**CSS Basics #2: Some Basics Components -** [Eric Hu](https://huericnan.medium.com/?source=post_page-----31d7b0c7a724--------------------------------) Feb 6, 2022

In this article, we are going to go through some of the most basic components in CSS. Let’s start with colours.

*You can download the source code for this tutorial here:*

[*Download the Source Code*](https://www.ericsdevblog.com/index.php/beginners-roadmap-to-web-development/)

**Colours**

There are four different ways we can define a colour in CSS. The easiest method is to use a predefined colour name like this:

<body>  
 <div>  
 <p>  
 ...  
 </p>  
 </div>  
</body>div {  
 background-color: tomato;  
}p {  
 color: white;  
}

Text

Description automatically generated

The colours can also be specified using RGB values, HEX values or HSL values like this:

<body>  
 <div>  
 <p>  
 ...  
 </p>  
 </div> <section>  
 <p>  
 ...  
 </p>  
 </section>  
</body>div {  
 background-color: rgb(255, 99, 71);  
}section {  
 background-color: #008080;  
}p {  
 color: hsl(0, 0%, 100%);  
}

Text

Description automatically generated

**Texts and Fonts**

We can customize texts using CSS. Besides setting text colours, we can add or remove decorations using the text-decoration property:

<body>  
 <h1>First Heading</h1>

<h2>Second Heading</h2>

<h3>Third Heading</h3>  
</body>

h1 {  
 text-decoration: overline;  
}

h2 {  
 text-decoration: line-through;  
}

h3 {  
 text-decoration: underline;  
}

Text

Description automatically generated

Or we can transform the text into uppercase, lowercase or the capitalized form:

<body>  
 <p class="uppercase">Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>  
 <p class="lowercase">Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>  
 <p class="capitalize">Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>  
</body>.uppercase {  
 text-transform: uppercase;  
}.lowercase {  
 text-transform: lowercase;  
}.capitalize {  
 text-transform: capitalize;  
}

Graphical user interface, text, application

Description automatically generated

In order to make the texts more appealing, we can assign different fonts to our texts like this:

<body>  
 <p class="p1">Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>  
 <p class="p2">Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>  
 <p class="p3">Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>  
</body>.p1 {  
 font-family: "Courier New", Courier, monospace;  
}.p2 {  
 font-family: "Lucida Sans", "Lucida Sans Regular", "Lucida Grande",  
 "Lucida Sans Unicode", Geneva, Verdana, sans-serif;  
}.p3 {  
 font-family: Georgia, "Times New Roman", Times, serif;  
}

Graphical user interface, text, application

Description automatically generated

As you can see, here we assigned several fonts for each paragraph. That is to ensure there is a fallback system, if the first font is not available, the browser will go to its generic “siblings” instead.

Finally, we can also define the style, weight and size for the font:

.p1 {  
 font-family: "Courier New", Courier, monospace;  
 font-style: italic;  
 font-weight: 100;  
 font-size: 30px;  
}.p2 {  
 font-family: "Lucida Sans", "Lucida Sans Regular", "Lucida Grande",  
 "Lucida Sans Unicode", Geneva, Verdana, sans-serif;  
 font-style: normal;  
 font-weight: 200;  
 font-size: 20px;  
}.p3 {  
 font-family: Georgia, "Times New Roman", Times, serif;  
 font-style: oblique;  
 font-weight: 300;  
 font-size: 10px;  
}

Text

Description automatically generated

**Links**

At the end of the previous article, we talked about pseudo-classes, which describe the state of the elements. Links are where we use pseudo-classes the most often.

The four links states are:

* a:link - a normal, unvisited link
* a:visited - a link the user has visited
* a:hover - a link when the user mouses over it
* a:active - a link the moment it is clicked

Using these four states, we can style a link like this:

<body>  
 <a href="#" target="\_blank">This is a link.</a>  
</body>/\* unvisited link \*/  
a:link {  
 color: blue;  
 text-decoration: underline;  
}/\* visited link \*/  
a:visited {  
 color: green;  
 text-decoration: underli;  
}/\* mouse over link \*/  
a:hover {  
 color: red;  
 text-decoration: overline;  
}/\* selected link \*/  
a:active {  
 color: orange;  
 text-decoration: underline;  
}

A picture containing graphical user interface

Description automatically generated

Unvisited Link

A picture containing shape

Description automatically generated

Hovered Link

A picture containing logo

Description automatically generated

Active Link

A picture containing icon

Description automatically generated

Visited Link

**Tables**

The look of an HTML table can also be greatly improved with CSS. A typical HTML table looks like this:

<body>  
 <table>  
 <tr>  
 <th>Firstname</th>  
 <th>Lastname</th>  
 <th>Age</th>  
 </tr>  
 <tr>  
 <td>Jill</td>  
 <td>Smith</td>  
 <td>50</td>  
 </tr>  
 <tr>  
 <td>Eve</td>  
 <td>Jackson</td>  
 <td>94</td>  
 </tr>  
 <tr>  
 <td>John</td>  
 <td>Doe</td>  
 <td>80</td>  
 </tr>  
 </table>  
</body>table, td, th {  
 border: 1px solid black;  
}

Table

Description automatically generated

The first thing we notice is that the table has double borders. That is because both the table itself (<table>) and the <th> and <td> elements have separate borders. We can use the border-collapse property to make them collapse into one single border:

table {  
 border-collapse: collapse;  
}

Table

Description automatically generated

We can use the width and height properties to set the size for the columns/rows or the entire table:

table {  
 border-collapse: collapse;  
 width: 100%;  
}th {  
 height: 70px;  
}

Table

Description automatically generated

We can define the alignment inside the table like this:

td {  
 text-align: center;  
}

Table

Description automatically generated

Lastly, we can set a colour for the table:

th {  
 height: 70px;  
 background-color: coral;  
 color: white;  
}td {  
 text-align: center;  
 background-color: gainsboro;  
}

Table

Description automatically generated

**Forms and Buttons**

In the final section of this article, we are going to talk about how to customize HTML forms. Let’s take a look at an example:

<body>  
 <form>  
 <label for="firstname">First Name:</label><br />  
 <input type="text" id="firstname" /><br />  
 <label for="lastname">Last Name:</label><br />  
 <input type="text" id="lastname" /><br />  
 <button>Submit</button>  
 </form>  
</body>

Table

Description automatically generated with medium confidence

Now we can add some CSS code:

input[type=text] {  
 width: 20%;  
 padding: 12px 20px;  
 margin: 8px 0;  
 box-sizing: border-box;  
}

A picture containing table

Description automatically generated

However, the button still doesn’t look very good, here is what we can do:

button {  
 background-color: #4CAF50;  
 color: white;  
 border: none;  
 padding: 10px 20px;  
 text-align: center;  
 font-size: 16px;  
}

A picture containing text

Description automatically generated

Finally, let’s take one step further, let’s also add a border to the button, and make it hoverable using pseudo-class:

button {  
 background-color: white;  
 color: #4CAF50;  
 border: solid #4CAF50;  
 padding: 10px 20px;  
 text-align: center;  
 font-size: 16px;  
 transition-duration: 0.2s;  
}button:hover {  
 background-color: #4CAF50;  
 color: white;  
 border: solid #4CAF50;  
 padding: 10px 20px;  
 text-align: center;  
 font-size: 16px;  
}

A picture containing shape

Description automatically generated

Original Button

A picture containing text

Description automatically generated

Hovered Button