

# Sprint 1 - The Journey Begins

Here we go.

---

## Epic 1: Working Environment Set-up

This first epic will focus on setting up your environment, and making sure you have the right tools to start the project.

You could spend hours surfing the internet finding what works best, but luckily for you, we've provided you with a list of the most common and useful ones.

### Task 1: Install your web development tools

Here's a list of everything you'll need:

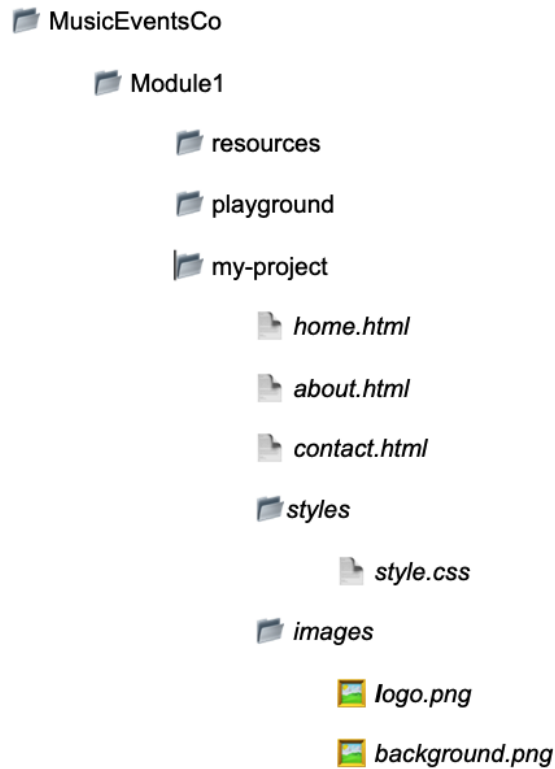
1. **A modern browser:** I recommend **Chrome** since it's the most widely accepted and really good for debugging.
2. **A text editor:** You will need some kind of software to write all your code in. While you can write software in a platform like notepad, this is neither considered good practice nor does it follow company standard. You need a **text editor**. We prefer you to use **Visual Studio Code** as it's very versatile.
3. **Live Server:** Once VSC (Visual Studio Code) is installed, run it and head to the Extensions on the left-hand side of your screen. Search and install the **Live Server** plugin. This will allow you to run a local server and see your work immediately, just by saving your progress.

Once we have all these tools set up we can move onto more **important** setup.

### Task 2: Folder Structure + Playground

It is crucial for a web developer or any kind of programmer to have great folder organisation.

There is no perfect way of doing things, everyone has a different way of organizing files, but this is an example of how I would structure the folders:



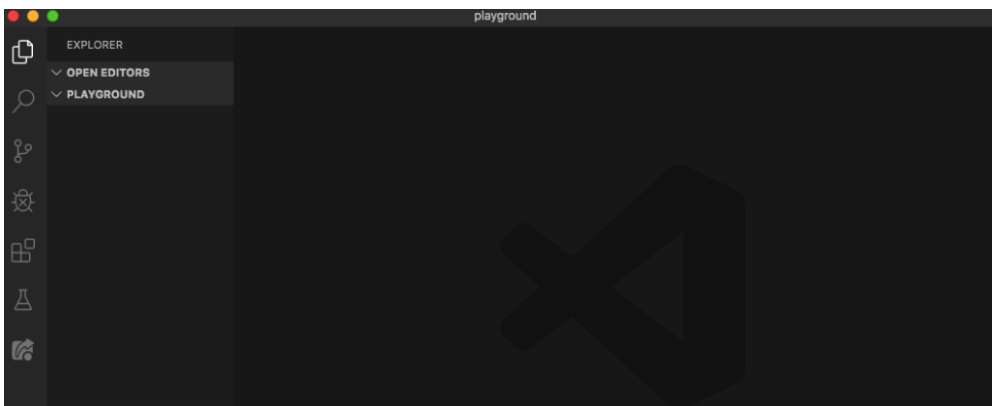
Folder organization

Within the *Module 1* folder, you'll file everything related to it within sub-folders.

