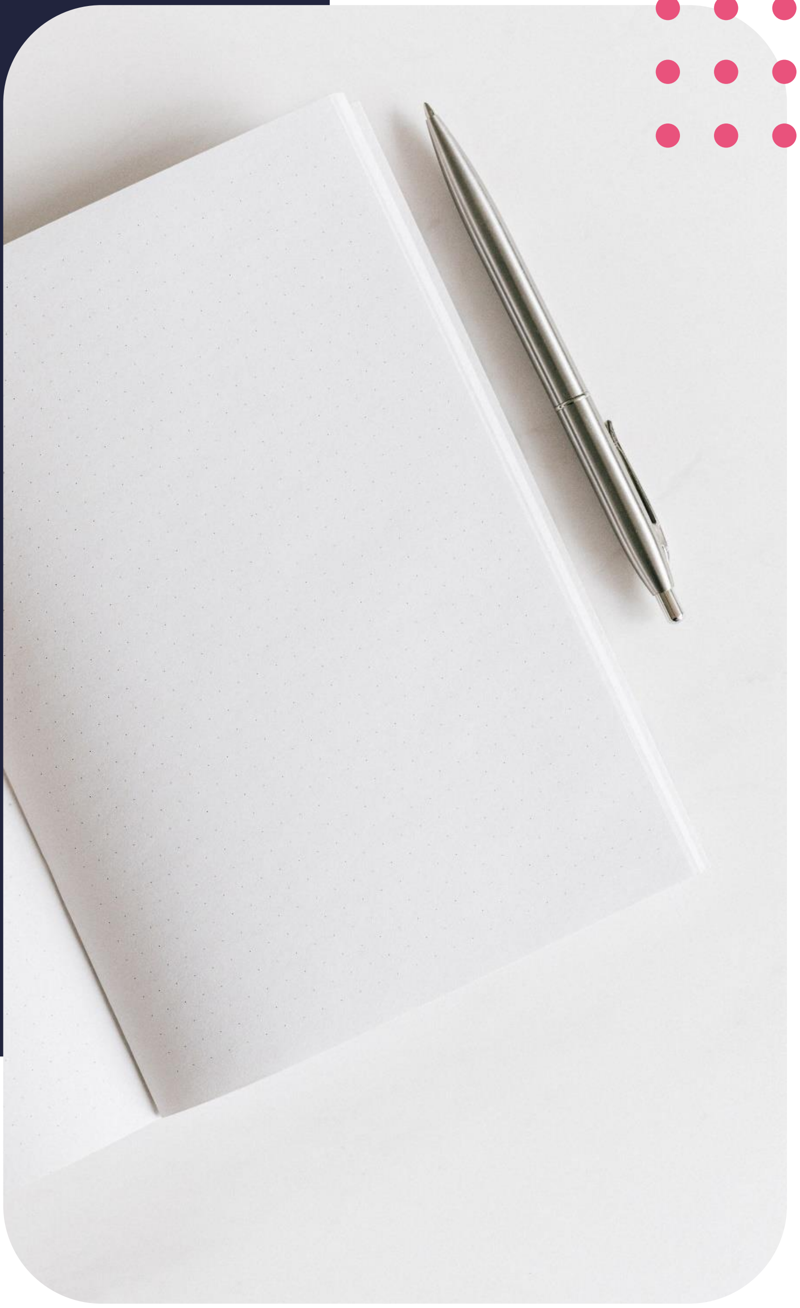
Diploma in Web Development

Advanced CSS

Contents



Lesson outcomes

In this lesson we will discuss the CSS box model and learn all about how we can use padding, margin and borders to style our elements. This will greatly improve the layout and dimensions of our elements. We will also learn all about positioning our styled elements in our viewport and working with animations. To wrap things up we will put everything we've learned so far into practice.

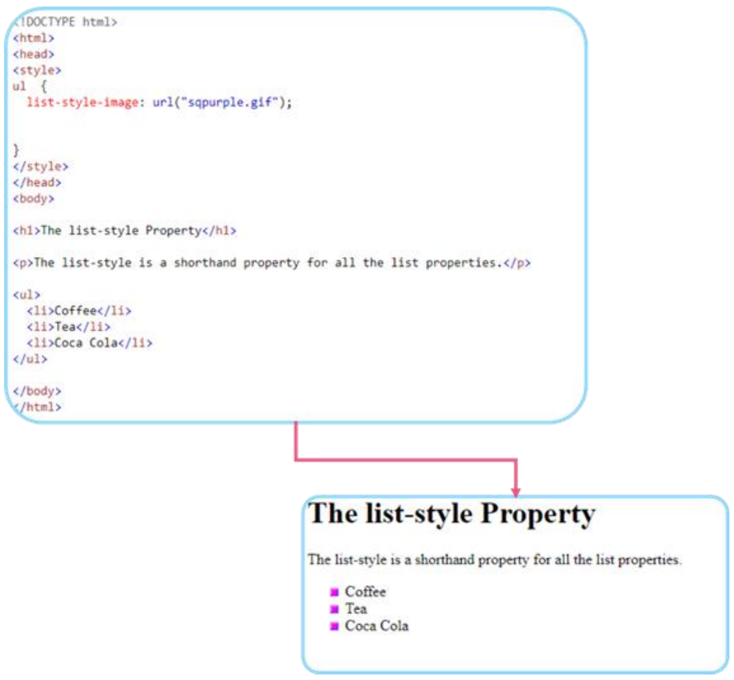
Styling the navigation bar

Formatting lists

The default browser style for ordered and unordered lists is to display each list item alongside a symbol known as a list marker. By default, unordered lists are displayed with a solid disc, while ordered lists are displayed with numerals. To alter the list marker property, use the list-style-type property.

Using images for list markers

Instead of using the predefined list markers, developers can also opt for using custom graphics. If the graphic can't be displayed, the default list marker will be shown. Graphics as list markers only work for unordered lists.



```
<!DOCTYPE html>
<html>
<head>
<style>
ul  {
  list-style-image: url("sqpurple.gif");


}
</style>
</head>
<body>

<h1>The list-style Property</h1>

<p>The list-style is a shorthand property for all the list properties.</p>

<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>

</body>
</html>
```

The list-style Property

The list-style is a shorthand property for all the list properties.

