



HAZZA INSTITUTE
OF TECHNOLOGY



MERN – ES6 + React

Module 04: CSS Advanced

Outline

Module 04

- ▶ learning about Fonts
- ▶ learning about Box Model
- ▶ learning about font properties with CSS
- ▶ learning about size
- ▶ learning about weight
- ▶ learning about text alignment

CSS Text

CSS has a lot of properties for formatting text.

▶ Text Color

- The color property is used to set the color of the text. The color is specified by:
 - a color name - like "red"
 - a HEX value - like "#ff0000"
 - an RGB value - like "rgb(255,0,0)"

- ▶ The default text color for a page is defined in the body selector.
- ▶ **Important:** High contrast is very important for people with vision problems. So, always ensure that the contrast between the text color and the background color (or background image) is good!

CSS Text

CSS has a lot of properties for formatting text.

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-color: lightgrey;
  color: blue;
}

h1 {
  background-color: black;
  color: white;
}

div {
  background-color: blue;
  color: white;
}
</style>
</head>
<body>

<h1>This is a Heading</h1>
<p>This page has a grey background color and a blue text.</p>
<div>This is a div.</div>

</body>
</html>
```

This is a Heading

This page has a grey background color and a blue text.

This is a div.

CSS Fonts

Choosing the right font has a huge impact on how the readers experience a website.

► Generic Font Families

- In CSS there are five generic font families:

1. **Serif** fonts have a small **stroke at the edges** of each letter. They create a sense of **formality** and elegance.
2. **Sans-serif** fonts have **clean lines** (no small strokes attached). They create a **modern** and minimalistic look.
3. **Monospace** fonts - here all the letters have the same fixed width. They create a **mechanical** look.
4. **Cursive** fonts imitate **human handwriting**.
5. **Fantasy** fonts are **decorative/playful** fonts.

► **Note:** On computer screens, sans-serif fonts are considered easier to read than serif fonts.



Sans-serif



Serif



Serif
(red serifs)

CSS Fonts

Choosing the right font has a huge impact on how the readers experience a website.

Generic Font Family	Examples of Font Names
Serif	Times New Roman Georgia Garamond
Sans-serif	Arial Verdana Helvetica
Monospace	Courier New Lucida Console Monaco
Cursive	<i>Brush Script MT</i> <i>Lucida Handwriting</i>
Fantasy	Copperplate Papyrus

CSS Fonts

Choosing the right font has a huge impact on how the readers experience a website.

```
<!DOCTYPE html>
<html>
<head>
<style>
.p1 {
  font-family: "Times New Roman", Times, serif;
}

.p2 {
  font-family: Arial, Helvetica, sans-serif;
}

.p3 {
  font-family: "Lucida Console", "Courier New", monospace;
}
</style>
</head>
<body>

<h1>CSS font-family</h1>
<p class="p1">This is a paragraph, shown in the Times New Roman font.</p>
<p class="p2">This is a paragraph, shown in the Arial font.</p>
<p class="p3">This is a paragraph, shown in the Lucida Console font.</p>

</body>
</html>
```

CSS font-family

This is a paragraph, shown in the Times New Roman font.

This is a paragraph, shown in the Arial font.

This is a paragraph, shown in the Lucida Console font.

CSS Pseudo-classes

A pseudo-class is used to define a special state of an element.

- ▶ For example, it can be used to:
 - Style an element when a user **mouses over** it
 - Style **visited** and unvisited links differently
 - Style an element when it gets **focus**

Syntax

The syntax of pseudo-classes:

```
selector:pseudo-class {  
  property: value;  
}
```


CSS Pseudo-classes

A pseudo-class is used to define a special state of an element.

