CAPÍTULO

HTML

JPSantos
2023

3. HTML – HiperText Markup Language

https://www.w3.org/wiki/Html/Specifications

3.1. Conceitos sobre HTML

HTML significa Hypertext Markup Language. Esta linguagem foi proposta por Tim-Berners e tinha por objectivo permitir escrever e formatar documentos com texto, tabelas
e hiperligações para outros documentos, imagens, filmes e ficheiros em geral.
De uma forma simples e incompleta podemos dizer que esta linguagem define um conjunto de palavras chave, etiquetas, <tags> que um browser Web sabe interpretar e em função delas apresenta ao utilizador uma página WEB totalmente formatada.
Com base nestas etiquetas é possível escrever documentos de texto com uma estrutura e uma formatação conhecida, não proprietária, podendo por isso ser utilizadas por todos.
As páginas WEB, escritas em HTML, são documentos de texto que têm de ser previamente gravados, por exemplo em disco, antes de poderem ser lidas pelos browsers WEB.

3.1.1. Estrutura de um documento HTML

Um documento HTML está organizado em duas partes principais: cabeçalho “header” e o corpo principal “body”. O documento HTML, como um todo, está contido pelas etiquetas <HTML> … </HTML>.

```
<HTML>
        <HEAD>
        ….
        </HEAD>
        <BODY>
        …..
        </BODY>
</HTML>
```

3.1.2. Etiquetas HTML

As etiquetas HTML estão definidas como um conjunto de palavras reservadas que aparecem no documento de texto precedidas pelo sinal menor e seguidas pelo sinal maior, envolvendo a palavra chave <etiqueta>. As palavras reservadas <HTML>, <HEAD> e <BODY> são alguns exemplos de etiquetas. A maior parte destas etiquetas são apresentadas na secção seguinte e são explicadas exemplo a exemplo.

3.2. Exemplos de páginas HTML

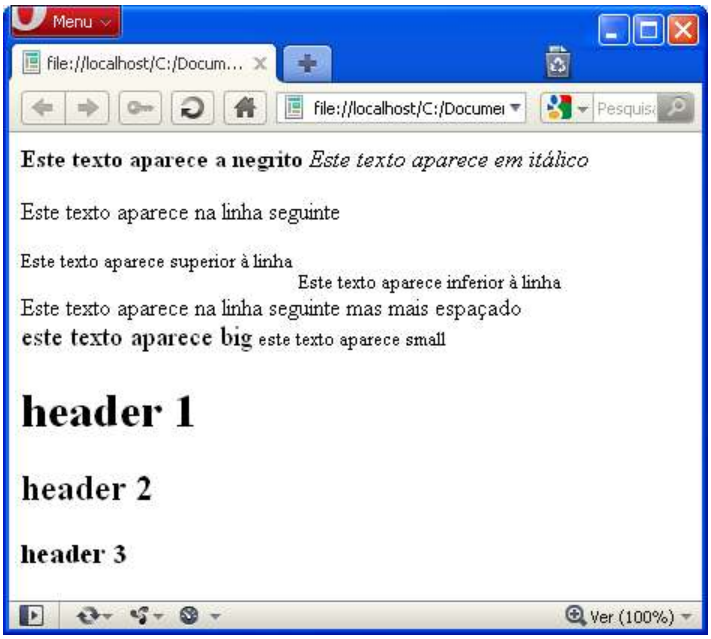
Guarde num ficheiro de texto diferente, cada um dos exemplos seguintes. O primeiro exemplo deve ficar gravado com o nome "exemplo1.html" e os exemplos seguintes deverão ter um nome à sua escolha. Depois de editar e guardar em disco cada um dos exemplos seguintes visualize-os com um browser WEB.
Em cada exemplo é apresentado o "texto" HTML e o aspecto da página WEB correspondente, tal como o Browser a interpreta e a visualiza. Estes exemplos são autoexplicativos, as etiquetas aparecem a avermelho nos exmplos.

3.2.1. Estrutura de um documento HTML

```
<!DOCTYPE html>
  <html>
        <head>
              <title>
              Um titulo para o documento
              </title>
        </head>
  <body>
        Este é o meu primeiro documento em HTML
  </body>
  </html>
```

3.2.2. Formatar texto em HTML

```
<!DOCTYPE html>
<html>
      <head>
      <meta http-equiv="refresh" content="1" />
      </head>
      <body>
            <b> Este texto aparece a negrito </b>
            <i> Este texto aparece em itálico </i>
            <sup> Este texto aparece superior à linha </sup>
            <sub> Este texto aparece inferior à linha </sub>
            <p> Este texto aparece na linha seguinte </p>
            <br> Este texto aparece na linha seguinte mas mais espaçado
            <br>
            <big>este texto aparece big</big>
            <small>este texto aparece small</small>
            <h1> header 1 </h1>
            <h2> header 2 </h2>
            <h3> header 3 </h3>
      </body>
</html>
```



3.2.3. Hiperligações

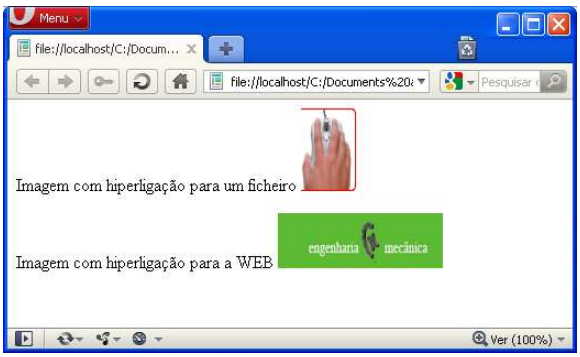
```
<!DOCTYPE html>
<html>
        <head>
        </head>
        <body>
                <p>
                <a href=teste2.html>  Este texto </a> liga a um ficheiro
                </p>

                <p>
                <a href=http://www.clix.pt>  Este texto </a> liga a um site
                </p>

                <p>
                <a href=teste2.html target =“_blank”>  Este texto </a>
                liga a um ficheiro e abre uma nova janela
                </p>

        </body>
</html>
```

3.2.4. Inserir imagens

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>

        <p>Inserir uma imagem em movimento:
        <img src="hand1.gif" border ="0" width="50" height="75">
        </p>

        <p>Inserir uma imagem da WWW:
        <img src="http://localhost/hand1.gif" border ="0" width="200" height="75">
        </p>
</body>
</html>
```

3.2.5. Inserir imagens com hiperligação

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
        <p> Imagem com hiperligação para um ficheiro
        <a href=teste2.html />  <img border=0 src=hand1.gif width=50 Height=75 >
        </a>
        </p>

        <p> Imagem com hiperligação para a WEB
        <a href =http://cim3.mec.ua.pt />  <img border=0 src=logo_2.jpg width=200 Height=50 >
        </a>
        </p>
</body>
</html>
```



3.2.6. Inserir imagens com áreas de hiperligação

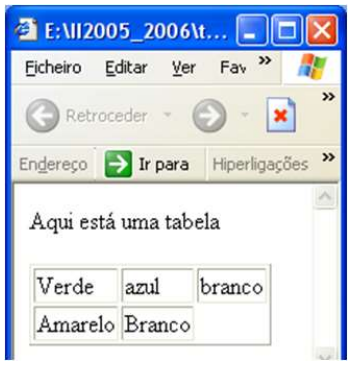
```
<!DOCTYPE html>
<html>
    <body>
        <map name="mapa" id="NavMapa">
        <area shape="rect" coords="0,0,5,5" href="teste.html" alt="1Quad" />
        <area shape="circle" coords="50,50,30" href="teste2.html" alt="2Quad" />
        </map>
    <img src="hand1.gif" with="100" height= "100"  usemap="#mapa"  />

    </body>
</html>
```