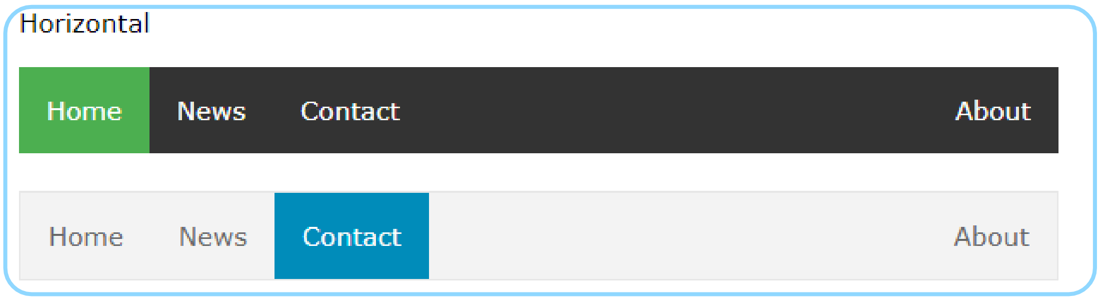
- Coffee
- Tea
- Coca Cola

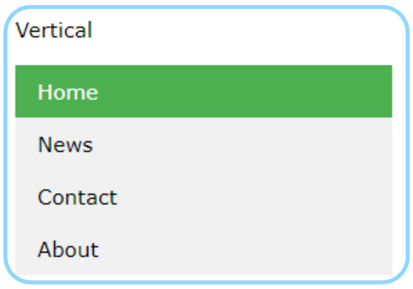
Styling the navigation bar

A navigation bar is one of the key elements used in websites. A navigation bar needs an HTML base which is transformed with CSS.

- A navigation bar is basically a styled unordered list <ul> and its accompanying list items <li>
- To remove list markers, use list-style-type: none; property.
- The display: inline; property aligns list items horizontal for more natural navigation bar look.
- To create a vertical navigation bar, use the display: block; property.
- To space list items, use the padding and margin properties.

Example of styled navigation bars





(Source: w3schools)

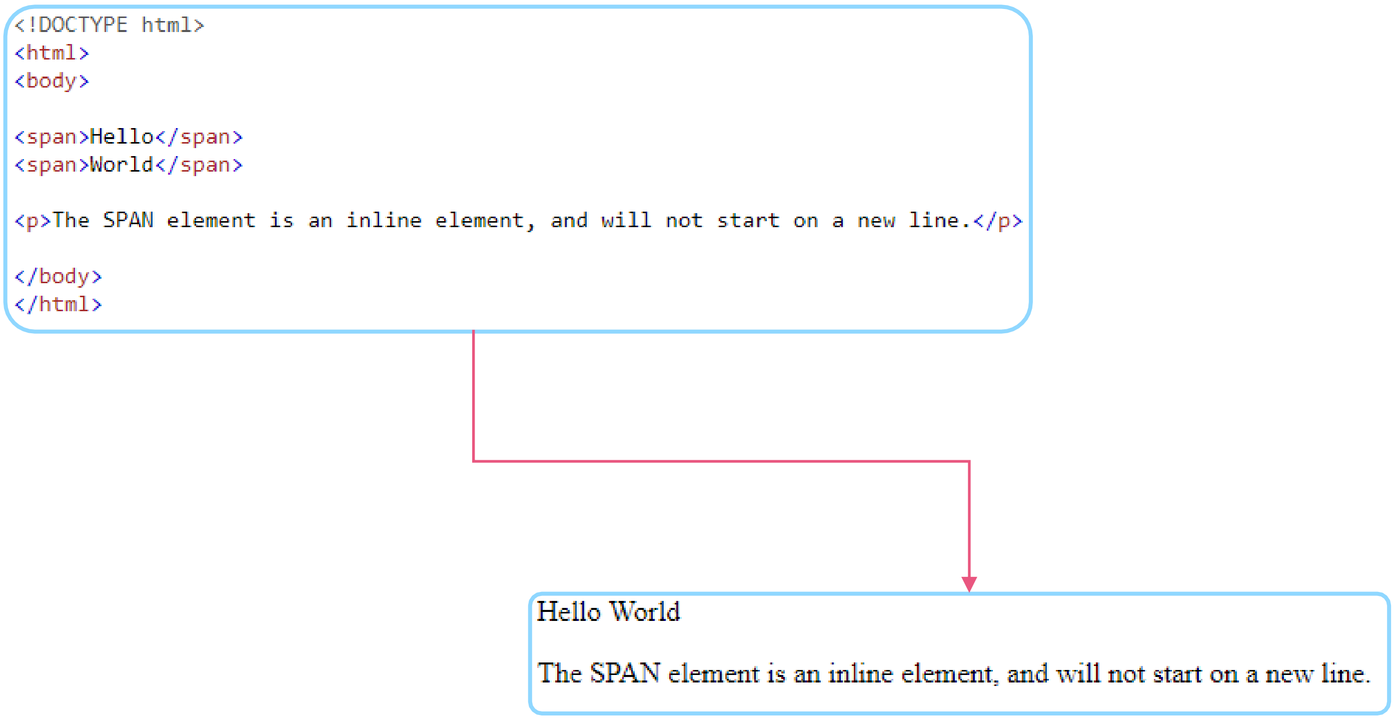
List style properties

Value	Details
list-style	Specifies all the list properties in one declaration
list-style-position	Specifies where to place the list-item marker.
list-style-image	Specifies the type of list-item marker.
list-style-type	Specifies the type of list-item marker.

Types of elements

Block level elements	**Inline level elements**
• HTML block level elements can appear in the body of an HTML page. • It can contain another block level as well as inline elements. • By default, block-level elements begin on new lines. • Block level elements create larger structures (than inline elements)	• HTML inline level elements can appear in the body of an HTML page. • It can contain data and other inline elements. • By default, inline elements do not begin on new lines. • Inline elements create shorter structures (than block level elements).

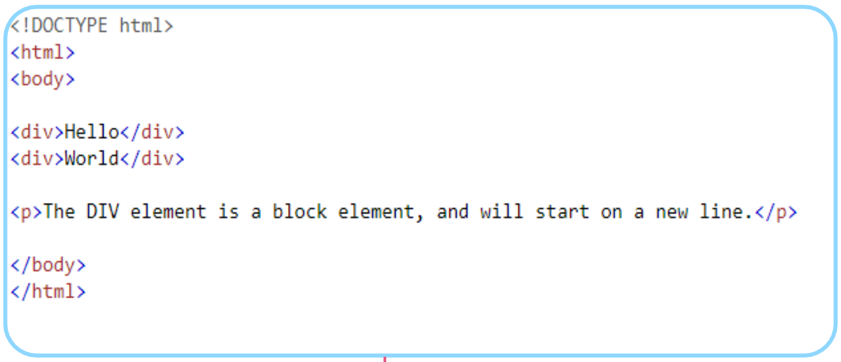
Inline level elements



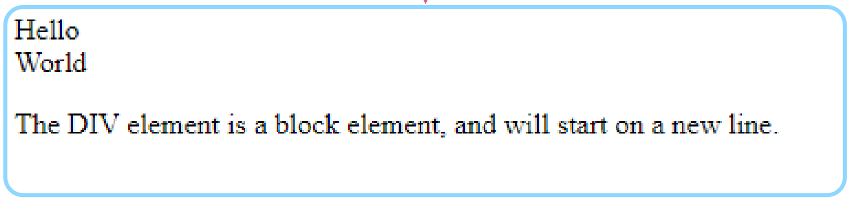
```
<!DOCTYPE html>
<html>
<body>

<span>Hello</span>
<span>World</span>

<p>The SPAN element is an inline element, and will not start on a new line.</p>

</body>
</html>
```

```
Hello World

The SPAN element is an inline element, and will not start on a new line.
```

Block level elements



```
<!DOCTYPE html>
<html>
<body>

<div>Hello</div>
<div>World</div>

<p>The DIV element is a block element, and will start on a new line.</p>

</body>
</html>
```



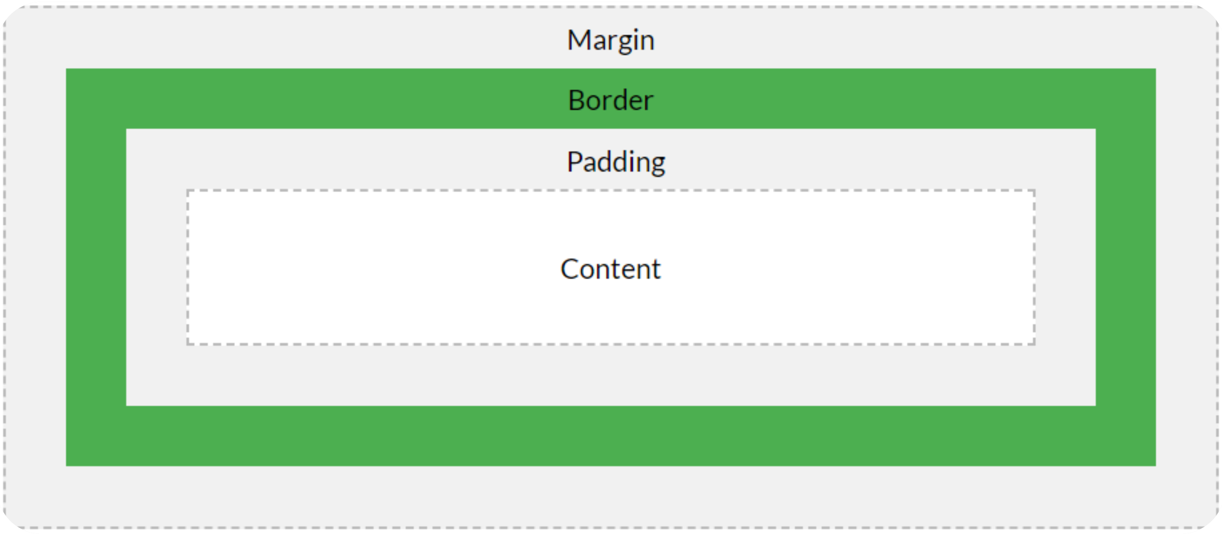
Hello
World

The DIV element is a block element, and will start on a new line.

CSS box model

CSS box model layout

The CSS box model is essentially a box that wraps around every HTML element.

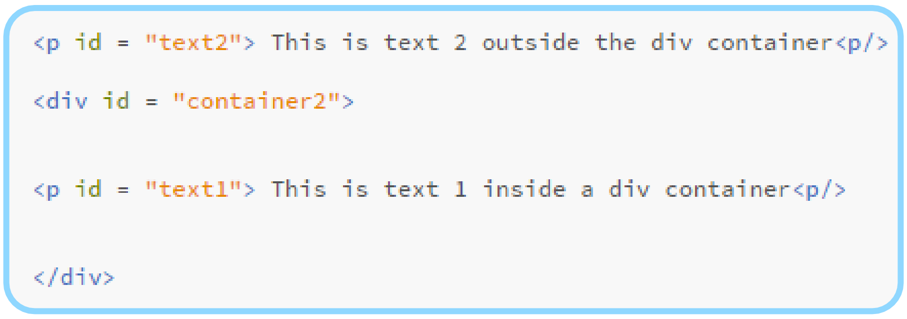


(Source: w3schools)

- **Content** - The content of the box, where text and images appear.
- **Padding** - Clears an area around the content. The padding is transparent. Extends from the content to the border.
- **Border** - A border surrounds the padding space and content.
- **Margin** - Clears an outside area beyond the border. Up to surrounding elements - the margin is transparent.

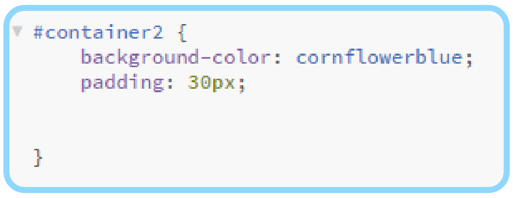
Padding

HTML

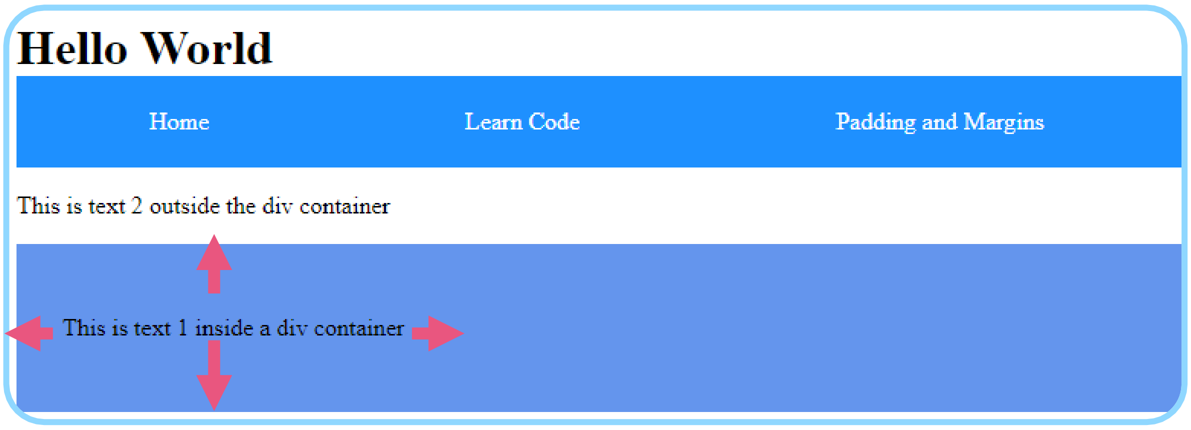


```html
<p id = "text2"> This is text 2 outside the div container<p/>

<div id = "container2">


<p id = "text1"> This is text 1 inside a div container<p/>


</div>
```

CSS



```css
#container2 {
    background-color: cornflowerblue;
    padding: 30px;

}
```



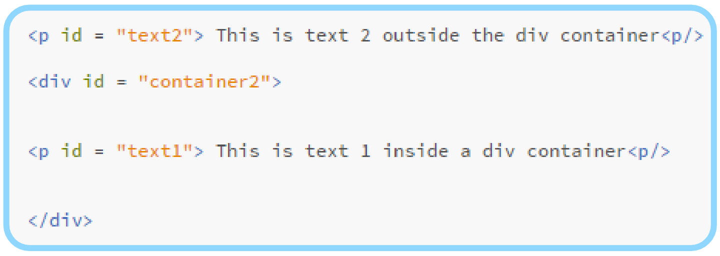
Hello World

This is text 2 outside the div container

This is text 1 inside a div container

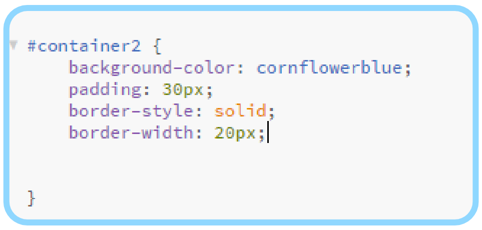
Borders

HTML

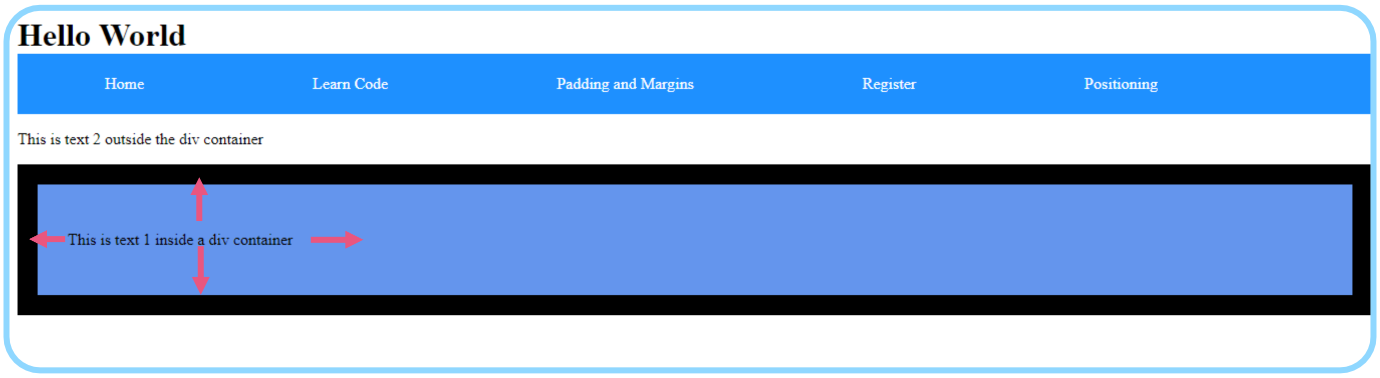


```
<p id = "text2"> This is text 2 outside the div container<p/>

<div id = "container2">


<p id = "text1"> This is text 1 inside a div container<p/>


</div>
```

CSS



```
#container2 {
    background-color: cornflowerblue;
    padding: 30px;
    border-style: solid;
    border-width: 20px;


}
```



Hello World

This is text 2 outside the div container

This is text 1 inside a div container

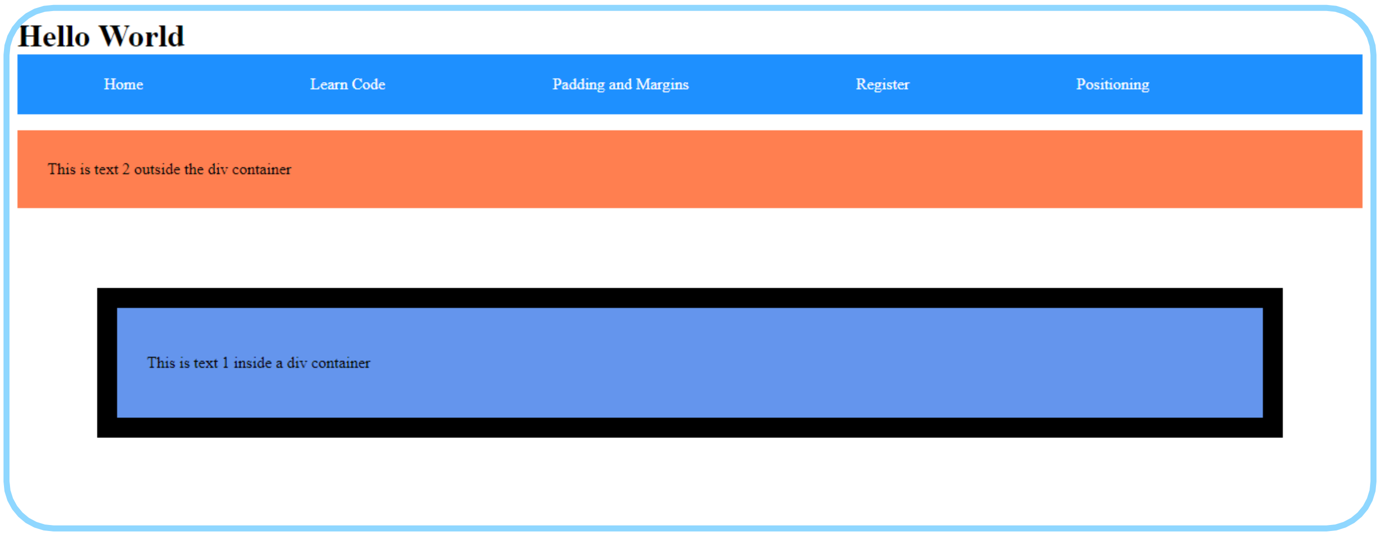
Margin

HTML

```
<p id = "text2"> This is text 2 outside the div container<p/>

<div id = "container2">


<p id = "text1"> This is text 1 inside a div container<p/>


</div>
```

CSS

```
#text2 {
    background-color: coral;
    padding:30px;

}

#container2 {
    background-color: cornflowerblue;
    padding: 30px;
    border-style: solid;
    border-width: 20px;
    margin: 80px;


}
```



Hello World

This is text 2 outside the div container

This is text 1 inside a div container

Positioning in CSS

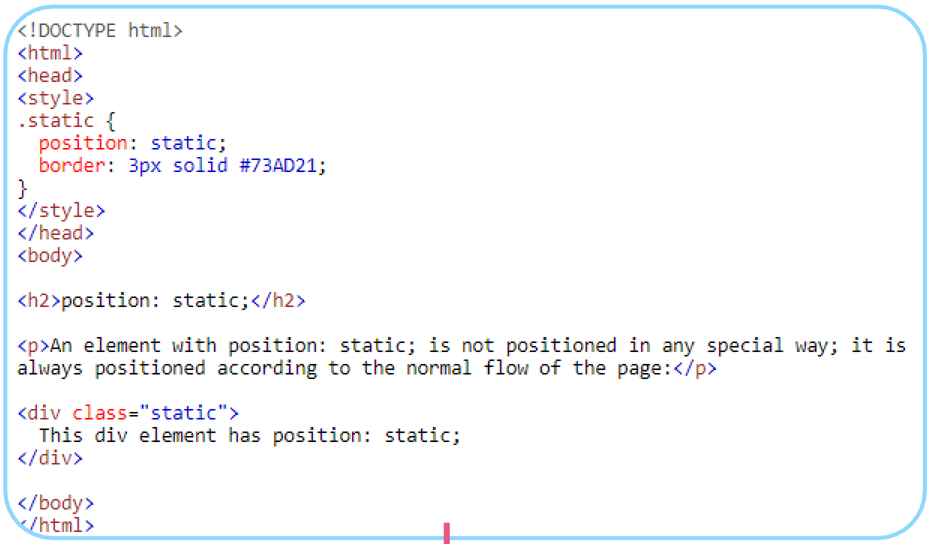
Positioning in elements

The positioning property specifies the type of positioning method used for an element. Elements are positioned using the ***top, bottom, left, and right*** values.

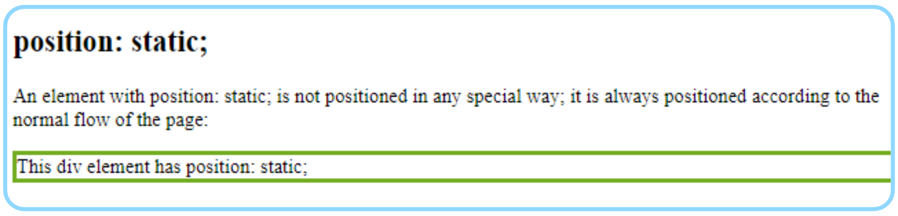
Types of positioning

- **Static** - Elements are positioned static by default and not affected by the top, bottom, left, and right properties.
- **Relative** - Positioned relative to its normal position.
- **Fixed** - Positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled.
- **Sticky** - Positioned based on the user's scroll position. Toggles between relative and fixed until a given offset position is met in the viewport.
- **Absolute** - Positioned relative to the nearest positioned ancestor.

Static



```html
<!DOCTYPE html>
<html>
<head>
<style>
.static {
  position: static;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>

<h2>position: static;</h2>

<p>An element with position: static; is not positioned in any special way; it is
always positioned according to the normal flow of the page:</p>

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

</body>
</html>
```

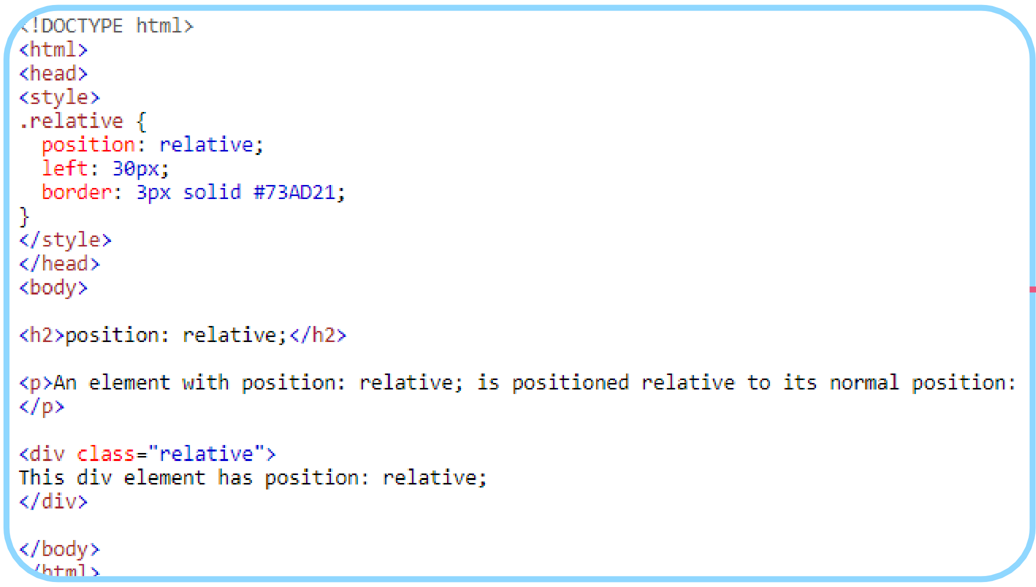


position: static;

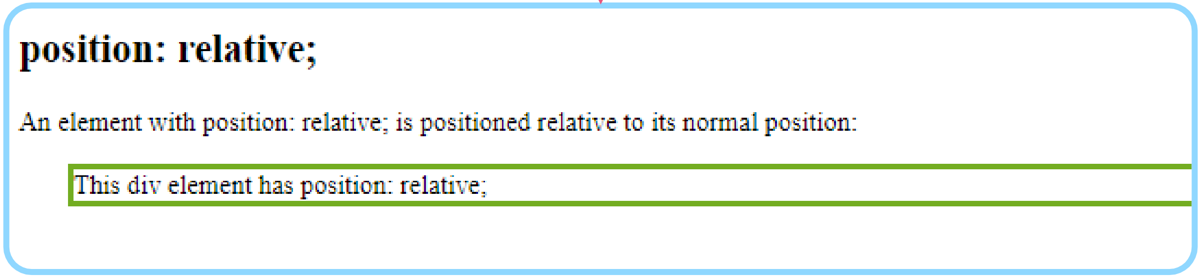
An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page:

This div element has position: static;