```
<!DOCTYPE html>
<html>
<head>
<style>
/* unvisited link */
a:link {
  color: red;
}

/* visited link */
a:visited {
  color: green;
}

/* mouse over link */
a:hover {
  color: hotpink;
}

/* selected link */
a:active {
  color: blue;
}
</style>
</head>
<body>

<h2>Styling a link depending on state</h2>

<p><b><a href="default.asp" target="_blank">This is a link</a></b></p>
<p><b>Note:</b> a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective.</p>
<p><b>Note:</b> a:active MUST come after a:hover in the CSS definition in order to be effective.</p>

</body>
</html>
```

Styling a link depending on state

[This is a link](#)

Note: a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective.

Note: a:active MUST come after a:hover in the CSS definition in order to be effective.

CSS Pseudo-elements

A CSS pseudo-element is used to style specified parts of an element.

- ▶ For example, it can be used to:
 - Style the **first** letter, or line, of an element
 - Insert content **before**, or **after**, the content of an element

Syntax

The syntax of pseudo-elements:

```
selector::pseudo-element {  
  property: value;  
}
```

CSS Pseudo-elements



A CSS pseudo-element is used to style specified parts of an element.

```
<style>
p::first-letter {
  color: #ff0000;
  font-size: xx-large;
}
```



```
p::first-line {
  color: #0000ff;
  font-variant: small-caps;
}
</style>
```

▶ **Y**OU CAN COMBINE THE `::FIRST-LETTER` AND `::FIRST-LINE` PSEUDO-ELEMENTS TO ADD A SPECIAL EFFECT TO THE FIRST LETTER and the first line of a text!

```
<style>
h1::after {
  content: url(smiley.gif);
}
h1::before {
  content: url(smiley.gif);
}
</style>
```

▶  **This is a heading** 

The `::before` pseudo-element inserts content before the content of an element.

 **This is a heading** 

CSS Links

With CSS, links can be styled in many different ways.

► Styling Links

- Links can be styled with any CSS property (e.g. color, font-family, background, etc.).
- In addition, links can be styled differently depending on what state they are in.
- The four links states are:

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

Text Link

Text Link

Link Button

Link Button

CSS Links

With CSS, links can be styled in many different ways.

```
<!DOCTYPE html>
<html>
<head>
<style>
a:link, a:visited {
  background-color: white;
  color: black;
  border: 2px solid green;
  padding: 10px 20px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
}

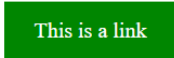
a:hover, a:active {
  background-color: green;
  color: white;
}
</style>
</head>
<body>

<h2>Link Button</h2>

<a href="default.asp" target="_blank">This is a link</a>

</body>
</html>
```

Link Button



This is a link

CSS Lists

Different List Item Markers

```
<!DOCTYPE html>
<html>
<head>
<style>
ul.a {
  list-style-type: circle;
}
ul.b {
  list-style-type: square;
}
ol.c {
  list-style-type: upper-roman;
}
ol.d {
  list-style-type: lower-alpha;
}
</style>
</head>
<body>
<h2>The list-style-type Property</h2>
<p>Example of unordered lists:</p>
<ul class="a">
<li>Coffee</li>
<li>Tea</li>
<li>Coca Cola</li>
</ul>
<ul class="b">
<li>Coffee</li>
<li>Tea</li>
<li>Coca Cola</li>
</ul>
<p>Example of ordered lists:</p>
<ol class="c">
<li>Coffee</li>
<li>Tea</li>
<li>Coca Cola</li>
</ol>
<ol class="d">
<li>Coffee</li>
<li>Tea</li>
<li>Coca Cola</li>
</ol>
</body>
</html>
```

The list-style-type Property

Example of unordered lists:

- Coffee
- Tea
- Coca Cola

- Coffee
- Tea
- Coca Cola

Example of ordered lists:

- I. Coffee
- II. Tea
- III. Coca Cola

- a. Coffee
- b. Tea
- c. Coca Cola

CSS Tables

The look of an HTML table can be greatly improved with CSS:

```
<!DOCTYPE html>
<html>
<head>
<style>
#customers {
  font-family: Arial, Helvetica, sans-serif;
  border-collapse: collapse;
  width: 100%;
}

#customers td, #customers th {
  border: 1px solid #ddd;
  padding: 8px;
}

#customers tr:nth-child(even){background-color: #f2f2f2;}

#customers tr:hover {background-color: #ddd;}

#customers th {
  padding-top: 12px;
  padding-bottom: 12px;
  text-align: left;
  background-color: #04AA6D;
  color: white;
}
</style>
</head>
<body>