Exemplo

```
<html>
<body>

<map name="mapa" id="NavMapa">
<area shape="rect" coords="400,400,500,500" href="teste2.html" />
<area shape="circle" coords="320,450,35" href="teste2.html" />
</map>
<img src="painel.png" with="500" height="500" usemap="#mapa" />

</body>
</html>
```

3.2.7. Criar tabelas

```
<!DOCTYPE html>
<html>
<head></head>
<body>
        <p> Aqui está uma tabela </p>
        <table border=1>
                <tr>  <td>Verde   </td> <td>azul</td> <td>branco</td>  </tr>
                <tr><td>Amarelo</td><td>Branco</td><td></td></tr>
        </table>
</body>
</html>
```



3.2.8. Criar formulários

```
<!DOCTYPE html>
<html>
<body>
        <form>
        Nome:           <input type=text   name= nome>   <br>
        Masculino:      <input type=radio checked name="sexo" value="M" >
        Feminino:       <input  type=radio name="sexo" value="F" >
        <br> Marca de carro preferida
        <select name=auto>
                <option value="v"> Volvo
                <option value="s" selected> Saab
                <option value="a"> Audi
        </select> <br> <br>
        <input type=reset value=Limpar>
        <input type=submit value=Enviar>
        </form>
</body>
</html>
```

3.2.9. Criar frames

```
<!DOCTYPE html>
```

Frames horizontais

```
<html>
<FRAMESET  ROWS="5%,*,50%">
        <frame src="teste1.html" name="cima" id="idcima" />
        <frame src="teste2.html" name="meio" id="idmeio" />
        <frame src="teste3.html" name="baixo" id="idbaixo" />
</FRAMESET>
</html>
```

Frames verticais

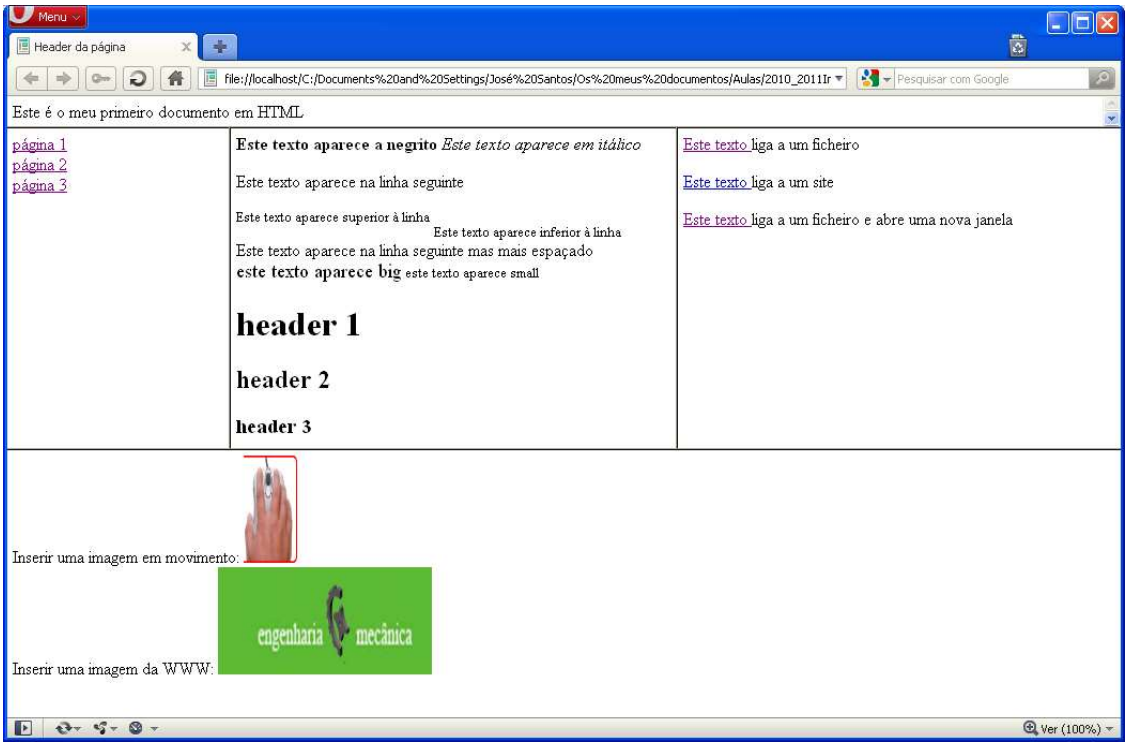
```
<html>
<FRAMESET  COLS="30%,*,50%">
        <frame src="teste1.html" name="esquerda" id="idesquerda" />
        <frame src="teste2.html" name="meio" id="idmeio" />
        <frame src="teste3.html" name="direita" id="iddireita" />
</FRAMESET>
</html>
```

3.2.10. Criar frames com targets

Como a etiqueta <FRAMESET> foi descontinuada, dessa forma não pode estar dentro das etiquetas <BODY> caso contrário os novos Browser não as interpretam corretamente.

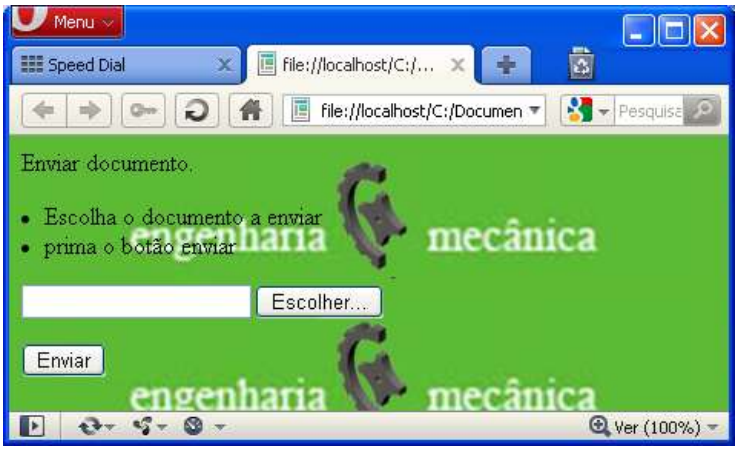
```
<!DOCTYPE html>
 <html>
  <FRAMESET  ROWS="5%,*,50%">
      <frame src="teste.html" name="cima" id="cima" />
      <frameset  cols="20%,*,40%">
           <frame src="indice.html" name="esq" id="esq" />
           <frame src="teste2.html" name="centro" id="centro" />
           <frame src="teste3.html" name="dir" id="dir" />
      </frameset>
      <frame src="teste4.html" name="baixo" id="baixo" />
      </FRAMESET>
 </html>
```

```
<html>
     <body>
     <a href="teste.html"  target="centro" >página 1 </a><br>
     <a href="teste1.html" target="centro" >página 2 </a><br>
     <a href="teste2.html" target="centro" >página 3 </a><br>
     </body>
</html>
```



3.2.11. Enviar um ficheiro para o servidor

```
<!DOCTYPE html>
<html>
<body background="logo_2.jpg" >
<form action="teste15FileSend.html" method="post">
        <p>Send me your resume.</p>

        <li>Escolha o documento a enviar</li><li>prima o botão enviar</li>
        <p><input type="file" enctype="multipart/form-data" name="resume" id="resume" /></p>
        <p><input type="submit" value="Enviar" name="submit" id="submit" /></p>
</body>
</html>
```



3.2.12. Criar Iframes com targets

Uma inline frame (iframe) permite referir uma página Web, dentro de outra página Web.
https://www.w3schools.com/html/html_iframe.asp

```
<iframe src="URL"></iframe>
<iframe src="demo_iframe.htm" height="200" width="300"></iframe>
```

O "target" de uma hiperligação pode ser uma iFrame. Neste caso na iframe será visualizada a página WEB referida na hiperligação.

```
<iframe src="demo_iframe.htm" name="iframe_a"></iframe>
<p><a href="https://www.w3schools.com" target="iframe_a">W3Schools.com</a></p>
```

