# HTML tutorial

## General elements

**<!doctype html>**

**<html></html>:** We create the whole between this markers.

|  |  |
| --- | --- |
| **<head>**  **<title>** Me encanta el código libre **</title>**  **</head>** |  |

**<body></body>:** In this part we write all the page information.

**<div id =** “name”**></div>:** Allows to divide the body in parts and use css classes.

|  |  |
| --- | --- |
| **<h1>** The Fish Web Site **</h1>** |  |
| **<h2>** This is a section on Bass fish **</h2>** |  |

## Escape characters

**<!--** comment -->**:** We can add comments in our web page. Otherwise

**<p></p>:** We use this tab to create paragraphs in our web page.

**<span class=**”name”**> </span>:** Allows to divide a paragraph an apply a class to a part of paragraph.

**<blockquote></blockquote>:** Indent the text that we write.

**<br>:** Creates a paragraph break.

**&nbsp**: We can use this to write many white spaces between to words.

**™ ---------- &#8482;**

**& ---------- &amp;**

**© ---------- &copy;**

**< ----------- &lt;**

**> ----------- &gt;**

**? ----------- &#63;**

## Bold, italic, sup and sub elements

**strong** and **em** tabs make different emphasis if a program is reading the text but visually are the same.

|  |  |
| --- | --- |
| <p> This is a **<b>**bold element**</b>**. </p>  <p> This is a **<strong>**strong element**</strong>**. </p>  <br>  <p> This is a **<i>**italic element**</i>.** </p>  <p> This is a **<em>**em element**</em>**. </p>  <br>  <p> The expert said a lot on August 23**<sup>**rd**</sup>**. </p>  <p> There is plenty of H**<sub>**2**</sub>**0 in the lake. </p> |  |

## Meta elements

**<head></head>**

* ***This part is going to appear in the text of Google and should have lest of 155 characters.***

**<meta name** ="description" **content=**"This is a web page about fish. We can cover all the fish in the world."**/>**

**<meta name =**"keywords" **content=**"fish, ocean, aqarurium"**/>**

* ***Here we can add more information for our search engine.***

**<meta http-equiv =**"author" **content=**"Joe Doe"**/>**

**<meta http-equiv =**"content-language" **content=**"en-us"**/>**

## List elements

|  |  |
| --- | --- |
| Unordered list  **<ul>**  **<li>** CD **</li>**  **<li>** DVD **</li>**  **<li>** Blu Ray **</li>**  **</ul>** |  |
| Ordered list  **<ol>**  **<li>** CD **</li>**  **<li>** DVD **</li>**  **<li>** Blu Ray **</li>**  **</ol>** |  |

## Table elements

**<table></table>:** Defines a whole table.

**<tr></tr>:** Creates a row.

**<th></th>:** Defines each the column.

**<td></td>:** Store the data in the same order defined in columns

## Creating links

**<a href=**"url"**> </a>:**  Creates text with links.

**<p></p>**

**<a href=**"url"**>**text **</a>**

**<ul></ul>**

**<li></li>**

**<a href=**"url"**>**text **</a>**

There two types of links:

* url = “www.algo.com” (external link )

**<p> <a href=**"https://twitter.com/Angel\_Feliz\_F"**>** Mi twitter **</a></p>**

* url = “#SomeName”

**<p>** Go to bottom with this **<a href=**"#Compra"**>** link **</a> </p>**

**<p> <a name=**"Compra"**>** The is the end **</a> </p>**

## Inserting images

Title attribute creates a tooltip over a picture of any object.

**<p></p>**

**<img src=**"BOOKDOWN.png" ***width =*** "700" ***height =*** "700"

**title=**"julius caesar" **alt=**"Image failed to load"**>**

## Inserting audio or video

Three standard formats for web browsers are .mp3 .wav and .ogg

Three standard formats for web browsers are mp4, webm and .ogg

**<body></body>**

**<audio controls *=*** *"audiocontrols"* autoplay loop ***draggable=***"true"**>**

**<source src=**"17 Verte Sonreir (Prod. Ockrams).mp3"/>

**<source src=**"test1.wav"**/>**

**</audio>**

**<video width=**"640" **height=**"360" **controls>**

**<source src=**"videotest.mp4"**/>**

**<source src=**"videotest.ogg"**/>**

**</video>**

## Form elements

### Fieldbox and title of form (legend)

**<body></body>**

**<form>**

**<fieldset>**

**<legend>**Enter day you want to travel to Cancun:**</legend>**

*<!-- Here we write all the inputs of the form -->*

**</fieldset>**

**</form>**

### Short text box

**<body></body>**

**<form action=**"" **method=**""**>**

**<label for=**"idtext"**>** Text out **</label>**

**<input type=**"text" **id =** "idtext" **name=**"" ***size =*** "50"**>**

**<br>**

**<input type=**"number" **min =**"0" **max=**"50" **>**

**<input type=**"date" **name=** "" **>**

Number of days: **<input type=**"number" **name=**"" ***min=***"1"***max=***"10"**>**

