



Frontend

HTML

# What's HTML?

Hyper Text Markup Language

# HTML basics

Tags

Attributes



# Tags

[Full tag list](#)

doctype, html, head, title, base, link, meta, style, script, noscript, body, article, nav, aside, section, header, footer, h1-h6, main, address, p, hr, pre, blockquote, ol, ul, li, dl, dt, dd, figure, figcaption, div, table, caption, thead, tbody, tfoot, tr, th, td, col, colgroup, form, fieldset, legend, label, input, button, select, datalist, optgroup, option, textarea, keygen, output, progress, meter, details, summary, command, menu, del, ins, img, iframe, embed, object, param, video, audio, source, canvas, track, map, area, a, em, strong, i, b, u, s, small, abbr, q, cite, dfn, sub, sup, time, code, kbd, samp, var, mark, bdi, bdo, ruby, rt, rp, span, br, wbr ...

# Attributes

```

```

```
<h1 id="header">Hello, world!</h1>
```

```
<div class="container">
```

```
  <p>I can be styled using CSS!</p>
```

```
</div>
```

# Tags + Attributes

- Load assets like images, css, javascript...
- Wrap and structure content
- Add meaning to the wrapped content
- Allow CSS styling
- Communicates browsers explicit features of your page.

# Comments

```
<!-- I'm an HTML comment -->
```

```
<!-- Use me with caution -->
```



# Basic HTML file

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>Page title</title>
  </head>
  <body>
    
    <h1 class="hello-world">Hello, world!</h1>
  </body>
</html>
```

# Basic HTML file breakdown

```
<!DOCTYPE html>
```

Enforce standards mode and more consistent rendering in every browser possible.

# Basic HTML file breakdown

```
<html lang="en">  
  <head>  
    <!-- Non graphically represented stuff -->  
  </head>  
  <body>  
    <!-- Your awesome content -->  
  </body>  
</html>
```

The `<html>` tag declares the start of the document, the `<head>` is mostly used to call external css files, and declare properties of the current page. The `<body>` contains everything represented in the viewport.

# Basic HTML file breakdown

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

A lang attribute on the root html element defines the document language. Mostly for accessibility (a11y) reasons.

[Full list of HTML languages.](#)

# Basic HTML file breakdown

```
<meta charset="UTF-8">
```

Ensure proper rendering of your content by declaring an explicit character encoding.

# Basic HTML file breakdown

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

Put this inside your <head>, it'll be useful in a while.

# Basic HTML file breakdown

```

```

```
<h1 class="hello-world">Hello, world!</h1>
```

The `<img>` tag represents an image and the `<h1>` is a header of level 1 (there's six levels of headers).

# CodePen

Frontend playground

[Check it out](#)





# Basic HTML file

[Live example](#)

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>Page title</title>
  </head>
  <body>
    
    <h1 class="hello-world">Hello, world!</h1>
  </body>
</html>
```

# Emmet

Front-end code helper for Sublime Text, Atom, etc.

[Get it](#)



# Exercise

Create a basic skeleton HTML file:

1. Name it so it's accessible from the root domain.
2. Place the text 'Hello world!' inside some element of your choice.
3. Congrats you're a front-end developer.

# HTML forms

# Basic HTML form

[Live example](#)

```
<form>
  <div>
    <label for="hi">Say hi</label>
    <input type="text" name="hi" id="hi">
  </div>
  <button type="submit">Send</button>
</form>
```

# Exercise

Create a login form, it should include:

1. An email or username field.
2. A password field.
3. A checkbox to remember my credentials.
4. A button to log in.

# HTML best practices

# CSS includes

[Live example](#)

```
<!-- External CSS -->
```

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

```
<!-- In-document CSS -->
```

```
<style>
```

```
  /* ... */
```

```
</style>
```

The CSS is included inside the <head> tag.



# JavaScript includes

```
<!-- External JavaScript -->  
<script src="app.js"></script>
```

The JS files should be appended just before the closing of the `</body>` tag.

# Use double quotes on attributes

```
<!-- Don't, just don't -->  
<input type='file' name='file'>
```

```
<!-- Much better -->  
<input type="file" name="file">
```

It might mess with your text editor's capability to highlight attribute values.

# Avoid the trailing slash in self-closing elements

```
<!-- Do -->
```

```

```

```
<!-- Don't -->
```

```

```

The HTML5 specification says they're optional.

# Boolean attributes

[Live example](#)

```
<input type="text" disabled>
```

```
<input type="checkbox" value="1" checked>
```

```
<select>
```

```
  <option value="1" selected>1</option>
```

```
</select>
```

The absence of the boolean attribute already represents the value as false, it's presence represents the true value.

# Reduce markup

```
<!-- Not so great -->  
<span class="avatar">  
    
</span>
```

```
<!-- Better -->  

```

Avoid superfluous parent elements when writing HTML. Usually this requires iteration and refactoring, but produces less HTML, which is good for maintenance.



## Learn to Code - HTML & CSS

# Exercise

Create the markup for an article with a comment form, it should include:

1. Different containers for the article content and the comment form
2. **Content:** A title, a subtitle, a date, an author (linked to it's personal Twitter account), a list, some placeholder text paragraphs, a picture of Nicolas Cage on a muffin, in space.
3. **Comment form:** A name field, an email field, a textarea, a 'remember my data' option and a submit button.