Relative



```
<!DOCTYPE html>
<html>
<head>
<style>
.relative {
  position: relative;
  left: 30px;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>

<h2>position: relative;</h2>

<p>An element with position: relative; is positioned relative to its normal position:
</p>

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

</body>
</html>
```

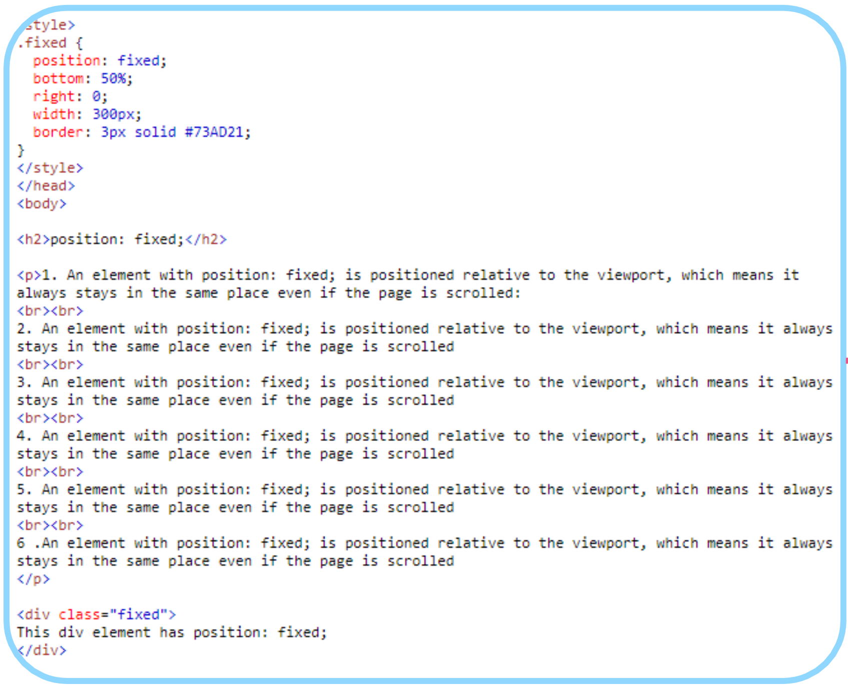


position: relative;

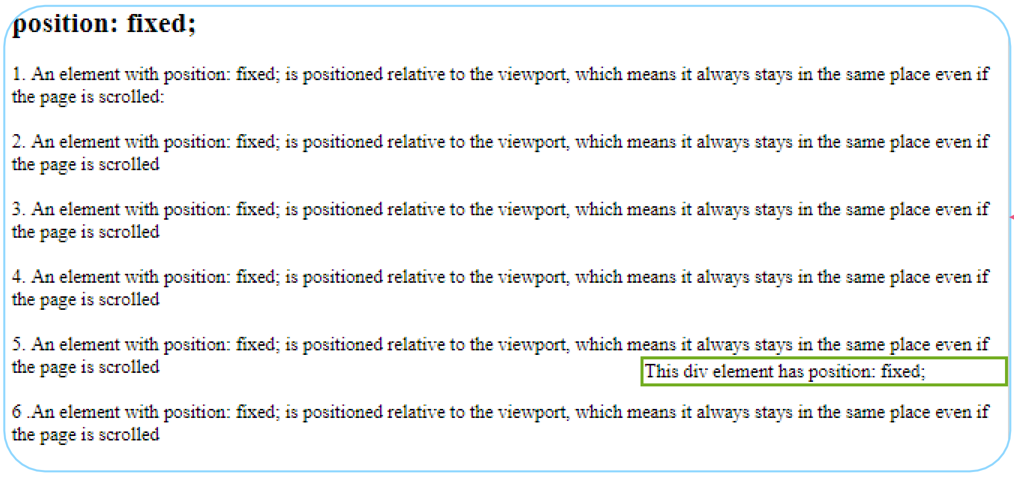
An element with position: relative; is positioned relative to its normal position:

This div element has position: relative;

Fixed



```
<style>
.fixed {
  position: fixed;
  bottom: 50%;
  right: 0;
  width: 300px;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>

<h2>position: fixed;</h2>

<p>1. 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:
<br><br>
2. 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
<br><br>
3. 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
<br><br>
4. 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
<br><br>
5. 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
<br><br>
6 .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>
```



position: fixed;

1. 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:

2. 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

3. 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

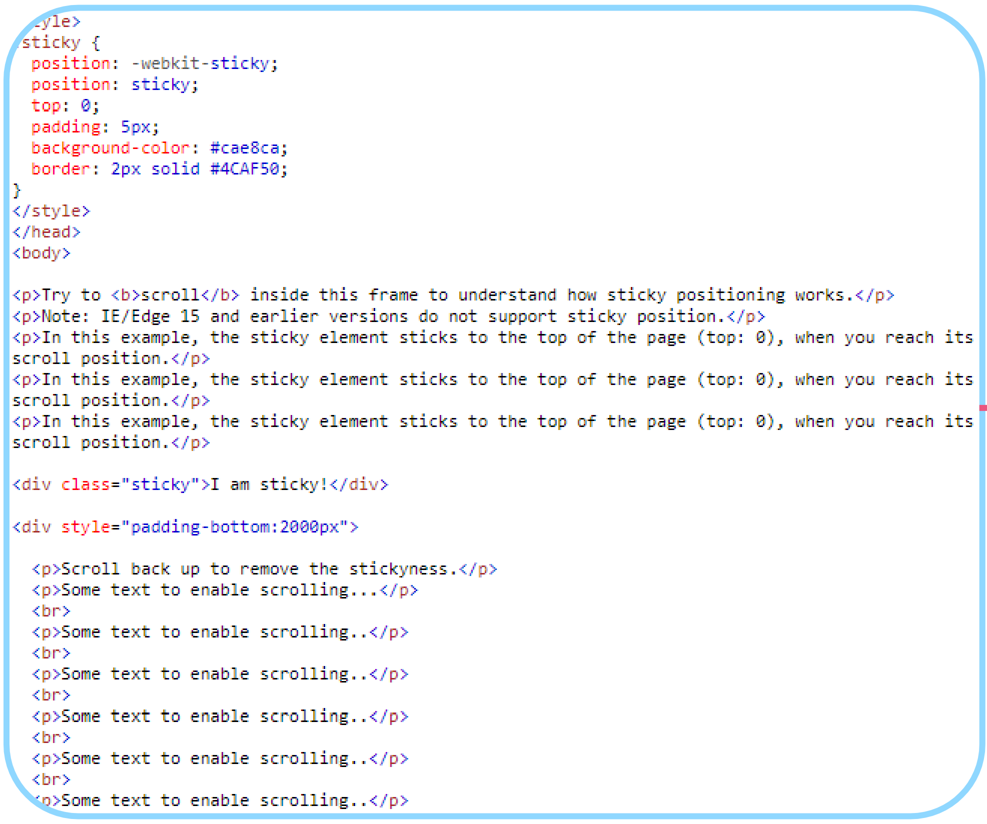
4. An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled

5. 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

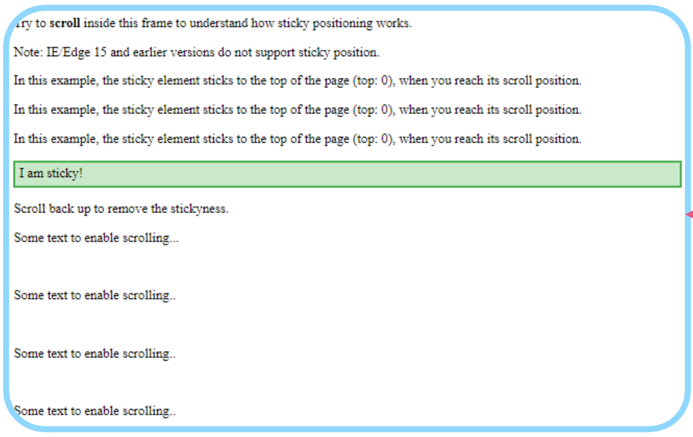
This div element has position: fixed;

6 .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

Sticky



```
yle>
sticky {
  position: -webkit-sticky;
  position: sticky;
  top: 0;
  padding: 5px;
  background-color: #cae8ca;
  border: 2px solid #4CAF50;
}
</style>
</head>
<body>

<p>Try to <b>scroll</b> inside this frame to understand how sticky positioning works.</p>
<p>Note: IE/Edge 15 and earlier versions do not support sticky position.</p>
<p>In this example, the sticky element sticks to the top of the page (top: 0), when you reach its
scroll position.</p>
<p>In this example, the sticky element sticks to the top of the page (top: 0), when you reach its
scroll position.</p>
<p>In this example, the sticky element sticks to the top of the page (top: 0), when you reach its
scroll position.</p>

<div class="sticky">I am sticky!</div>

<div style="padding-bottom:2000px">

  <p>Scroll back up to remove the stickyness.</p>
  <p>Some text to enable scrolling...</p>
  <br>
  <p>Some text to enable scrolling..</p>
  <br>
  <p>Some text to enable scrolling..</p>
  <br>
  <p>Some text to enable scrolling..</p>
  <br>
  <p>Some text to enable scrolling..</p>
  <br>
  p>Some text to enable scrolling..</p>
```



ry to **scroll** inside this frame to understand how sticky positioning works.

Note: IE/Edge 15 and earlier versions do not support sticky position.

In this example, the sticky element sticks to the top of the page (top: 0), when you reach its scroll position.

In this example, the sticky element sticks to the top of the page (top: 0), when you reach its scroll position.

In this example, the sticky element sticks to the top of the page (top: 0), when you reach its scroll position.

I am sticky!

Scroll back up to remove the stickyness.

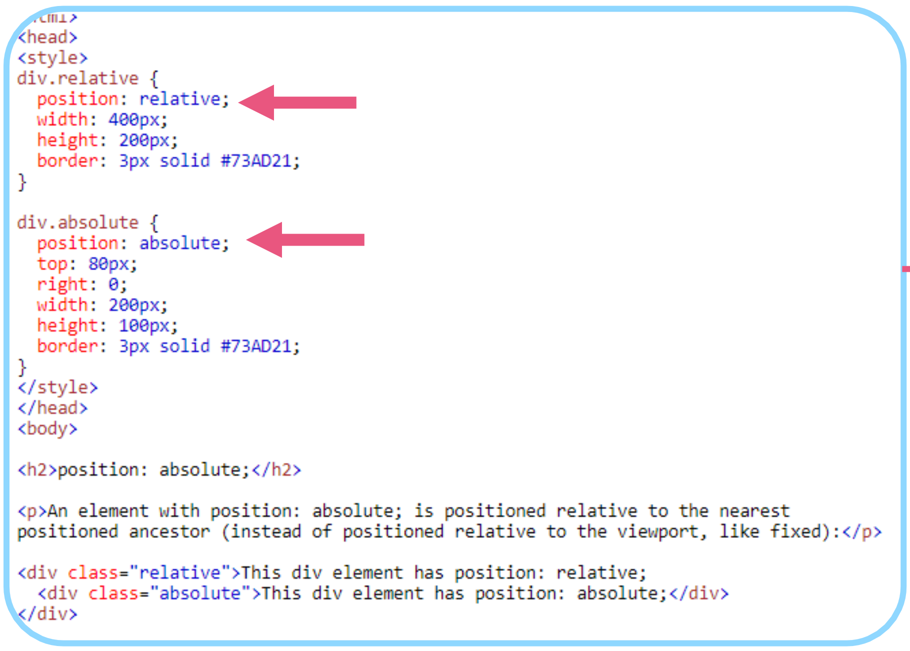
Some text to enable scrolling...

Some text to enable scrolling..

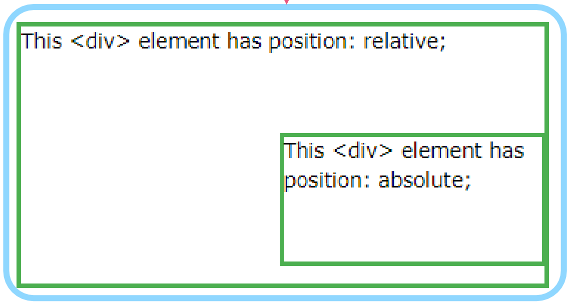
Some text to enable scrolling..

Some text to enable scrolling..

Absolute



```
tml>
<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>
```



This <div> element has position: relative;

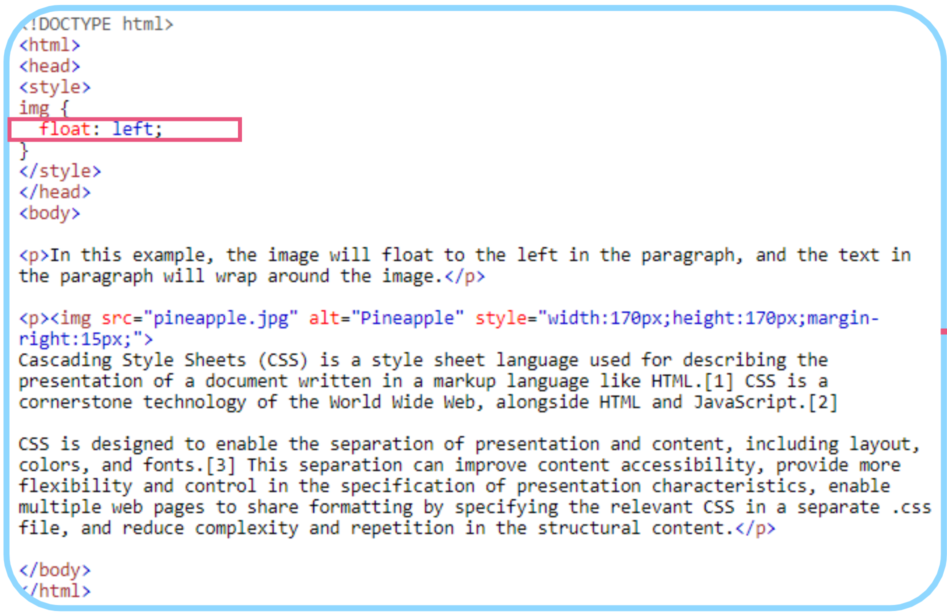
This <div> element has position: absolute;

CSS float

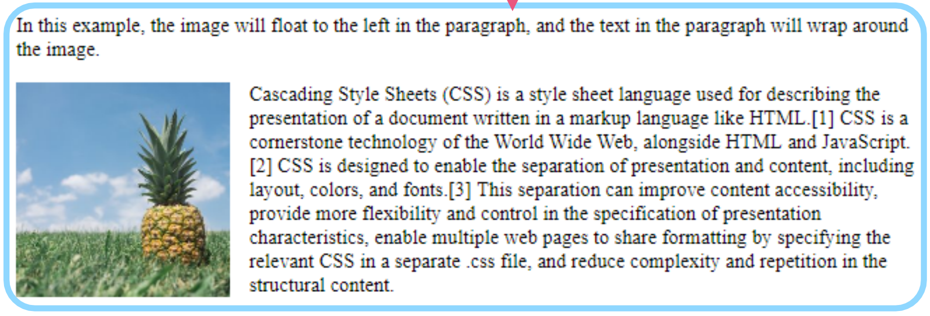
The CSS ***float*** property specifies how an element should float and is used for positioning elements such as images or containers.

Float properties

Property value	**Details**
left	Element floats to the left of its container
right	Element floats to the right of its container
inherit	The element inherits the float value of its parent
none	The element does not float. Remains in default position



```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  float: left;
}
</style>
</head>
<body>

<p>In this example, the image will float to the left in the paragraph, and the text in
the paragraph will wrap around the image.</p>

<p><img src="pineapple.jpg" alt="Pineapple" style="width:170px;height:170px;margin-
right:15px;">
Cascading Style Sheets (CSS) is a style sheet language used for describing the
presentation of a document written in a markup language like HTML.[1] CSS is a
cornerstone technology of the World Wide Web, alongside HTML and JavaScript.[2]

CSS is designed to enable the separation of presentation and content, including layout,
colors, and fonts.[3] This separation can improve content accessibility, provide more
flexibility and control in the specification of presentation characteristics, enable
multiple web pages to share formatting by specifying the relevant CSS in a separate .css
file, and reduce complexity and repetition in the structural content.</p>

</body>
</html>
```



In this example, the image will float to the left in the paragraph, and the text in the paragraph will wrap around the image.

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML.[1] CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.[2] CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts.[3] This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

Animations in CSS

Working with animations

CSS animations allow dynamic changes of most HTML elements without using JavaScript. An animation lets an element gradually change from one style or state to another.

To use animations, we must call ***@keyframes*** in CSS.

Animation properties

Property	**Details**
animation-name	Specifies the name of the keyframe you want to bind to the selector.
animation-duration	Specifies how many seconds or milliseconds an animation takes to complete.
animation-delay	Specifies a delay before the animation will start.
animation-iteration-count	Specifies how many times an animation should be played.
animation-direction	Specifies whether the animation should play in reverse on alternate cycles.
animation-fill-mode	Specifies what values are applied by the animation outside the time it is executing.
animation-play-state	Specifies whether the animation is running or paused.

Animating colours and borders

HTML

```html
<div id = "square"></div>
```

CSS

```css
square {
  width: 100px;
  height: 100px;
  border-style:solid;
  background-color: red;
  -webkit-animation-name: square-anim; /* Safari 4.0 - 8.0 */
  -webkit-animation-duration: 4s; /* Safari 4.0 - 8.0 */
  animation-name: square-anim;
  animation-duration: 4s;
}

/* Safari 4.0 - 8.0 */
@-webkit-keyframes square-anim {
  0%   {background-color: red;}
  25%  {background-color: yellow;}
  50%  {background-color: skyblue;}
  100% {background-color: greenyellow;}
}

/* Standard syntax */
@keyframes square-anim {
  0%   {background-color: red; border-radius:0px;}
  25%  {background-color: yellow; border-radius:20px;}
  50%  {background-color: blue; border-radius:40px;}
  100% {background-color: green; border-radius:60px;}
}
```

Animating size

HTML

```html
<div id = "square2"></div>
```

CSS

```css
#square2 {
width: 100px;
height: 100px;
margin-top: 5%;
float:right;
background-color: cornflowerblue;
animation-name: square-anim2;
animation-duration: 5s;
animation-iteration-count: 3;
}

@keyframes square-anim2 {
  0%   {background-color: orange; border-radius:0px;}
  25%  {background-color: lightsteelblue; border-radius:20px; width:200px ; height:200px}
  50%  {background-color: blue; border-radius:40px; width:250px ; height:250px}
  100% {background-color: green; border-radius:60px; width:300px ; height:300px}

}
```

Animating position

HTML

```
<div id = "square3"></div>
```

CSS

```
#square3 {
width: 100px;
height: 100px;
margin-top: 5%;
background-color: orange;
    animation-name: square-anim3;
  animation-duration: 5s;
    animation-iteration-count: 3;
    position: relative;

}

@keyframes square-anim3 {
  0%   {background-color:greenyellow; left:12.5%;}
  25%  {left:25%;}
  50%  {left:37.5%;}
  100% {background-color:skyblue; left:50%;}

}
```

References

w3schools. CSS Box Model. Retrieved January 19, 2021, from https://www.w3schools.com/css/css_boxmodel.asp

w3schools. CSS Navigation Bar. Retrieved January 19, 2021, from https://www.w3schools.com/css/css_navbar.asp