**</form>**

### Large text box and submit bottom

**<body></body>**

**<form action="" method="">**

**Comments: </br>**

**<textarea rows=**"30" **cols=**"80"**></textarea></br>**

**<input type="submit" value=**"Submit"**>**

**</form>**

### One answer selection

**<body></body>**

**<form action="" method="">**

**Yes: <input type=**"radio" **name=**"yesorno" **value="">** **</br>**

**No: <input type=**"radio" **name=**"yesorno" **value="">**

**</form>**

### Check box

**<body></body>**

**<form action="" method="">**

Which subjects are you interested in? **<br>**

**<input type="checkbox" name=""** checked="checked"**>** Football **<br>**

**<input type="checkbox" name="">** Basketball **<br>**

**<input type="checkbox" name="">** Baseball **<br>**

**<input type="checkbox" name="">** Hockey

**</form>**

### Drop-down list

**<body></body>**

**<form action="" method="">**

**<select>**

**<option value=**"Michigan"**>**Michigan**</option>**

**<option value=**"Alabama"**>**Alabama**</option>**

**<option value=**"North Dakota"**>**North Dakota**</option>**

**</select>**

**</form>**

## iframe element (external page)

This part creates a box where we can store many objects and even change the web page with the tab a. We can link objects like:

* Advertences
* Navigation menus
* Images
* External web pages

**<iframe frameborder=**"1" **width=**"200" **height=**"200"

**src=**"first.html" **name=**"ouriframe" **id=**"ouriframe"**>**

**</iframe>**

**<br/>**

**<a href=**"second.html" **target=**"ouriframe"**>** switch to new document **</a>**

https://developer.mozilla.org/en-US/docs/Web/HTML

# CSS tutorial

<https://htmlcheatsheet.com/css/>

**<head></head>**

**<style></style>**

OR

**<head>**

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

**</head>**

OR

**<p style=**"color:black;front-size:25px"**>** This is inline style. **</p>**

## Colors

|  |  |
| --- | --- |
|  |  |

## Styling a tag globally

**body{**

color: green;

background-color: *lime | rgb(240,28,235) | #ffcf79*;

**}**

**h1, h2 {**

background-color: purple;

border: 1px solid black;

border-bottom: 1px solid black;

text-decoration: *overline* ***|*** *underline | line-through | none;*

text-align: *center | right | left;*

font-weight: *lighter | normal | bold | bolder;*

font-style: *italic | oblique;*

**}**

**h1{**color: pink;**}**

**p {**

color: red;

font-size: 25px;

font-family: Helvetica;

**}**

## Styling a tag by class

### Creating specific classes

If we have <p class = "name"></p>

**<head></head>**

**<style></style>**

**p.name {**color: blue;**}**

### Creating specific ID attributes

If we have <p id = "namepage"></p> is perfect for unique elements.

**<head></head>**

**<style></style>**

**#**namepage**{**color: blue;**}**

### Creating general classes

**<head></head>**

**<style></style>**

**.blue {**color: blue;**}**

**.underline {**text-decoration: underline;**}**

Now we can combine classes

**<blockquote class =** "blue underline"**></blockquote>**

## Relative font-size and line height

body{font-size: 25px}

h1{font-size: 150%} so the px will be 25\*150/100 = 37.5px

h2{font-size: 1.2em} so the px will be 25\*1.2 = 30px

p {line-height: 1.5em;}

## CSS Box Model

We can write in every class these properties:

Graphical user interface, application

Description automatically generated

| **margin:** Space between one tag and other tag. | **Border:** Shows lines surrounding the content. | **padding:** Space between the border and content.  background-color works. |
| --- | --- | --- |
| margin: 20px | **0** | border-color: back; | padding: 20px |
| margin-left, margin-right | **auto** (to center) | border-width: 2px  *thin | medium | thick* | padding-left, padding-right |
| margin-top, margin-bottom | border-style: solid;  *dashed | dotted | double | groove | ridge | inset | outset* | padding-top, padding-bottom |
|  | WE CAN COMBINE THE 3 TOP with:  padding-left, padding-right, padding-top, padding-bottom |  |
|  | border-radius: 10px |  |