3.2.13. Criar um botão

Criar um botão numa página html (www.w3school.com)

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

<h1>The button Element</h1>

<button type="button" onclick="alert('Ligar!')">Ligar</button>

</body>
</html>
```

3.3. Exemplos em Javascript

3.3.1. Exemplo I em javascript

Neste exemplo o "texto" HTML e Javascript residem no mesmo ficheiro "pag1_javascript.html", com a extensão html (https://www.w3schools.com).

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

<h2>My First JavaScript</h2>

<button type="button"
onclick="document.getElementById('demo').innerHTML = Date()">
Click me to display Date and Time.</button>

<p id="demo"></p>

</body>
</html>
```

A página que aparece no Browser:

A página que aparece no browser depois de premir o botão:

Experimentar: https://www.w3schools.com/js/js_intro.asp

3.3.2. Exemplo II em javascript - função

Criação de uma função em javascript. Neste exemplo a função javascript é criada no cabeçalho do documento HTML , entre as etiquetas <head>…. </head>

O script javascript é criado entre as etiquetas <script> …. </script>

A função javascript é chamada pelo browser quando o utilizador premir o botão.

```
<!DOCTYPE html>
<html>
<head>
<script>
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>
</head>
<body>

<h2>Demo JavaScript in Head</h2>

<p id="demo">A Paragraph</p>
<button type="button" onclick="myFunction()">Try it</button>

</body>
</html>
```

Exemplo de uma função que recebe parametros quando é chamada e que retorna um valor

```
function toCelsius(fahrenheit) {
  return (5/9) * (fahrenheit-32);
}
document.getElementById("demo").innerHTML = toCelsius(77);
```

Ver também as funções javascript:

- Writing into an HTML element, using innerHTML.
- Writing into the HTML output using document.write().
- Writing into an alert box, using window.alert().
- Writing into the browser console, using console.log().

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

<h1>My First Web Page</h1>
<p>My First Paragraph</p>

<p id="demo"></p>

<script>
```

```
document.getElementById("demo").innerHTML = 5 + 6;
</script>

</body>
</html>
```

3.3.3. Exemplo III em Javascript – função com parâmetros

Exemplo III - Criação de variáveis com operações matemáticas num script javascript, dentro de um documento HTML, interpretado por um browser WEB.

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

<h2>JavaScript Statements</h2>

<p>Multiple statements on one line are allowed.</p>

<p id="demo1"></p>
<script>
let a, b, c;
a = 5; b = 6; c = a + b;
document.getElementById("demo1").innerHTML = c;
</script>

</body>
</html>
```

Exemplo IV - Eventos HTML
Alguns eventos HTML que podem ser usados por exemplo para chamar scripts em Javascript.

- An HTML web page has finished loading
- An HTML input field was changed
- An HTML button was clicked

```
<button onclick="document.getElementById('demo').innerHTML = Date()">The time is?</button>
Ou simplesmente
<button onclick="displayDate()">The time is?</button>
```

onchange	An HTML element has been changed
Onclick	The user clicks an HTML element
onmouseover	The user moves the mouse over an HTML element
onmouseout	The user moves the mouse away from an HTML element
onkeydown	The user pushes a keyboard key
onload	The browser has finished loading the page

3.3.4. Exemplo IV em Javascript – setTimeout

Neste exemplo a função “myFunction” em javascript é chamada pelo browser de 3 em 3 segundos, usando a função “setTimeout”

```
<!DOCTYPE html>
<html>

<body>
<h2>JavaScript Callback</h2>
<p>Wait 3 seconds (3000 milliseconds) for this page to change.</p>
<h1 id="demo"></h1>

<script>
setTimeout(myFunction, 3000);


function myFunction() {
  document.getElementById("demo").innerHTML = "OLA !!";
}
</script>
</body>

</html>
```

3.4. HTML5

HTML Tom Berners in, 1991
HTML4.01 in 1999 w3c
HTML5 in 2012

```
<! DOCTYPE html>
<html lang="eng">
    <head>
        <meta charset= "utf-8">
        <meta  name="description"  value="html5">
        <meta  name="keywords"  value="html5">
    <head>
    <body>
    </body>
</html>
```

HTML5
https://www.youtube.com/watch?v=CPcS4HtrUEU

HTML 5.2, W3C Recommendation, 14 December 2017
https://www.w3.org/TR/html52/

3.4.1. Diferenças entre HTML e HTML5

https://www.w3schools.com/html/html5_new_elements.asp

- html :permite criar web pages and web applications
- html5: permite estruturar a apresentação
  - vídeo e áudio
  - gráficos (Canvas e SVG)
  - Armazenamento (Web SQL database & web storage)
  - Web browser

Em HMTL5 passaram a existir novas etiquetas:
<canvas>, <menu> <figure><audio>, <video>, <article>,<aside>
<nav><section><footer><header><main><nav>

exemplos:

```
<canvas id="myCanvas" width="200" height="100" style="border:1px solid #000000;" >
</canvas>

<body contextmenu="new-context-menu">
        <menu id="new-menu"  type="context">
                <menu item> Hello</menuitem>
        </menu>
</body>

<figure>
        <img src="...jpg"  alt="img description">
</figure>

<audio autoplay="autoplay" controls="controls" >
        <source src="music.ogg" />
        <source src="music.mp3" />
</audio>
```

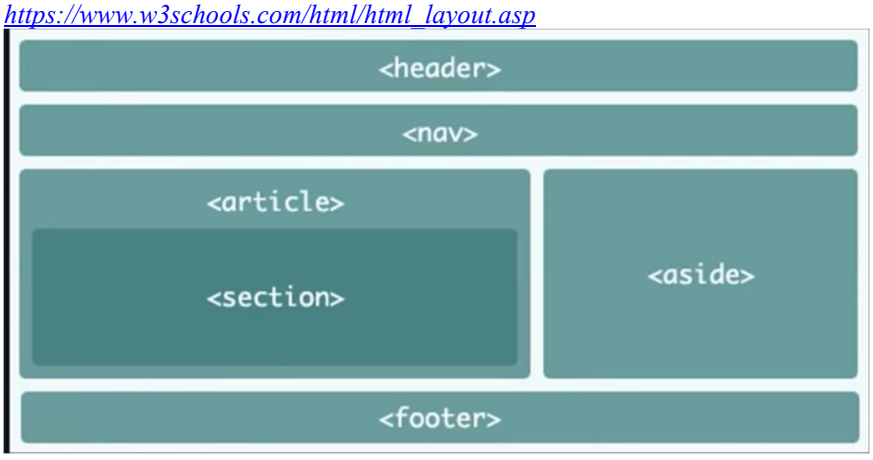
3.4.2. HTML5 - Layout elements

Antes podiam-se criar divisões no documento HTML desta forma:

```
<div class="article">...</div>
```

Em HTML5 surgiram novas etiquetas:
<article>...</article>
<aside> ... </aside>
<nav> ... </nav>
<section> ... </section>
<footer> ... </footer>
<header> ... </header>

https://www.w3schools.com/html/html_layout.asp

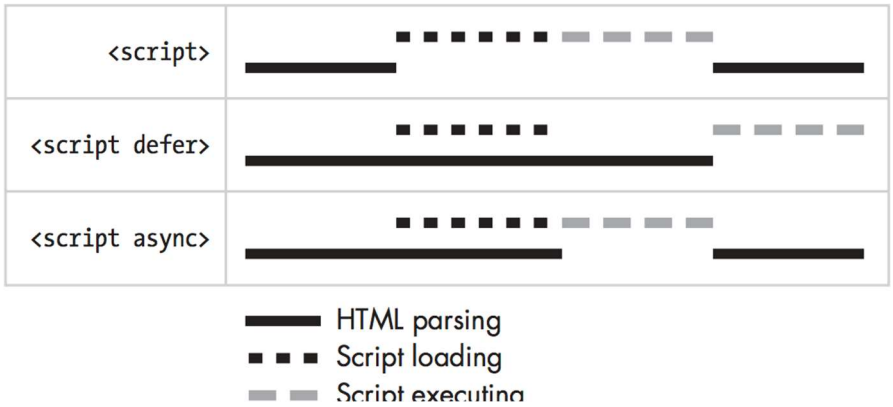


3.4.3. HTML5 - Referência a ficheiros em Javascript

No documento HTML pode ser feita referência a ficheiros/sripts .js
O Browser abre o ficheiro HTML e enquanto interpreta/mostra o documento HTML pode:
- descarregar o ficheiro javascript em simultâneo e só depois da página HTML estar visível, executar o script/ficheiro .js
- Ou, enquanto interpreta o HTML e visualiza a página WEB, pode em simultâneo descarregar o script javascript e começar logo a executá-lo.