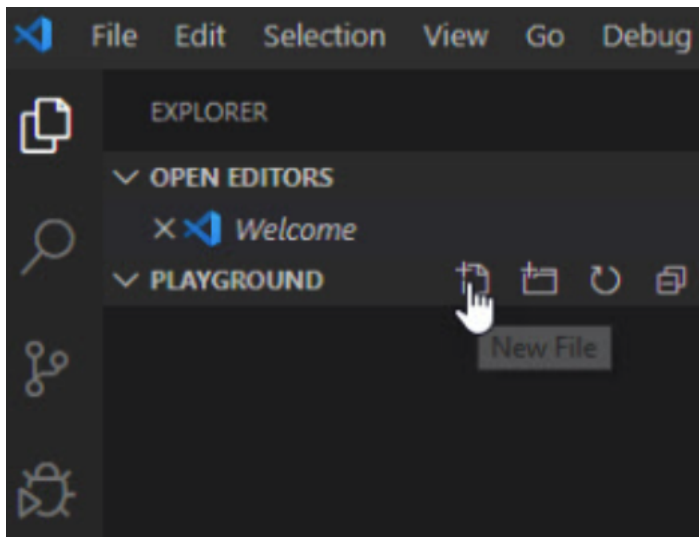
For example, things like **wireframes** and **pdfs** belong in the *resources* folder, while something like **test code** belongs in the *playground* folder.

**Note:** Your *my-project* folder (or however you choose to name it) is exclusively for the files and images which will be a part of your website.

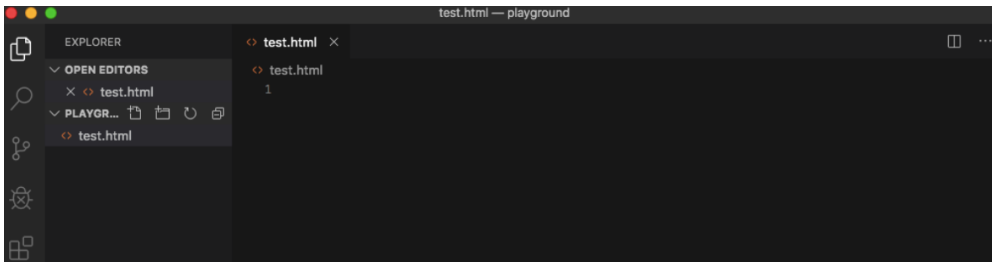
Don't worry, you don't have to create this folder structure just yet, it's merely an example of what you'll be doing in your future projects.



Pointing over the folder name, four icons will appear. To create a new file, click the furthest icon on the left.

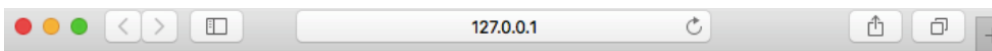


Create a new file called **test.html**. ⚠️ Make sure that the extension is *.html* and nothing else.



test.html within playground folder

Congratulations 🎉, you have created your first HTML file. Now, to see the results of your code, and to visualize it on the browser, you need to **use the plugin that you installed in the previous task - the Live Server**.



Now, let's insert some code into this file.

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
```

```

5   <meta charset="utf-8">
6   <title>Hello World</title>
7 </head>
8
9 <body>
10   <!-- Content code goes here -->
11 </body>
12 </html>

```

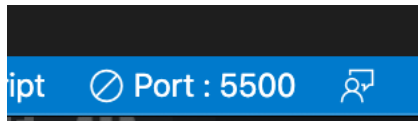
The code above is the "skeleton" of an **html** page, the basic structure. You add the content (what will be *visible* to anyone looking at your site) inside the `<body>` tag:

```

1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="utf-8">
6   <title>Hello World</title>
7 </head>
8
9 <body>
10   <p>HTML Chess Game</p>
11   <p>The king (♔,♚)</p>
12   <p>
13     A king can move one square in any direction (horizontally, vertically, or
14     diagonally) unless the square is already occupied by a friendly piece or
15     the move would place the king in check.
16   </p>
17   <p>The queen (♕,♛)</p>
18   <p>
19     The queen can be moved any number of unoccupied squares in a straight line
20     vertically, horizontally, or diagonally, thus combining the moves of the
21     rook and bishop.
22   </p>
23   <p>The bishop (♗,♝)</p>
24   <p>
25     The bishops may be differentiated according to which wing they begin on,
26     i.e. the king's bishop and queen's bishop. As a consequence of its
27     diagonal movement, each bishop always remains on either the white or black
28     squares, and so it is also common to refer to them as light-squared or
29     dark-squared bishops.
30   </p>
31   <p>The knight (♞,♟)</p>
32   <p>
33     The knight move is unusual among chess pieces. It moves to a square that
34     is two squares away horizontally and one square vertically, or two squares
35     vertically and one square horizontally. The complete move therefore looks
36     like the letter "L". Unlike all other standard chess pieces, the knight
37     can "jump over" all other pieces (of either color) to its destination
38     square.
39   </p>
40   <p>The rook (♖,♜)</p>
41   <p>
42     The rook moves horizontally or vertically, through any number of
43     unoccupied squares (see diagram).
44   </p>
45   <p>The pawn (♙,♟)</p>
46   <p>
47     Unlike the other pieces, pawns cannot move backwards. Normally a pawn
48     moves by advancing a single square, but the first time a pawn moves, it
49     has the option of advancing two squares.
50   </p>
51 </body>
52 </html>

```

If you have started the Live Server before, you don't need to click it again. Save your work (Ctrl+S or ⌘+S), and open the browser at **localhost:5500** (or the port Live Server has opened).



This is how it should look like:

1

#### HTML Chess Game

##### The king (♔,♚)

A king can move one square in any direction (horizontally, vertically, or diagonally) unless the square is already occupied by a friendly piece or the move would place the king in check.

##### The queen (♕,♛)

The queen can be moved any number of unoccupied squares in a straight line vertically, horizontally, or diagonally, thus combining the moves of the rook and bishop.

##### The bishop (♗,♝)

The bishops may be differentiated according to which wing they begin on, i.e. the king's bishop and queen's bishop. As a consequence of its diagonal movement, each bishop always remains on either the white or black squares, and so it is also common to refer to them as light-squared or dark-squared bishops.

##### The knight (♞,♟)

The knight move is unusual among chess pieces. It moves to a square that is two squares away horizontally and one square vertically, or two squares vertically and one square horizontally. The complete move therefore looks like the letter "L". Unlike all other standard chess pieces, the knight can "jump over" all other pieces (of either color) to its destination square.

##### The rook (♖,♜)

The rook moves horizontally or vertically, through any number of unoccupied squares (see diagram).

##### The pawn (♙,♟)

Unlike the other pieces, pawns cannot move backwards. Normally a pawn moves by advancing a single square, but the first time a pawn moves, it has the option of advancing two squares.

As a little summary:

3

#### What is HTML

Well, according to Wikipedia, Hypertext Markup Language or the well known, HTML is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web (www).

In other words, HTML is the structure of the skeleton of your web application. What the browser will load every time anyone opens your webpage. Here you can find an example of the basic HTML structure.

#### BASIC STRUCTURE (syntax):

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Page Title</title>
5 </head>
6 <body>
7
8 <h1>My First Heading</h1>
9 <p>My first paragraph.</p>
10
11 </body>
12 </html>
```

As Wikipedia said, HTML is a markup language, which means that we are going to use different types of tags in order to build our skeleton of the webpage. We met some of the basic tags in one HTML file, `<!DOCTYPE html>` , `<head>` and `<body>` . We are going to focus more on what to insert into the `<body>` tag because, if you remember, **this is the part that our users will see** 😊. There are a lot of different tags appearing day by day so there is no need to memorize all of them.

⚠ The only thing that you should remember, is that **once you open an html tag you should always close it!**

Observe the example below. You can see that there are several html tags in the `<body>` . h1 tag stands for your website's heading title and p stands for paragraph (text). We open the tag with `<h1>` and we close it `</h1>` .

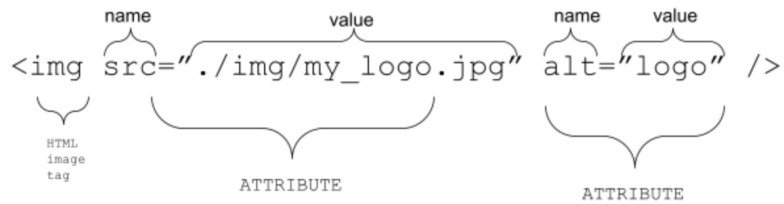
```
1 <!DOCTYPE html>
2 <html>
3 <body>
4
5
6 <h1>My First Heading</h1>
7 <p>My First Paragraph.</p>
8
9 </body>
10 </html>
```

#### ⚠ Things to keep in mind.

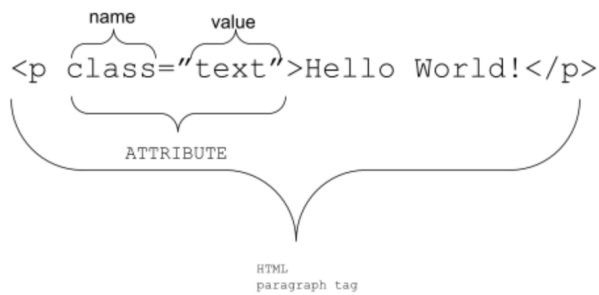
Not all the tags in HTML should finish like the h1 `</h1>` . There are some tags called **SELF CLOSING** tags. An example for tag like this will be the image tag, `<img>` . Instead of doing `<img></img>` you can do just `<img/>` .

## ATTRIBUTES

To add additional information to the HTML tags we are going to use ATTRIBUTES. Any html element can have attributes. They are always specified in the starting tag. They come in pairs like **name=value**. Some of the most famous are **id** and **class**. Let's take for example the previously used tag `<img/>`. An important attribute for the image tag is `src` (the source of the image). So as we said before, the attribute can be applied in this order, **name=value**.



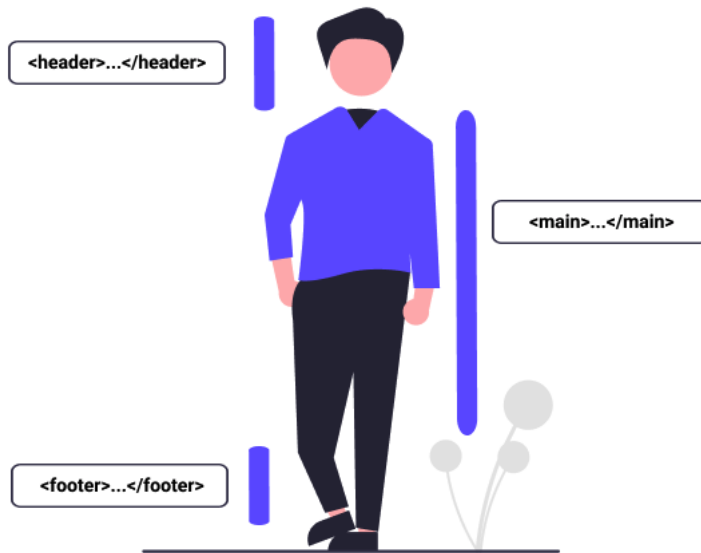
The diagram illustrates the structure of the HTML image tag ``. It uses curly braces to group parts of the tag. A brace under `<img` is labeled "HTML image tag". Two braces under the `src` attribute are labeled "name" and "value". Another two braces under the `alt` attribute are also labeled "name" and "value". Braces under the entire `src="./img/my_logo.jpg"` and `alt="logo"` sections are both labeled "ATTRIBUTE".



The diagram illustrates the structure of the HTML paragraph tag `<p class="text">Hello World!</p>`. It uses curly braces to group parts of the tag. Two braces under the `class` attribute are labeled "name" and "value". A brace under the entire `class="text"` section is labeled "ATTRIBUTE". A large brace under the entire `<p class="text">Hello World!</p>` section is labeled "HTML paragraph tag".

## GOOD PRACTICE

In order to have a well organized html file we need to prepare the skeleton ( `<body>` ) in the proper way. The key word skeleton is used on purpose. The html layout structure is similar to the human one. Each main part of the human body can be represented with html tag.



So let's apply this to a real life project shall we?

Imagine that you receive something like this:



The best way to achieve a great html layout (skeleton) is to make a blueprint.