Text

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
|  | border-color | border-width | border-style |
| **border:** | back | 2px | solid |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | padding-top | padding-right | padding-bottom | padding-left |
| **padding:** | 20px | 20px | 20px | 20px |

## Special effects

### Links

a:link{color:blue;}

a:visited {color: #8c8c9d;}

a:hover{color: #f15a51;}

# Web scraping

## Importing page

**minimal\_html(**"html code"**):** Lets you create an xml\_document from literal HTML.

**read\_html(**"url"**):** This returns a xml\_document2 object which you’ll then manipulate using rvest functions.

Logo

Description automatically generated with low confidence

Text

Description automatically generated

|  |  |
| --- | --- |
| Graphical user interface, text, application  Description automatically generated |  |

Graphical user interface, text, application

Description automatically generated

## Filtering the page

### rvest functions

html %>% **html\_children("**tag"**):** Selects all the children of a tag.

html %>%**html\_element(**"tag" or ".class\_name " or "#tag\_id"**):** Can select the elements of the first matching tag, class\_name or tag\_id. When applied to a node set (tag), always returns a vector the same length as the input, using a ***"missing"*** element where needed.

html %>% html\_element(‘ccs\_selector1, ccs\_selector2’ or ‘\*’)

html %>% **html\_elements(**"tag" or ".class\_name" or ".tag\_id"**):** Can select the elements of all matching tags, class\_names or tag\_ids. When applied to a node set (tag), returns all matching elements beneath any of the inputs, flattening results into a new node set.

### CSS selectors and combinators

|  |  |
| --- | --- |
| Text  Description automatically generated | Table  Description automatically generated  p:last-child  , == OR  .x.y == .x Y .y |

**space** **(descendant combinator**)**:** selects all the descendants of a certain HTML element, whether they are direct descendants or elements further down the tree.

Graphical user interface, text, application

Description automatically generated

**> (child combinator):** It only selects direct descendants of a parent element.

Graphical user interface, text, application, website

Description automatically generated

**+ (adjacent sibling combinator):** We can used to reference to the second sibling of a tag.

Graphical user interface, text, application

Description automatically generated

**~ (general sibling combinator):** We can used to reference to all siblings of a tag.

Graphical user interface, text, application

Description automatically generated

**\*:** Allow us to get all the siblings after div.first tag.

Graphical user interface, application, website

Description automatically generated

### XPATH

|  |  |
| --- | --- |
| **XPATH:** '*//***p**' | **XPATH:** '//**div**/**p**' |
| **meaning:** Selects all **p** tags | **meaning:** From all **div** selects direct child **p** |
|  |  |
| **XPATH:** ' //**div**[**p**[@class=**"third"**]] | **XPATH:** ' //**\***[@id=**"third"**]//**p**' |
| **meaning:** Select all **div** with **p** with class **third** | **meaning:** From all **tags** with id **third** select child **p** |
|  |  |
| **XPATH:** '//**div**[position() = 2]/**\***[position() >= 2]' | |
| **meaning:** From all **div** in second place selects direct child **tags** in second or grader place | |
|  | |
| **XPATH:** ''//**div**[count(**h2**) = 1 and count(**p**) > 1]'' | |
| **meaning:** Select all **div** with one **h2** and more than one **p** | |
|  | |
| **XPATH:** programming\_html %>%  html\_elements(xpath = '//**li**//**em**[text() = "twice"]') %>%  html\_elements(xpath = '..') | |
| **meaning:** From all **li** select child **em** with “twice” text inside then select parent | |
|  | |

## Extracting information

html %>% **html\_text()**: Shows the text as it was written in the source code.

html %>% **html\_text2()**: Shows the text in the way it would be displayed in the browser.

html %>% **html\_attr():** Gets a single attribute.

html %>% html\_elements("a") %>% html\_attr("href")

html %>% html\_elements("img") %>% html\_attr("src")

html %>% **html\_attrs():** Gets all attributes.

html %>% html\_node("table") %>% **html\_table()**: If we found a table tag we can extract its info.

* If the table has the *<th></th>* we should set **header=TRUE**

## Best practice

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated