
         position:google.maps.ControlPosition.TOP_RIGHT,
     },

     mapTypeControl:true,
     scaleControl:true,
     streetViewControl:true,
 }
```

Google API

- mapType Control

```
var mapOptions={
   center: {lat: 40.192, lng:-8.412},
   mapTypeId: 'roadmap',
   zoom:13,

   zoomControl:true,
   zoomControlOptions:{
       style:google.maps.ZoomControlStyle.SMALL,
       position:google.maps.ControlPosition.TOP_RIGHT,
       },

   mapTypeControl:true,
   mapTypeControlOptions:{
       style:google.maps.MapTypeControlStyle.DROPDOWN_MENU,
       position:google.maps.ControlPosition.TOP_LEFT,
       },
   scaleControl:true,
   streetViewControl:true,
};
```

End

Task 5.4

Google API

- Style a map
  - Property **styles** of the mapOptions Object
  - The value of this property is an **array of objects**
    - Different objects are defined to define different types of features on the map
    - For each object, you can define three properties:
      - featureType
        - The type of feature you want to change ex: roads, parks, ...; If the featureType is not defined the change applies to the entire map
      - elementType
        - The part of the feature that you want to change ex: geometry, labels, ...
      - stylers (array of objects)
        - properties that allow you to adjust the color and visibility of map elements:
          - hue (color adjustment, value in hexadecimal
          - saturation / lightness (assuming values between -100 and 100)
          - visibility ("on", "off", "simplified")

Google API

```
var mapOptions={
    center: {lat: 40.192, lng:-8.412},
    mapTypeId: 'roadmap',
    zoom:13,

    zoomControl:true,
    zoomControlOptions:{
        style:google.maps.ZoomControlStyle.SMALL,
        position:google.maps.ControlPosition.TOP_RIGHT,
    },

    mapTypeControl:true,
    mapTypeControlOptions:{
        style:google.maps.MapTypeControlStyle.DROPDOWN_MENU,
        position:google.maps.ControlPosition.TOP_LEFT,
    },
    scaleControl:true,
    streetViewControl:true,

    styles:[  {
                stylers:[{hue:'#00ff6f',
                        saturation:-50}]
              },
              {
                featureType:'road',
                elementType:'geometry',
                stylers: [{lightness:100,
                        visibility:"simplified"}]
              },
        ]
};
```

Applied to the entire map; roads

Google API

```
styles:[        {
                stylers:[{hue:'#00ff6f',
                          saturation:-50}]
                 },
                {
                    featureType:'road',
                    elementType:'geometry',
                    stylers: [{lightness:100,
                              visibility:"simplified"}]
                },
                {
                    featureType:'water',
                    elementType:'geometry',
                    stylers: [{hue:'#C4F4F4'}]
                }
        ]
    };
```

Applied to water (rivers, …)

Google API

```
styles:[  {
            stylers:[{hue:'#00ff6f',
                    saturation:-50}]
          },

          {
              featureType:'road',
              elementType:'geometry',
              stylers: [{lightness:100,
                      visibility:"simplified"}]
          },
          {
              featureType:'water',
              elementType:'geometry',
              stylers: [{hue:'#C4F4F4'}]
          },
          {
              featureType:'transit',
              elementType:'labels',
              stylers: [{hue:'#FFA079', saturation:+80}]
          }
    ]
```

Applied to Public Transportation

Coimbra Engineering Institute

Google API

```
styles:[  {
            stylers:[{hue:'#00ff6f',
                    saturation:-50}]
          },

          {
              featureType:'road',
              elementType:'geometry',
              stylers: [{lightness:100,
                      visibility:"simplified"}]
          },
          {
              featureType:'water',
              elementType:'geometry',
              stylers: [{hue:'#C4F4F4'}]
          },
          {
              featureType:'transit',
              elementType:'labels',
              stylers: [{hue:'#FFA079', saturation:+80}]
          },
          {
              featureType:'road',
              elementType:'labels',
              stylers: [{visibility:'off'}]
          }

    ]
};
```

Hide road labels

Coimbra Engineering Institute

Google API

- Markers
  - The Marker() constructor creates a marker object whose properties are:
    - position (position on the map)
    - map (map that contains the bookmark, a page can have more than one bookmark)
    - icon (the location of the file containing an image to display)

Coimbra Engineering Institute

Google API

```
            ...,
                    {
                        featureType:'road',
                        elementType:'labels',
                        stylers: [{visibility:'off'}]
                    }
                ],
        };

        var isecMap = new google.maps.Map(document.getElementById('map'), mapOptions);

        var startPosition = new google.maps.Marker({
            position:{lat:40.192, lng:-8.412},
            map:isecMap,
            icon:'isec1.png'
        });
    }

</script>
<script src='http://maps.googleapis.com/maps/api/js?callback=init'></script>
```

Coimbra Engineering Institute

END!

Thank you…

sparedes@isec.pt