<h1>A Fancy Table</h1>

<table id="customers">
```

A Fancy Table

Company	Contact	Country
Alfreds Futterkiste	Maria Anders	Germany
Berglunds snabbköp	Christina Berglund	Sweden
Centro comercial Moctezuma	Francisco Chang	Mexico
Ernst Handel	Roland Mendel	Austria
Island Trading	Helen Bennett	UK
Königlich Essen	Philip Cramer	Germany
Laughing Bacchus Winecellars	Yoshi Tannamuri	Canada
Magazzini Alimentari Riuniti	Giovanni Rovelli	Italy
North/South	Simon Crowther	UK
Paris spécialités	Marie Bertrand	France

CSS



CSS Layout

Display

Position

Float

Overflow

CSS Layout - The display Property

The display property is the most important CSS property for controlling layout.

► The display Property

- The display property specifies **if/how** an element is displayed.

► Inline Elements

- An inline element **does not start on a new line** and only takes up as much width as necessary.
-
- <a>
-

► Display: none;

► Block-level Elements

- A block-level element always **starts on a new line and takes up the full width** available (stretches out to the left and right as far as it can).
- <div>
- <h1> - <h6>
- <p>
- <form>
- <header>
- <footer>
- <section>

[Tryit Editor v3.7 \(w3schools.com\)](https://www.w3schools.com/tryit/tryit.asp?filename=tryit_css_display_none)

CSS Layout - The position Property

The position property specifies the type of positioning method used for an element

► The position Property

- There are five different position values:
 1. static
 2. relative
 3. fixed
 4. absolute
 5. Sticky

- Elements are then positioned using the **top**, **bottom**, **left**, and **right** properties. However, these properties will not work unless the position property is set first.

CSS Layout - The position Property

The position property specifies the type of positioning method used for an element

► position: static;

- HTML elements are positioned static by **default**.
- Static positioned elements are **not affected by the top, bottom, left, and right properties**.

This <div> element has position: static;

Here is the CSS that is used:

Example

```
div.static {  
  position: static;  
  border: 3px solid #73AD21;  
}
```

CSS Layout - The position Property

The position property specifies the type of positioning method used for an element

► position: relative;

- An element with position: relative; is positioned **relative to its normal position**.
- Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

This <div> element has position: relative;

Here is the CSS that is used:

Example

```
div.relative {  
  position: relative;  
  left: 30px;  
  border: 3px solid #73AD21;  
}
```

CSS Layout - The position Property

The position property specifies the type of positioning method used for an element

► position: fixed;

- An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element.

```
<!DOCTYPE html>
<html>
<head>
<style>
div.fixed {
  position: fixed;
  top: 300px;
  right: 0;
  width: 300px;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>
<h2>position: fixed;</h2>
<p>An element with position: fixed; is positioned relative to the viewport, which means it
always stays in the same place even if the page is scrolled:</p>
<div class="fixed">
This div element has position: fixed;
</div>
</body>
</html>
```

position: fixed;

An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled:

This div element has position: fixed;

CSS Layout - The position Property

The position property specifies the type of positioning method used for an element

► position: absolute;

- An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).

```
<!DOCTYPE html>
<html>
<head>
<style>
  div.relative {
    position: relative;
    width: 400px;
    height: 200px;
    border: 3px solid #73AD21;
  }

  div.absolute {
    position: absolute;
    top: 80px;
    right: 0;
    width: 200px;
    height: 100px;
    border: 3px solid #73AD21;
  }
</style>
</head>
<body>

<h2>position: absolute;</h2>

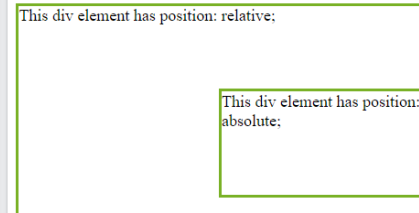
<p>An element with position: absolute; is positioned relative to the nearest positioned
ancestor (instead of positioned relative to the viewport, like fixed):</p>

<div class="relative">This div element has position: relative;
<div class="absolute">This div element has position: absolute;</div>
</div>