A inclusão do ficheiro javascript pode ser feita de duas formas:

```
<script src="foo.js" ></script>
ou
 <script src="foo.js" defer></script>
ou
<script src="foo.js" async></script>
```



Os elementos que existem numa página HTML podem ser identificados pelas suas propriedades "name", "id", ...
Em Javascript e em CSS é possível identificar esses elementos HTML e actuar sobre eles, fazendo:

```
getElementById()
getElementsByTagName()

var foo = $('p.foo');
var foo = document.querySelector('p.foo');
var foo = document.querySelectorAll('p.foo');
var el = document.getElementsByClassName('foo');
```

3.4.4. HTML5 - Responsive

Em HTML e CSS é possível indicar ao Browser para adaptar a página WEB às dimensões do écran do telemóvel, computador.

Em HTML

```
<meta name="viewport" content="width=device-width, initial-scale=1.0”>
```

O tamanho do texto altera-se de acordo com a dimensão da janela

```
<h1 style="font-size:10vw">Hello World</h1>
```

Em CCS

Se a propriedade “width” CSS de uma imagem estiver definida como 100%, a imagem será responsiva e aumentará e diminuirá de acordo com as dimensões do écran.

```
<img src="img_girl.jpg" style=“width:100%;”>
```

3.4.5. HTML5 - Web Workers

Enquanto o browser executa um script em Javascript pode ficar ou não dedicado em exclusivo a essa atividade.
Um “Web worker” permite executar código Javascript em paralelo, em “background”, em simultâneo, sem a página WEB ficar a aguardar a conclusão do script.

----- Criar o WebWorkers que executa o script “demo_workers.js”

```
if (typeof(w) == "undefined") {
  w = new Worker(“demo_workers.js");     // cria o Web workers
}
```

---- Documento em Javascript que será tratado como um WebWorker “demo_workers.js”

```
var i = 0;
function timedCount() {
  i = i + 1;
  postMessage(i);                    // retorna o valor de “i”
  setTimeout("timedCount()",500);
}

timedCount();
```

exemplo:
https://www.w3schools.com/html/html5_webworkers.asp

--- Documento HTML com Javascript e o WebWorker

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

<p>Count numbers: <output id="result"></output></p>
<button onclick="startWorker()">Start Worker</button>
<button onclick="stopWorker()">Stop Worker</button>

<script>
var w;

function startWorker() {
  if (typeof(Worker) !== "undefined") {
    if (typeof(w) == "undefined") {
      w = new Worker("demo_workers.js");          // executa o script Javascript como um "Web Worker"
    }
    w.onmessage = function(event) {               // os eventos trocados com o WebWorker
      document.getElementById("result").innerHTML = event.data;
    };
  } else {
    document.getElementById("result").innerHTML = "Sorry! No Web Worker support.";
  }
}

function stopWorker() {
  w.terminate();
  w = undefined;
}
</script>

</body>
</html>
```

3.4.6. HTML5 - CANVAS

<canvas id="myCanvas" width="200" height="100"></canvas>

// o Browser desenha uma linha na página

```
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.moveTo(0, 0);
ctx.lineTo(200, 100);
ctx.stroke();
```

// O browser desenha o texto

```
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.font = "30px Arial";
ctx.strokeText("Hello World", 10, 50);
```

3.4.7. HTML5 - SVG

A etiqueta SVG podem conter uma imagem SVG

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

<svg width="100" height="100">
  <circle cx="50" cy="50" r="40" stroke="green" stroke-width="4" fill="yellow" />
</svg>

</body>
</html>
```

Bibliografia:

HTML5 Tutorial For Beginners - part 1 of 6 - Getting Started
https://www.youtube.com/watch?v=9gTw2EDkaDQ

<h1 style="text-align:center; color:red"> Exemplo </h1>
<img src="bola.png" alt="Bola" height="168" width="100">
*Página da UA *
* <img src="bola.png" alt="Bola" height="168" width="100"> *
<audio> src="bell_audio.mp3" controls> Audio element not supported by your browser </audio>

<audio controls autoplay loop>
<source src="bell_audio.mp3" type="audio/mpeg">
<source src="bell_audio.ogg" type="audio/ogg">
Audio element not supported by your browser.
</audio>

https://www.youtube.com/watch?v=dDn9uw7N9Xg
<article>,<aside> <nav><section><footer><header><main><nav><section>

.....

HTML vs HTML5 | Difference between HTML and HTML5 | HTML Tutorial | Edureka
https://www.youtube.com/watch?v=vHmUVQKXlVo

CAPÍTULO

CSS

JPSantos
2021

4. CSS – Cascade Style Sheet

Num documento HTML diversos elementos podem ser formatados de diferentes formas: tipo de letra, cor, alinhamento, etc.

Se no documento HTML, para além dos "elementos HTML", com diferentes "name" , "id", "class", etc, for feita referência a outro documento/ficheiro do tipo "stylesheet", o browser pede esse ficheiro e de acordo com esse ficheiro CSS apresenta no écran a página WEB devidamente formatada.

https://www.w3schools.com/cssref/default.asp

Uma forma de usar o CSS, "extern CSS", consiste em criar um ficheiro com a extensão CSS, e no documento HTML fazer referencia a ele.

Também deve fazer usar o nome dos elementos/selector HTML, no ficheiro CSS, para os "formatar" de acordo com um conjunto de "CSS properties":

http://www.littlewebhut.com/css/

Se no documento, ficheiro html:

```
<body>
<h1>My First CSS Example</h1>
<p>Este parágrafo.</p>
</body>
```

E no documento CSS, ou no mesmo documento HTML fizermos

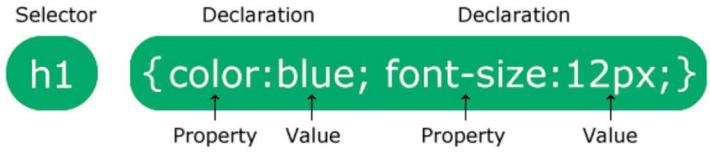
```
<style>
p {
  color: red;
  text-align: center;
}
</style>
```

A frase “Este parágrafo” aparece com a formatação indicada
(exemplo seguinte, com HTML e CSS no mesmo documento).

Ficheiro CSS0.html

```
<html>
<head>
<meta charset=\"UTF-8\">\
<style>
  p { color: red; text-align: center;}
</style>
</head>

<body>
  <h1>My First CSS Example</h1>
  <p>Este parágrafo.</p>
</body>

</html>
```

CSS Syntax



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

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

4.1. CSS - Exemplo I

O exemplo seguinte tem dois ficheiros, um ficheiro HTML e um ficheiro CSS.
O botão aparece formatado de acordo com o ficheiro CSS.

Para isso ser possível, na página HTML é feita referência a um ficheiro CSS com o nome "style.css".
A página HTML contém também um botão do tipo "submit" com *class="button"* .

No documento CSS é definido que a *class="button"* deve ter um conjunto de atributos/properties.