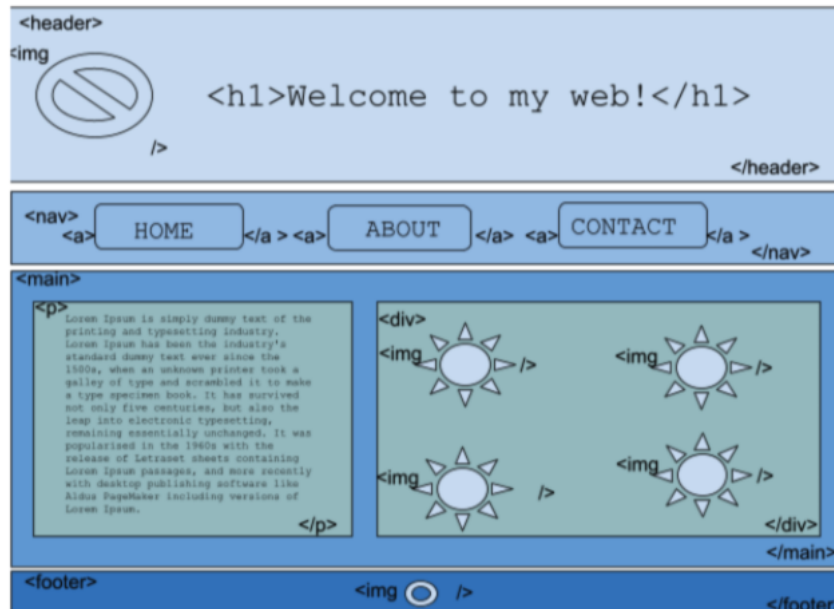
Have you seen anybody build a building without a blueprint? No? Of course not! So we follow the same rule. Grab a pen and paper and let's make our blueprint according to the picture above. As we said in the beginning, eyes on the `<body>` tag. The blue photo between code blocks represents what a user sees in the browser. In other words, the HTML **body**. The user **only** sees what is in the HTML **body**.



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <title>My first web</title>
6 </head>
7 <body>

```



```

1 </body>
2 </html>

```

Now this is a job well done! Visible and easy to be transformed into html. Stick to the plan and you never gonna waste time on tasks like this 😊. Most likely, the result wont look exactly like the image above. Remember we are **ONLY** using HTML!

Now **let's customise this page** to make the webpage more reader-friendly. When you load your browser, all the information will appear in the same text-size and font. Like in chess, each figure has its own place in the hierarchy of the game.

Using the HTML **headings** tags, try to establish a hierarchy among your `<p>` tags. In other words, change the **names** of the chess pieces by assigning to them a proper heading tag. Done? Check the changes in the browser! The piece names should now be bolded with a greater font-size, according to their place within the hierarchy. Like this:

## The king (♔,♚)

A king can move one square in any direction (horizontally, vertically, or diagonally) unless the square is already occupied by a friendly piece or the move would place the king in check.

## The queen (♕,♛)

The queen can be moved any number of unoccupied squares in a straight line vertically, horizontally, or diagonally, thus combining the moves of the rook and bishop.

## The bishop (♗,♝)

The bishops may be differentiated according to which wing they begin on, i.e. the king's bishop and queen's bishop. As a consequence of its diagonal movement, each bishop always remains on either the white or black squares, and so it is also common to refer to them as light-squared or dark-squared bishops.

## The knight (♞,♞)

The knight move is unusual among chess pieces. It moves to a square that is two squares away horizontally and one square vertically, or two squares vertically and one square horizontally. The complete move therefore looks like the letter "L". Unlike all other standard chess pieces, the knight can "jump over" all other pieces (of either color) to its destination square.

## The rook (♖,♜)

The rook moves horizontally or vertically, through any number of unoccupied squares (see diagram).

## The pawn (♙,♟)

Unlike the other pieces, pawns cannot move backwards. Normally a pawn moves by advancing a single square, but the first time a pawn moves, it has the option of advancing two squares.

Using header tags

⚠ When the exercise is done, don't forget to validate your code using this [HTML VALIDATOR](#). This tool checks all the HTML syntax and tells you all the possible errors you might have. Even if your page seems fine in the browser, if it has bad syntax it can crash in other browsers, so it is important to use it in all your pages to avoid future problems.