</body>
</html>
```

position: absolute;

An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed):



CSS Layout - The position Property

The position property specifies the type of positioning method used for an element

► position: sticky;

- An element with position: sticky; is positioned based on the user's scroll position.
- A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position:fixed).

I am sticky!

Lorem ipsum dolor sit amet, illum definitiones no quo, malisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.

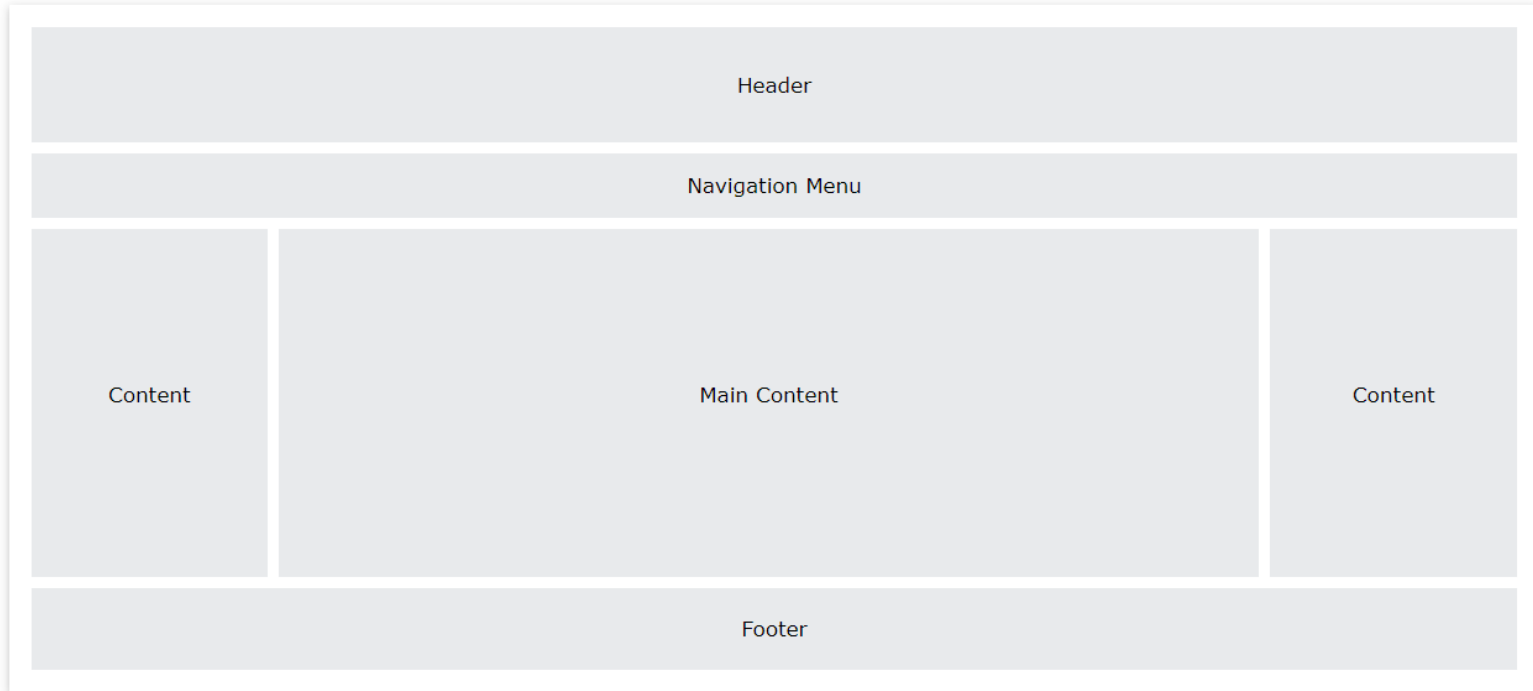
Lorem ipsum dolor sit amet, illum definitiones no quo, malisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.

Example

```
div.sticky {  
  position: -webkit-sticky; /* Safari */  
  position: sticky;  
  top: 0;  
  background-color: green;  
  border: 2px solid #4CAF50;  
}
```

CSS Website Layout

A website is often divided into headers, menus, content and a footer



CSS Layout – float and clear

The CSS float property specifies how an element should float.

► The float Property

- **left** - The element floats to the left of its container
- **right** - The element floats to the right of its container
- **none** - The element does not float (will be displayed just where it occurs in the text). This is default
- **inherit** - The element inherits the float value of its parent

► The clear Property

- When we use the float property, and we want the next element below (not on right or left), we will have to use the clear property.
- **none** - The element is not pushed below left or right floated elements. This is default
- **left** - The element is pushed below left floated elements
- **right** - The element is pushed below right floated elements
- **both** - The element is pushed below both left and right floated elements
- **inherit** - The element inherits the clear value from its parent

[Tryit Editor v3.7 \(w3schools.com\)](https://www.w3schools.com/tryit/tryit.asp?filename=tryit_css_float_clear)

CSS Layout - Overflow

The CSS overflow property controls what happens to content that is too big to fit into an area.

- ▶ The overflow property has the following values:
 - **visible** - **Default**. The overflow is not clipped. The content renders outside the element's box
 - **hidden** - The overflow is clipped, and the rest of the content will be invisible
 - **scroll** - The overflow is clipped, and a scrollbar is added to see the rest of the content
 - **auto** - Similar to scroll, but it adds scrollbars only when necessary

Visible

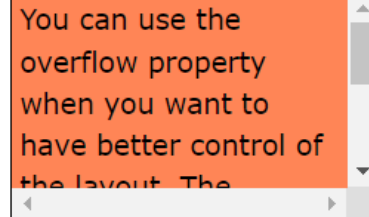
You can use the overflow property when you want to have better control of the layout. The overflow property specifies what happens if content overflows an element's box.

hidden

You can use the overflow property when you want to have better control of the layout. The overflow property specifies what

scroll

You can use the overflow property when you want to have better control of the layout. The



CSS Vertical Navigation Bar

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  margin: 0;
}

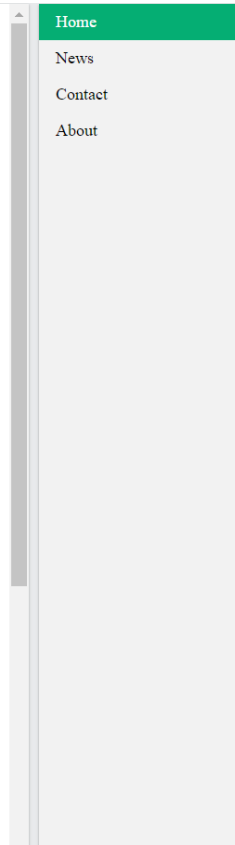
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  width: 25%;
  background-color: #f1f1f1;
  position: fixed;
  height: 100%;
  overflow: auto;
}

li a {
  display: block;
  color: #000;
  padding: 8px 16px;
  text-decoration: none;
}

li a.active {
  background-color: #04AA6D;
  color: white;
}

li a:hover:not(.active) {
  background-color: #555;
  color: white;
}
</style>
</head>
<body>

<ul>
<li><a class="active" href="#home">Home</a></li>
<li><a href="#news">News</a></li>
<li><a href="#contact">Contact</a></li>
<li><a href="#about">About</a></li>
</ul>
```



Fixed Full-height Side Nav

Try to scroll this area, and see how the sidenav sticks to the page

Notice that this div element has a left margin of 25%. This is because the side navigation is set to 25% width. If you remove the margin, the sidenav will overlay/sit on top of this div.

Also notice that we have set `overflow:auto` to sidenav. This will add a scrollbar when the sidenav is too long (for example if it has over 50 links inside of it).

Some text..

Some text..

Some text..

Some text..

Some text..

Some text..

Some text..

CSS Horizontal Navigation Bar

```
<html>
<head>
<style>
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
  background-color: #333;
}

li {
  float: left;
}

li a {
  display: block;
  color: white;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
}

li a:hover:not(.active) {
  background-color: #111;
}

.active {
  background-color: #04AA6D;
}
</style>
</head>
<body>

<ul>
<li><a href="#home">Home</a></li>
<li><a href="#news">News</a></li>
<li><a href="#contact">Contact</a></li>
<li style="float:right"><a class="active" href="#about">About</a></li>
</ul>

</body>
</html>
```

Home News Contact

About

Summary

Module 04

- ▶ learning about Fonts
- ▶ learning about Box Model
- ▶ learning about font properties with CSS
- ▶ learning about size
- ▶ learning about weight
- ▶ learning about text alignment