-----Ficheiro HTML , com o nome "index_button.html"-----

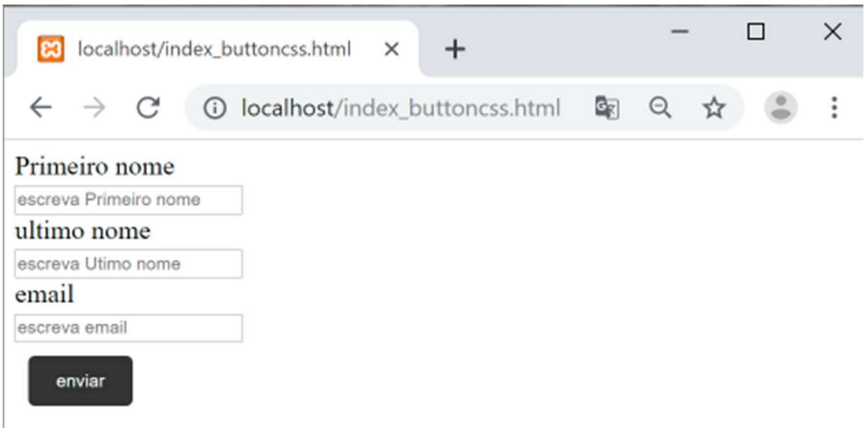
```
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset=\"UTF-8\">
    <link rel="stylesheet" href="styleButton.css"> // é feita referência ao ficheiro CSS:   style.css
</head>

<body>
        <form>
                <label for="primeironome">Primeiro nome</label><br>
                <input type="text" placeholder="escreva Primeiro nome"><br>
                <label for="ultimonome">ultimo nome</label><br>
                <input type="text" placeholder="escreva Utimo nome"><br>
                <label for="email">email</label><br>
                <input type="text" placeholder="escreva email"><br>
                <input type="submit" value="enviar" class="button"> <br>
        </form>

</body>
</html>
```

-----Ficheiro CSS , com o nome "styleButton.css"-----

```
.button{
   margin-top: 10px;
   margin-left: 10px;
   padding: 10px 20px;
   border: 1px solid transparent;
   background: #333;                    // fundo a preto
   color: white;                        // letras a branco
   border-radius: 5px;                  // cantos do botão arredondados
   cursor: pointer;
   transition: all 1s;
}

.button:hover{
   background: transparent;
   color: black;
   border: 1 px solid black;
}
```

Conforme a figura seguinte ilustra, o browser apresenta o botão "enviar" com o fundo a preto, as letras a branco, e os cantos arredondados.

Quando o cursor passar por cima do Botão, o fundo do botão passa a "transparent" e o texto do botão a "Black".



4.2. CSS - Exemplo II

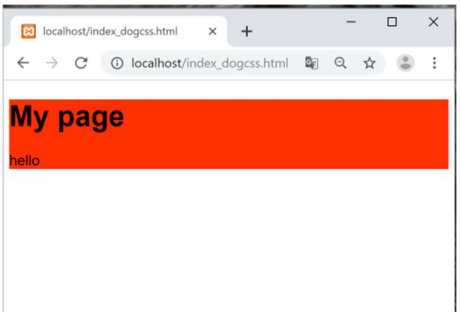
Outro exemplo com dois ficheiros, um ficheiro HTML e um ficheiro CSS.
Neste exemplo, o texto relativo à divisão aparece formatado de acordo com o documento CSS.

-----Ficheiro HTML ----- ElementoHeaderDiv.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset=\"UTF-8\">
      <link rel="stylesheet" href="elementoheaderdiv.css">
</head>

<body>
        <header class="main-header" id= "headerMain">

                <div class="header">
                        <h1>My page</h1>
                        <div id="box-1">hello</div>
                </div>
        </header>
</body>
</html>
```

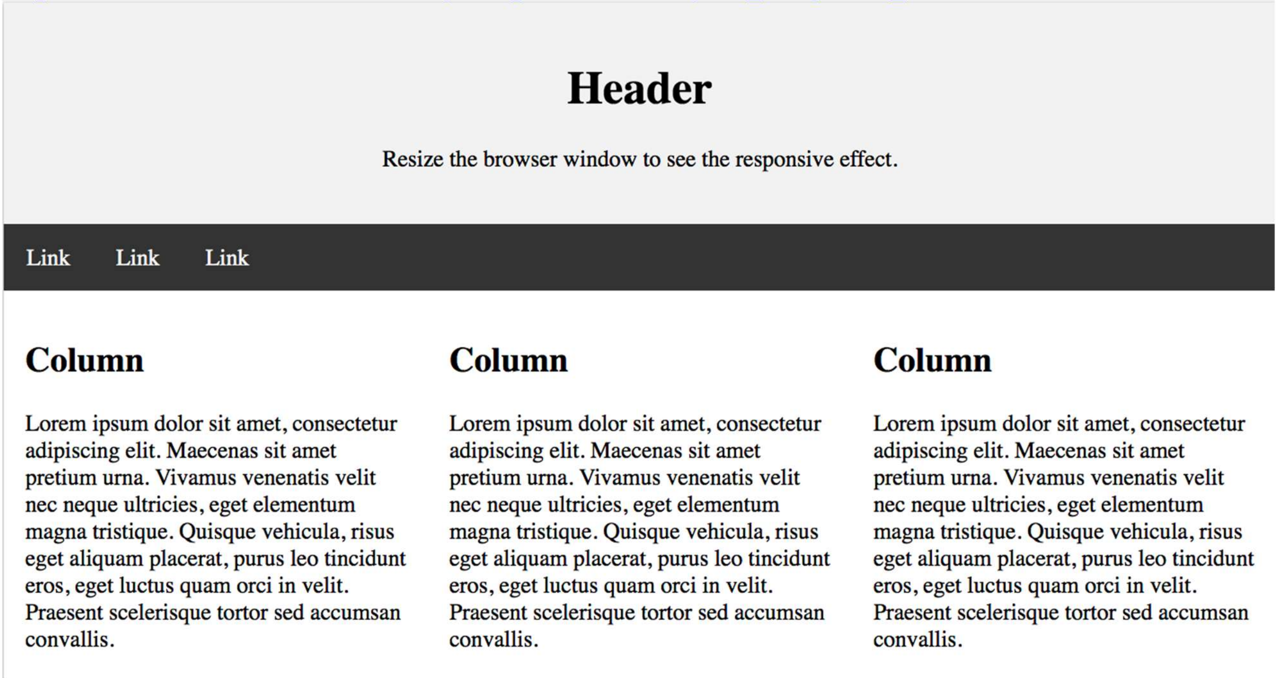
-----Ficheiro CSS, com o nome "ElementoHeaderDiv.css" -----

```
body{
font-family: sans-serif;
}

.header{
   background-color: red;
   padding: 20 px
}
```


localhost/index_dogcss.html
localhost/index_dogcss.html
My page
hello

4.3. CCS Layout - Exemplo III

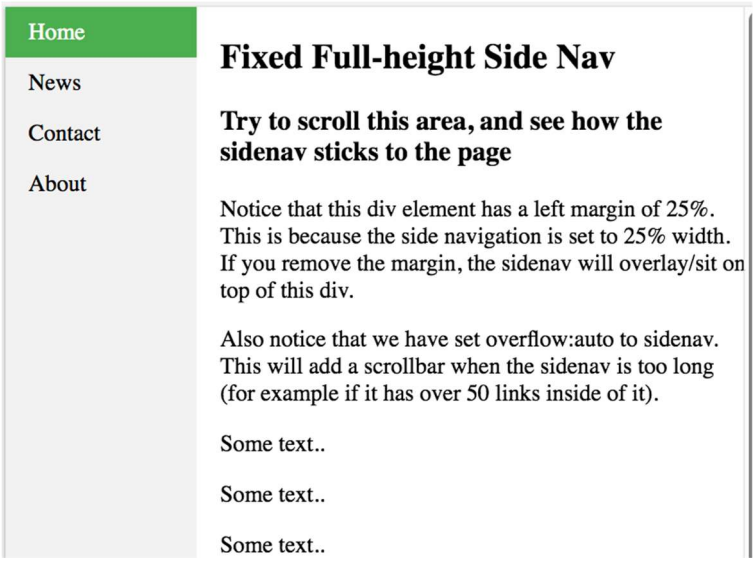
https://www.w3schools.com/css/css_website_layout.asp
https://www.w3schools.com/css/tryit.asp?filename=trycss_template1_float




