# Module 3 – Web Development Basics

# HTML Document Setup and Basic Elements

In this article, we are going to learn about the basic HTML document setup and how to use some basic HTML elements.

## About HTML

HTML stands for Hyper Text Markup Language and it describes the structure of Web pages by using markup. All the HTML elements are represented as tags, which we will describe in more detail a bit later. We can’t see these HTML tags in a browser, HTML uses them just to render the content on the page.

To work with examples, we are going to use VisualStudio Code editor. To create a new HTML file click on File menu and then NewFile. We get the untitled document. So, all we have to do is to save it with the .html extension at the end of the document name.

## HTML Document

Every HTML file must start with the document type declaration:

<!DOCTYPE html>

The HTML document begins with the <html> tag and ends with the </html> closing tag.

Inside the <html> tag, we use the <body></body> tags to specify the body of the HTML document:

<!DOCTYPE html>

<html>

<body>

This is some text inside the body tag.

</body>

</html>

Between the <html> and the <body> tag, we use the <head></head> tags. The <head> tag contains the metadata about the HTML document. With metadata, we define doucment title, styles, character set, links, script and other informations. For all the mentioned metadata, we can specify tags: <title>, <style>, <meta>, <link>, <script> and <base> :

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>This is page title.</title>

</head>

<body>

This is some text inside the body tag.

</body>

</html>

## HTML Headings

We can define headings in HTML with the tags from <h1> to <h6>. The <h1> tag produce the most important heading and <h6> the least important one:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>This is page title.</title>

</head>

<body>

This is some text inside the body tag.

<h1>This is heading 1</h1>

<h2>This is heading 2</h2>

<h3>This is heading 3</h3>

<h4>This is heading 4</h4>

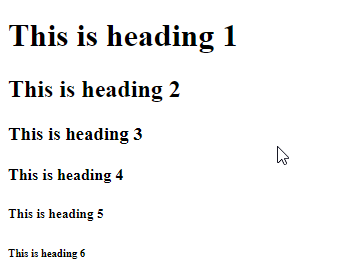
<h5>This is heading 5</h5>

<h6>This is heading 6</h6>

</body>

</html>

Result:



## HTML Paragraph

To define paragraph in HTML, we use the <p> tag:

<p>

This is the first paragraph.

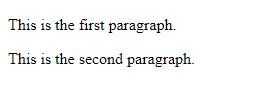
</p>

<p>

This is the second paragraph.

</p>

Result:



## HTML <div> Element

We use the <div> element to represent a block or a section of the HTML content. We can think of it as a placeholder for our content:

<div>

<h3>This is the first document section</h3>

<p>

Some content in here.

</p>

</div>

<div>

<h4>This is the second section.</h4>

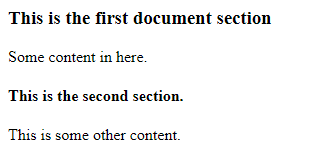
<p>

This is some other content.

</p>

</div>

Result:



## HTML <span> Element

The <span> tag is an inline element in HTML and it will not start in another row:

<div>

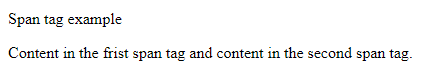
<p>Span tag example</p>

<span>Content in the frist span tag </span>

<span>and content in the second span tag.</span>

</div>

Result:



## HTML Images

To display image in the HTML document, we can use the <img> tag. This tag is self-closing tag, which means that we don’t have the </img> tag.

The sole <img> tag is not enough to display image, we need to use additional attribute src, which describes the path to the image. If the image is in the same place as the HTML document then for the path we can use just image name and extension. But if the image is outside the folder where the HTML document is, we need to use this syntax: ../image-name-and-extension.