```
<!-- Ficheiro   Layout_Inline_Css.html -->

<!DOCTYPE html>
<html lang="en">
<head>
<title>CSS Website Layout</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
        * {
          box-sizing: border-box;
        }

        body {
          margin: 0;
        }

        /* Style the header */
        .header {
          background-color: #f1f1f1;
          padding: 20px;
          text-align: center;
        }

        /* Style the top navigation bar */
        .topnav {
          overflow: hidden;
          background-color: #333;
        }

        /* Style the topnav links */
        .topnav a {
          float: left;
          display: block;
          color: #f2f2f2;
          text-align: center;
```

```
    padding: 14px 16px;
    text-decoration: none;
  }

  /* Change color on hover */
  .topnav a:hover {
    background-color: #ddd;
    color: black;
  }

  /* Create three equal columns that floats next to each other */
  .column {
    float: left;
    width: 33.33%;
    padding: 15px;
  }

  /* Clear floats after the columns */
  .row:after {
    content: "";
    display: table;
    clear: both;
  }

  /* Responsive layout - makes the three columns stack on top of each other instead of next to each other */
  @media screen and (max-width:600px) {
    .column {
      width: 100%;
    }
  }
</style>
</head>
<body>

<div class="header">
  <h1>Header</h1>
  <p>Resize the browser window to see the responsive effect.</p>
</div>

<div class="topnav">
  <a href="#">Link</a>
  <a href="#">Link</a>
  <a href="#">Link</a>
</div>

<div class="row">
      <div class="column">
      <h2>Column</h2>
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit..</p>
      </div>

      <div class="column">
      <h2>Column</h2>
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
      </div>

      <div class="column">
      <h2>Column</h2>
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
      </div>
</div>
</body>
</html>
```

4.4. CCS Navigation bar - Exemplo IV



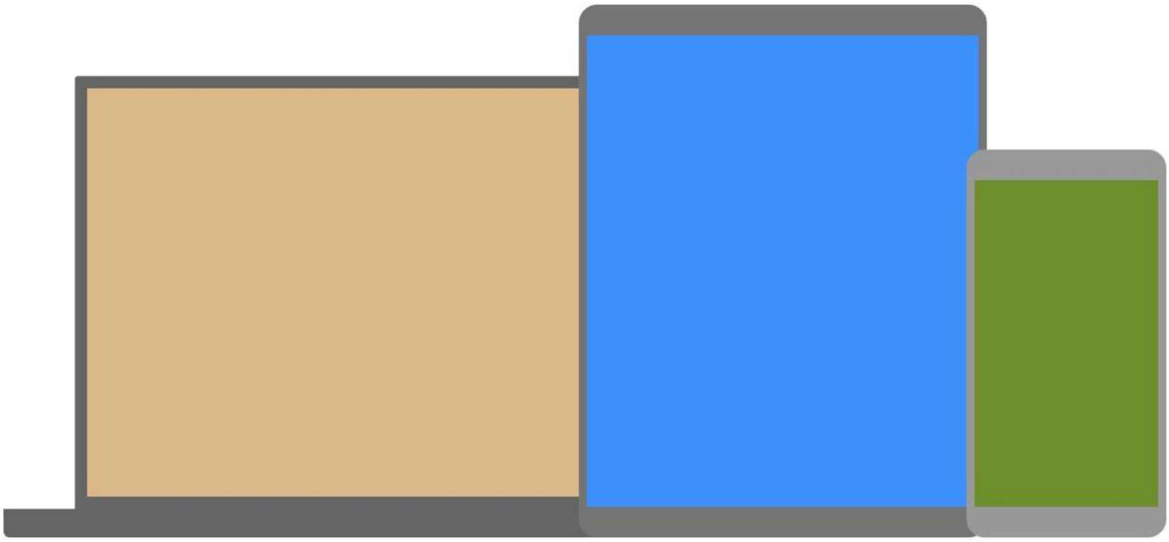
<!-- Ficheiro: Hmtl_InLine_Css.html-->

```
<!DOCTYPE html>
<html>
<head>

<style>
        body {
          margin: 0;
        }

        ul {
          list-style-type: none;
          margin: 0;
          padding: 0;
          width: 25%;
          background-color: #f1f1f1;
          position: fixed;
          height: 100%;
          overflow: auto;
        }

        li a {
          display: block;
          color: #000;
          padding: 8px 16px;
          text-decoration: none;
        }

        li a.active {
          background-color: #4CAF50;
          color: white;
        }

        li a:hover:not(.active) {
          background-color: #555;
          color: white;
        }
        </style>
```

```
</head>
<body>

<ul>
  <li><a class="active" href="#home">Home</a></li>
  <li><a href="#news">News</a></li>
  <li><a href="#contact">Contact</a></li>
  <li><a href="#about">About</a></li>
</ul>

<div style="margin-left:25%;padding:1px 16px;height:1000px;">
        <h2>Fixed Full-height Side Nav</h2>
        <h3>Try to scroll this area, and see how the sidenav sticks to the page</h3>
        <p>Notice that this div element has a left margin of 25%. This is because the side navigation is set to 25% width.
       If you remove the margin, the sidenav will overlay/sit on top of this div.</p>
        <p>Also notice that we have set overflow:auto to sidenav. This will add a scrollbar when the sidenav is too long
       (for example if it has over 50 links inside of it).</p>
        <p>Some text..</p>
        <p>Some text..</p>
        <p>Some text..</p>
        <p>Some text..</p>
        <p>Some text..</p>
        <p>Some text..</p>
        <p>Some text..</p>
</div>

</body>
</html>
```

4.5. CCS Media queries - Exemplo V

https://www.w3schools.com/css/css3_mediaqueries_ex.asp

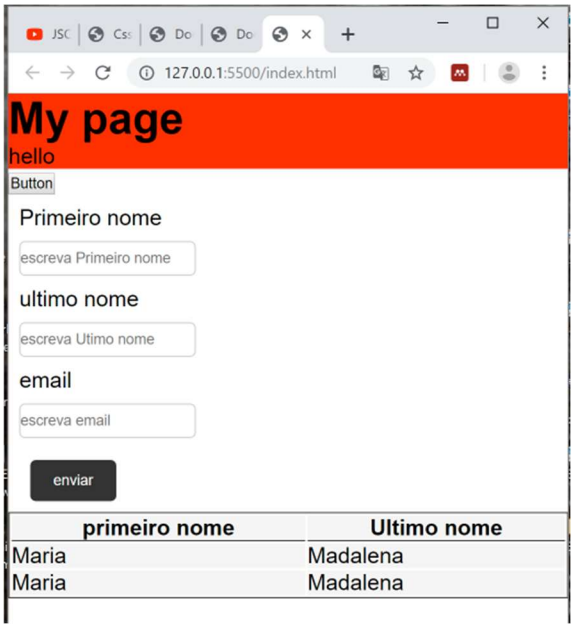


```
/* Set the background color of body to tan */
body {
  background-color: tan;
}

/* On screens that are 992px or less, set the background color to blue */
@media screen and (max-width: 992px) {
  body {
    background-color: blue;
  }
}