One more attribute is required by HTML standards and it is the alt attribute. With it we can specify what user see if the image is not loaded:

<p>This is Code Maze picture:</p>

<img src="code-maze.png" alt="Code-Maze Picture" />

Result:



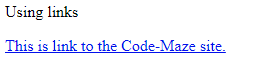
## HTML Links

We can use links in HTML to direct a user to another HTML page. In HTML we specify links with the <a> tag. As with the <img> tag, we need additional attributes to specify on which location this link is pointing and do we want to open that page in the same tab we are currently on or in another one. To specify location we use the href attribute:

<p>Using links</p>

<a href="https://codemaze.com">This is link to the Code-Maze site.</a>

Result:



If we want to open the linking document in a new tab, we can use the target attribute with the \_blank value:

<p>Using links</p>

<a href="https://codemaze.com" target="\_blank">This is link to the Code-Maze site.</a>

## Empty HTML Elements

Html elements without content are called empty elements. The <br /> is an empty element. We use the <br> tag to create a line break in the HTML content.

## Conclusion

We have learned about the basic setup in HTML document and how to use basic HTML elements to render content on the HTML page. There are a lot more elements that we can use in HTML and we are going to cover many of them in the future articles.

In the next article, we are going to talk about formatting in HTML and how to use formatting elements.

# HTML Formatting Elements And Comments

With the help of the HTML formatting elements, we can render a special type of text on the page. So, in this article, we are going to learn what are the most used elements for text formatting and how to use them.

## HTML <b> and <strong> tags

Both of this tags will make our text bold, but there is one difference. The <b> tag doesn’t apply any semantic importance to the text, while the <strong> tag does that:

<!DOCTYPE html>

<html>

<body>

<p>

<b>This text is bold.</b>

</p>

<p>

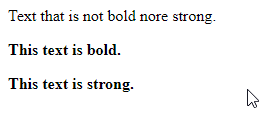
<strong>This text is strong.</strong>

</p>

</body>

</html>

Result:



## HTML <i> and <u> Elements

The <i> tag will render italic text and the <u> tag will render underline text:

<p>

<i>This is italic text.</i>

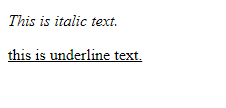
</p>

<p>

<u>this is underline text.</u>

</p>

Result:



## HTML <small>, <mark> and <del> Elements

By using the <small> tag, our text will become smaller. If we want to mark our text, we use the <mark> tag. The <del> tag defines removed element:

<p>

This word is <small>smaller</small> than other ones.

</p>

<p>

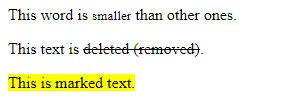
This text is <del>deleted (removed)</del>.

</p>

<p>

<mark>This is marked text.</mark>

</p>



## HTML <sub> and <sup> Elements

The <sub> tag defines subscripted text and the <sup> tag renders the superscripted text:

<p>

Let's write math formula: a<sup>2</sup> + b<sup>2</sup> = c<sup>2</sup>.

</p>

<p>

Let's write chemichal formula: H<sub>2</sub> + O = H<sub>2</sub>O.

</p>

Result:

## 

## HTML Comments

When we want to comment some part of our HTML code, we can use the following syntax <!---->. The commented part won’t be rendered in our page:

<!--This is the comment in HTML-->

## Conclusion

With this knowledge, we can format our text on an HTML page. Once we combine them together we can get a great variety of formatted text.

# CSS Introduction, Syntax and Style Insertion

In this article, we are going to learn about basic CSS concepts. We will learn what is CSS, how to use its syntax and how to inject it into our HTML page.

## What is CSS?

CSS stands for Cascading Style Sheets and we can use it to style our pages. With the HTML tags we render content on the page but with the CSS we can style that content. In combination, HTML and CSS make a powerful tool in web development.

## CSS Syntax

A CSS styling rules consist of a selector and the declaration between curly braces:



With a selector, we are pointing to the HTML element we want to style. In the picture above we are pointing to the HTML <p> tag. The curly braces present a body for the CSS declaration. Each declaration consists of property name and its value separated by a colon. As we can see from the picture, every declaration statement must end with the semicolon. We can add as many declarations inside the curly braces as we need:

p{

property:value;

property2: value2;

property3: value3;

...

}

## CSS Selectors

We use CSS selectors to point to the element we want to style.

There are three basic selectors in CSS:

* Element
* Id
* Class

The **element** points to the HTML tag in our HTML page. So, for example, this CSS rule:

div{

color: blue;

}

will style all the <div> elements inside the HTML page.

The **Id** selector points to the element on the HTML page that has the unique Id attribute with the same name as the selector. To use the Id selector we must use the # sign in front of the element’s id:

#selectorId{

color: blue;

}

This style applies to the single HTML element that has the Id attribute with the selectorId value:

<div id="selectorId">Text to style.</div>

We can use Id’s all over the HTML page, but every single one of them must be unique.

The **class** selector points to all the elements that contain the class attribute. As you can see, we are saying „all the elements“ because the same css class can be included into multiple HTML tags. To create a class rule we need to write a „.“ sign in front of the class name:

.newClass{

color: red;

}

This style will apply to every element with the same class attribute:

<p class="newClass">Text to style.</p>

<span class="newClass">Text to style.</span>

We can group our selectors as well. For example, if we have the same style rules for the <div> and <p> and <span> elements, we don’t have to repeat the same declaration for each element. What we can do is to group them into a single rule:

div, p, span{

color: green;

}

## Comments in CSS

From the previous article, we know that we can use comments in the HTML page. Well, we can do the same thing with CSS:

/\* This is the comment \*/

/\*

This is the

multiline

comment

\*/

## Injecting Style Sheet Into HTML Page

We can write as many CSS rules as we like, but until we include them into the HTML page, they won’t be applied. So, let’s talk about the ways we can include our CSS rules.

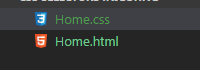
There are three ways to do that:

* External style sheet
* Internal style sheet
* Inline style sheet

## External Style Sheet

With the external style sheet, we create our CSS rules in the separate CSS file and then use the <link> tag to include it into the HTML page.

So let’s create one CSS file and one HTML file in our editor:



Then, let’s add one class to the Home.css file:

.homeTextColor{

color: red;

}

Finally, let’s modify the Home.html file:

<!DOCTYPE html>

<html>

<body>

<p class="homeTextColor">This is a paragraph.</p>

</body>

</html>

If we open the Home.html file in a browser, we won’t be able to see any text with a red color. That's because we didn’t insert the CSS file inside the HTML file. So let’s do that:

<!DOCTYPE html>

<html>

<head>

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

</head>

<body>

<p class="homeTextColor">This is a paragraph.</p>

</body>

</html>

Inside the <head> tag we use the <link> tag with several attributes to insert our CSS file. The rel attribute specifies the relationship between the current document and the linked document. The type attribute specifies the type of the inserted file and the href attribute specifies the location of the CSS file.

Now if we open the Home.html page in a browser, we are going to see this result:



## Internal Style Sheet

We can declare internal styles inside the HTML page with a help of the <style> tag. The <style> element must reside in the <head> section:

<!DOCTYPE html>

<html>

<head>

<style>

.homeTextColor{

color: red;

}

</style>

</head>

<body>

<p class="homeTextColor">Welcome to the HOME page!!!</p>

</body>

</html>

## Inline Style

We can use inline style inside the HTML tag with the style attribute:

<!DOCTYPE html>

<html>

<body>

<p style="color:red;">Welcome to the HOME page!!!</p>

</body>

</html>

The result is the same for any of these three examples.

## Conclusion

In this article, we have learned what is CSS, how to use its syntax and how to include it in the HTML page. Of course, we saw only one attribute in work (the color attribute), but in the next article, we are going to talk a lot more about CSS attributes.

So, see you there.

# CSS Properties

In the CSS world, we can find a great variety of properties to style our HTML pages. In this article, we will explain how to use some of the CSS properties. In the previous article, we have learned how to import CSS into an HTML page, so, feel free to choose any way you are comfortable with in the following examples.

## CSS Colors and Backgrounds

To color our text in HTML, we can use the color property. If we want to set a background color for our HTML elements, we can use the background-color property:

<p class="colorText">

This is white text with the blue background.