/* On screens that are 600px or less, set the background color to olive */
@media screen and (max-width: 600px) {
  body {
    background-color: olive;
  }
}
```

4.6. CSS - Exemplo VI

CSS Tutorial for Beginners - part 1 of 4 - Applying Styles
https://www.youtube.com/watch?v=Wz2klMXDqF4&list=PL07598CCC0961C10C

Intro to HTML & CSS - Tutorial
https://www.youtube.com/watch?v=kLO4X_3VYdg (1h33)
(- instalar code.visualstudio.com - Instalar liveServer)



-----Ficheiro HTML -----

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <meta http-equiv="X-UA-Compatible" content="ie=edge">
      <title>Css teste</title>
      <link rel="stylesheet" href="style.css">
</head>

<body>
<header class="main-header" id= "headerMain">

   <div class="header">
      <h1>My page</h1>
      <div id="box-1">hello</div>
   </div>

   <section>
      <button>Button</button>

<form>
<label for="primeironome">Primeiro nome</label><br>
<input type="text" placeholder="escreva Primeiro nome"><br>
<label for="ultimonome">ultimo nome</label><br>
<input type="text" placeholder="escreva Utimo nome"><br>
<label for="email">email</label><br>
<input type="text" placeholder="escreva email"><br>
```

```
<input type="submit" value="enviar" class="button"> <br>
</form>

<table>
<thead>
<tr><th>primeiro nome</th><th>Ultimo nome</th></tr>
</thead>
<tbody>
   <tr><td> Maria</td><td> Madalena</td></tr>
   <tr><td> Maria</td><td> Madalena</td></tr>
</tbody>
</table>

   </section>
</header>
</body>
</html>
```

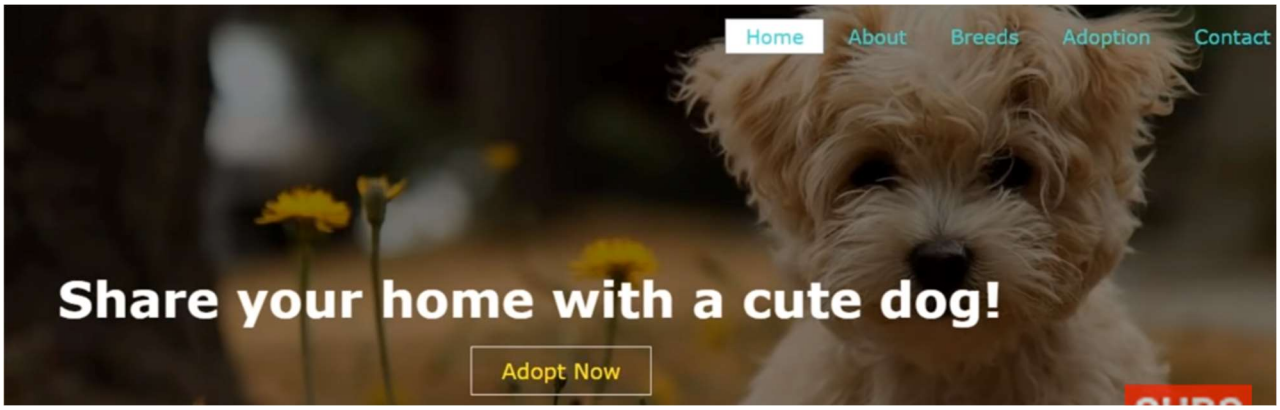
-----Ficheiro CSS -----

```
*{
   margin:0;
   padding:0;
}

/*#box-1 {
   background-color: red;
   margin-top: 10px;
   margin-left: 10px;
}*/

body{
font-family: sans-serif;
}

.header{
   background-color: red;
   padding: 20 px
}

.button{
   margin-top: 10px;
   margin-left: 10px;
   padding: 10px 20px;
   border: 1px solid transparent;
   background: #333;
   color: white;
   border-radius: 5px;
   cursor: pointer;
   transition: all 1s;
}

.button:hover{
   background: transparent;
   color: black;
   border: 1 px solid black;
}

form{
margin: 10px;
margin-top:20 px;
}
```

```
form label{
    display : blocç;
    margin : 5px 0;
}

form input[type="text"],
input[type="email"]{
    padding: 7px 0;
    margin: 10px 0;
    border-radius: 5px;
    border: 1px solid #ccc;
}

table{
width: 100%;
border : 1px solid #333;
margin: auto

}

table td, th{
    background-color: #f4f4f4;
}

table th{
border-bottom: 1px solid #333;

}
```

4.7. CSS Exemplo VII

Create a Website using HTML and CSS Under 30 Minutes | HTML Tutorial | CSS Tutorial (23 min)
https://www.youtube.com/watch?v=Tm3pEiRax00
(explica bem o HTML e o CSS)



-----Ficheiro HTML -----

```
<!DOCTYPE html>
<html lang="en">

<head>
        <title> titulo</title>
        <link href="dog.css" rel="stylesheet" type="text/css" >
</head>

<body>
<header>

        <div class="top">
                <ul class ="nav-menu">
                        <li class="homebtn"><a href=""> Home</a></li>
                        <li><a href=""> About</a></li>
                        <li><a href=""> Breeds</a></li>
                        <li><a href=""> Adoption</a></li>
                        <li><a href=""> Contacts </a></li>
                </ul>
        </div>

        <div class="tagline">
                <h1>Share your </h1>
                <div class="adopt">
                <a href="" class="bttn">Adopt now</a>
                </div>
        </div>

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

---- Ficheiro CSS - “dog.css”---

```
*
{
margin:0;
padding:0;
}

header{
background-image: linear-gradient(rgba(0,0,0,0.5),rgba(0,0,0,0.5)),url('dog.jpg');
height:100px;
background-size: cover;
background-position: center;
}

.nav-menu{
float: right;
list-style: none;
margin-top:30px;

}

.nav-menu li{
display: inline-block;

}

.nav-menu li a{
    color: turquoise;
    padding: 5px 20px;
    font-family:"Verdana","sans-serif";
    font-size:20px;
}


.nav-menu li a:hover{
    border: 1px solid grey;
    background-color: white
}

.tagline{
    position: absolute;
    width:1200px;
    margin-left: 0;
    margin-top: 0;
}

h1{
    color: black;
    font-size:50px;
    font-family: "verdana","sans-serif";
    text-align: center;
```

```
    margin-top: 275px;
}

.adopt{
margin-top: 100 px;
margin-left: 540px;
}

.adopt a:hover{
   background-color: burlywood;
}

.bttn{
    border: 1px solid white;
    padding: 10px 30px;
    color: blue;
    font-family: "Verdana";
    font-size: 22px;