</p>

.colorText{

color: white;

background-color: blue;

}



We can use a color name to set the value of the color property, as we did in previous examples. But, we can use the hexadecimal values as well. On this [page](https://htmlcolorcodes.com/color-names/), we can find many different color names with its hex-a values.

If we want to set an image as our background we can do that by using the background-image property:

<p class="imageBackground">

Lorem ipsum dolor sit amet, no quot postulant salutatus vis, nonumy maiestatis est ex. Eripuit reprimique nec ex.<br>

Vel maiestatis disputationi cu, no paulo facete graecis nam, ad ius virtute alienum deseruisse. <br>

Intellegam constituam efficiantur no qui, in vel nonumes principes consetetur, at duo altera iuvaret. <br>

Eu vim duis temporibus, cum dicam saperet et.

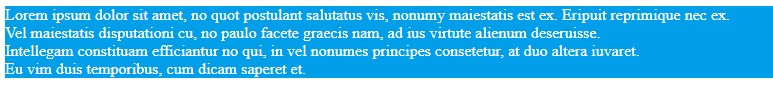
</p>

.imageBackground{

color: white;

background-image: url("blue-background.png");

}



## Borders

To create borders on the HTML elements, we can use the border property. As a value for that property, we need to provide border width, border style, and border color:

border: 3px solid green;

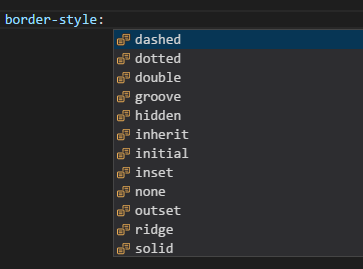
Of course, we can style our borders with the border width, border style, and border color as a separate property:

border-width: 5px;

border-style: solid;

border-color: green;

There are several different border styles:



## Margins

To create space around HTML elements, from other elements, we can use the margin property. As with the border property, we can use the margin property as a single property with top, bottom, left and right values. But we can use them separately as well:

margin: 2px 3px 5px 6px; /\*TOP, RIGHT, BOTTOM, LEFT\*/

margin-top: 2px;

margin-right: 3px;

margin-bottom: 5px;

margin-left: 6px;

If the margin property has only one value, that value is applied on all four sides. If the margin property has two values, then the first one is applied to the top and bottom margins, and the second one is applied to the right and left margins.

## Padding

We use the padding attribute to create a space around a content inside defined borders. The same story with the margin property applies on the padding property. So we can define the space to all four sides separately or with the single padding property:

padding: 2px 3px 5px 6px; /\*TOP, RIGHT, BOTTOM, LEFT\*/

padding-top: 2px;

padding-right: 3px;

padding-bottom: 5px;

padding-left: 6px;

We can create the padding property with only one value or with two values in the same way as with the margin attribute.

The combination of the margin, padding, and border properties represent the box model:



## Width – Height

With these properties we can set the width and height of our elements:

width: 120px;

height: 300px;

We can set the values for the width and height properties by using different units: px, cm, %(percent) of the containing block etc.

Now let’s see one combined example of the above properties:

<div class="firstBox">

This is first box

</div>

<div class="secondBox">

This is second box

</div>

<div class="thirdBox">

This is third box

</div>

.firstBox{

border: 1px solid green;

color: tomato;

margin: 10px 10px 5px 20px;

padding: 10px 60px;

width: fit-content;

}

.secondBox{

border: 3px dotted red;

color: green;

margin: 45px 10px 5px 40px;

padding: 15px;

width: fit-content;

}

.thirdBox{

border: 5px dotted blue;

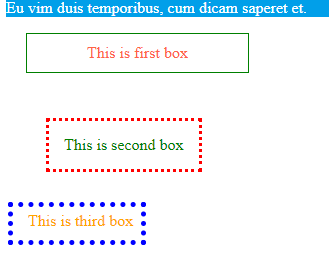
color: orange;

margin: 30px 10px 5px 2px;

padding: 5px 8px 10px 15px;

width: fit-content;

}



## Text in CSS