}
```

Trab2 Módulo IOT como servidor TCP/IP c/ pág HTML para controlo de saída digital

Pretende-se ter um só botão na página WEB que permita ligar e desligar o led azul pequeno (como fizemos na aula) mas agora o botão formatado com CSS. (Neste trabalho pretende-se criar uma página HTML + CSS "dentro" do ESP.*)*

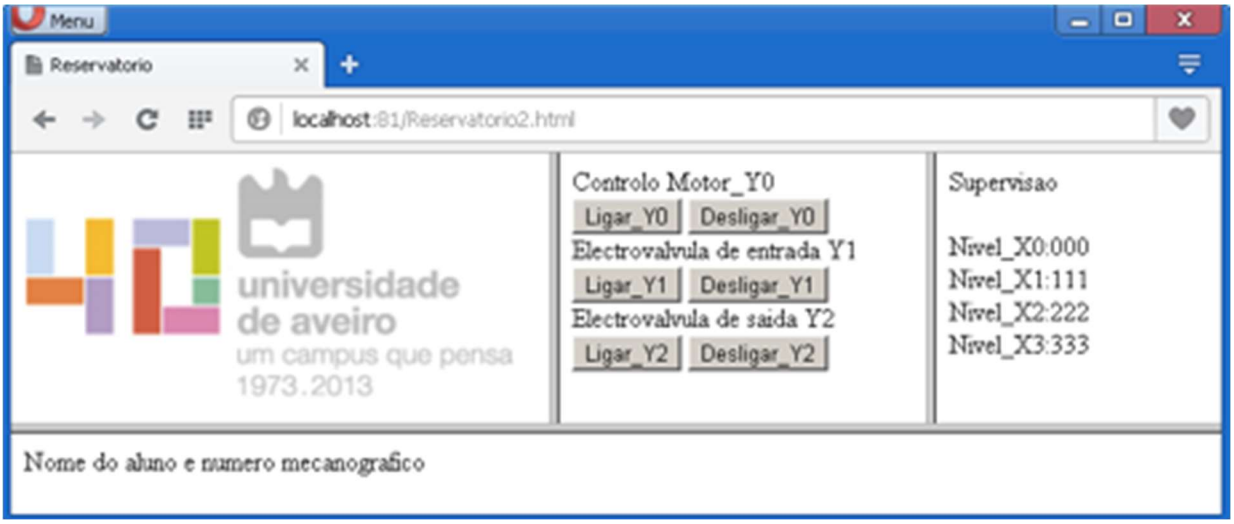
– Módulo IOT Wifi Servidor
Com o Módulo deve criar um servidor HTTP, enviar páginas HTML e CSS, por HTTP, para o telemóvel, permitindo monitorizar no Browser do telemóvel as entradas e saídas digitais e analógicas do módulo para detectar o nível e água. *Utilize um sensor de nível analógico para adquirir também o nível de água..*

Bibliografia:
Além dos apontamentos, vejam este youtube: Como Fazer um WebServer com ESP8266 sendo Controlado pela Web - Video #4 - ESP8266 Primeiros Passos
https://www.youtube.com/watch?v=woPFuwvHPoA&list=PL7CjOZ3q8fMe6DxojEFuDx4BP0qbbpKtP&index=4

– Web Front end - HTML e CSS
desenvolva a páginas Web de supervisão e controlo do reservatório (com as várias frames e CSS)

Recordam-se da interface que desenvolvemos em Informática Industrial para controlar o reservatório ?!

Não necessita de ser tão elaborada:



1º objetivo: ter um só botão na página WEB que permita ligar e desligar o led azul pequeno (como fizemos na aula) mas agora o botão formatado com CSS. (Neste trabalho pretende-se criar uma página HTML + CSS "dentro" do ESP.)

2º objetivo: além do botão que controla o led azul, receber no browser o estado de outros pinos do ESP (n1, n2 e do próprio led azul)

O Browser WEB pede páginas HTML ao ESP, via HTTP/TCP/IP/Wifi 802.11

Neste exemplo são utilizados: um ESP, um Router local e um computador com um browser.
Com este código exemplo, o ESP estabelece uma ligação Wifi com o Router local, e atua como um servidor TCP/IP na porta 80. Depois de estar a correr, o ESP aceita pedidos de ligação TCP/IP de um browser (de uma máquina remota) que esteja ligada ao mesmo Router local e dessa forma ao ESP.

O router tem o SSID = default e não tem password.

Neste exemplo são usados três objetos, um de cada tipo WiFi, WiFiServer , e WiFiClient

Objeto do tipo WiFi

.begin (pede ligação wifi ao router)
.status (estado da ligação Wifi)
.localIP (retorna o end IP atribuído pelo Router ao ESP)

Objeto do tipo WiFiServer

.begin (fica à escuta de um pedido de ligação TCP/IP)
.available (retorna a referencia da nova ligação)

Objeto do tipo WiFiClient

.available (devolve nº bytes recebidos)
.readString (lê vários bytes)
.flush (limpa a msg recebida)
.print (envia uma string para o cliente, via TCP/IP)

https://arduino-esp8266.readthedocs.io/en/latest/esp8266wifi/client-class.html

Nos dois exemplos seguintes, o ESP devolve uma página HTML ao Browser, nas no exemplo
a) a página HTML é estática
b) a página altera-se em função do estado da saída digita nº2 (Led azul pequeno) do ESP.
Assumindo que o IP atribuído ao ESP é o 192.168.43.49
no browser deverá aceder a http://192.168.43.49/GPIO=1 ou http://192.168.43.49/GPIO=0

Exemplo 2a) ESP como servidor WEB, envia uma página HTML estática para o Browser

```
#include <ESP8266WiFi.h>

WiFiServer server(80);

void setup() {
        // Inicia o objecto Rs232
        Serial.begin(9600);

        // Pede ligação WiFi, ao Router
        WiFi.begin("NOS-88C0","armagedao202");

        // espera que o Router aceite a ligação WiFi
        while(WiFi.status() != WL_CONNECTED ) {
                delay(500);  Serial.print(".");
                }

        Serial.println("ESP connecto to router , by Wifi");

        // ativa o servidor TCP/IP
        server.begin();
        Serial.println("Servidor TCPIP ativo");

        //  IP address
        Serial.println(WiFi.localIP());
        }

void loop() {
  // Espera que o Browser peça ligação TCP/IP
  WiFiClient client = server.available();

  if (client) {
      // Espera que o Browser envie dados , ex. GET index.html
      Serial.println("new client\r\n");
      while(!client.available()){ delay(1); }
      Serial.println("Mensagem HTTP + HTML enviada PELO Browser");

      String ss = client.readStringUntil('\r'); //String ss = client.readString();
      Serial.println(ss);
      client.flush();

      // Send the response to the client/Browser
      client.print("HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE HTML>\r\n<html>\r\n");
      client.print("Hello");
      client.print("</html>\n");

      // Debug Send the response to Rs232
      Serial.println("Mensagem HTTP + HTML enviada PARA o Browser");
      Serial.print("HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE HTML>\r\n<html>\r\n");
      Serial.print("Hello");
      Serial.print("</html>\n");

    } // if(client)
} // Loop
```

Exemplo 2b) ESP como servidor WEB, o Browser controla saída digital do ESP

```
#include <ESP8266WiFi.h>
WiFiServer server(80);
int val;

void setup() {
// Inicia o objecto Rs232
Serial.begin(9600);
// Pede ligação WiFi, ao Router
WiFi.begin("NOS-88C0","armagedao202");

// espera que o Router aceite a ligação WiFi
while(WiFi.status() != WL_CONNECTED ) { delay(500);  Serial.print("."); }
Serial.println("ESP connecto to router , by Wifi");

// ativa o servidor TCP/IP
server.begin();
Serial.println("Servidor TCPIP ativo");

//  IP address
Serial.println(WiFi.localIP());
pinMode(2,OUTPUT);
}

void loop() {
  // Espera que o Browser peça ligação TCP/IP
  WiFiClient client = server.available();
  if (client) {  // Se chegou uma ligação TCPIP
       // Espera que o Browser envie dados , ex. GET index.html
       Serial.println("nova ligação TCPIP \r\n");
       while(!client.available()){ delay(1); }
       Serial.println("Mensagem HTTP + HTML enviada PELO Browser");
       // Lê a mensagem recebida pelo ESP, enviada pelo Browser
       String ss = client.readStringUntil('\r'); //String ss = client.readString();
       Serial.println(ss);
       client.flush();
       // Processa mesg recebida pelo ESP
        if (ss.indexOf("GPIO=1") >0 ){val = 1;}
        if (ss.indexOf("GPIO=0") >0 ){val = 0;}
        digitalWrite(2,val);

       // Envia pág HTML para o Browser
       client.print("HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE HTML>\r\n<html>\r\n");
            if(val==1){
                        client.print("GPIO=1");
                        }else{
                        client.print("GPIO=0");
                        }
       client.print("</html>\n");

       // Debug Send the response to Rs232
       Serial.println("Mensagem HTTP + HTML enviada PARA o Browser");
       Serial.print("HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE HTML>\r\n<html>\r\n");
       if(val==1){
          Serial.print("GPIO is 1");
          }else{
          Serial.print("GPIO is 0");
          }
       Serial.print("</html>\n");

   } // if(client)
} // Loop
```