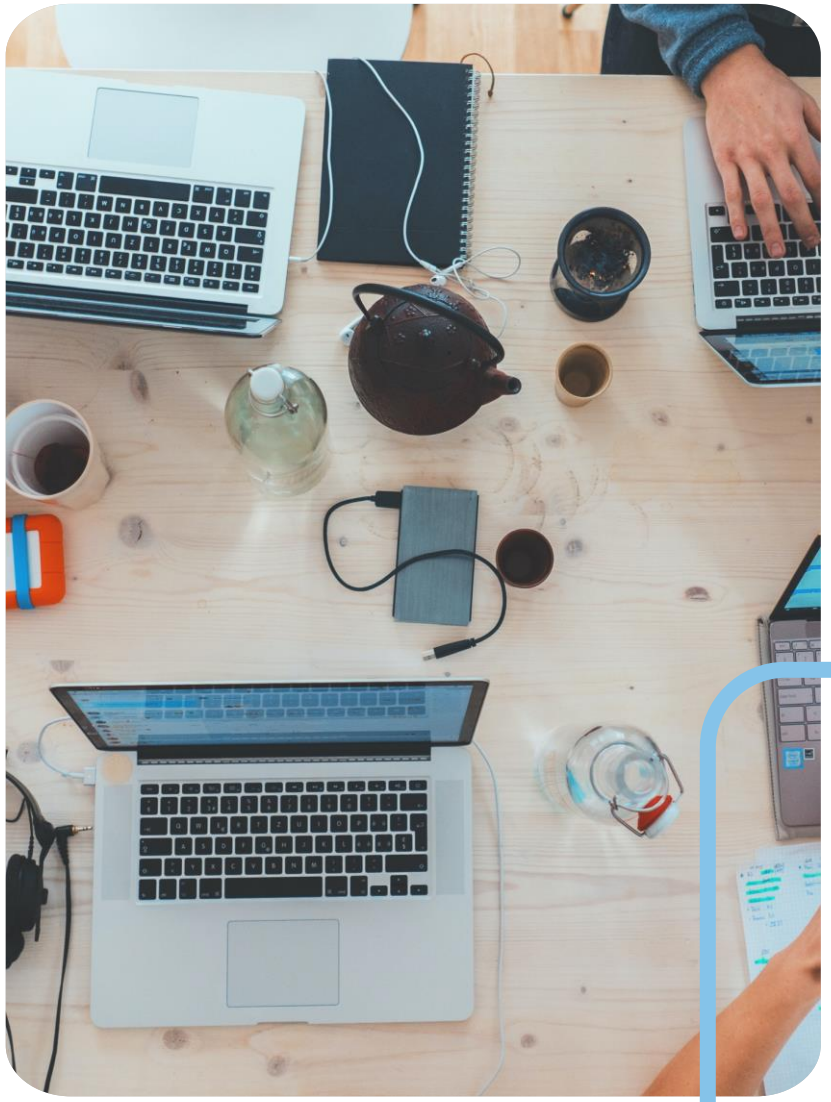
Diploma in

Web Development

Advanced CSS

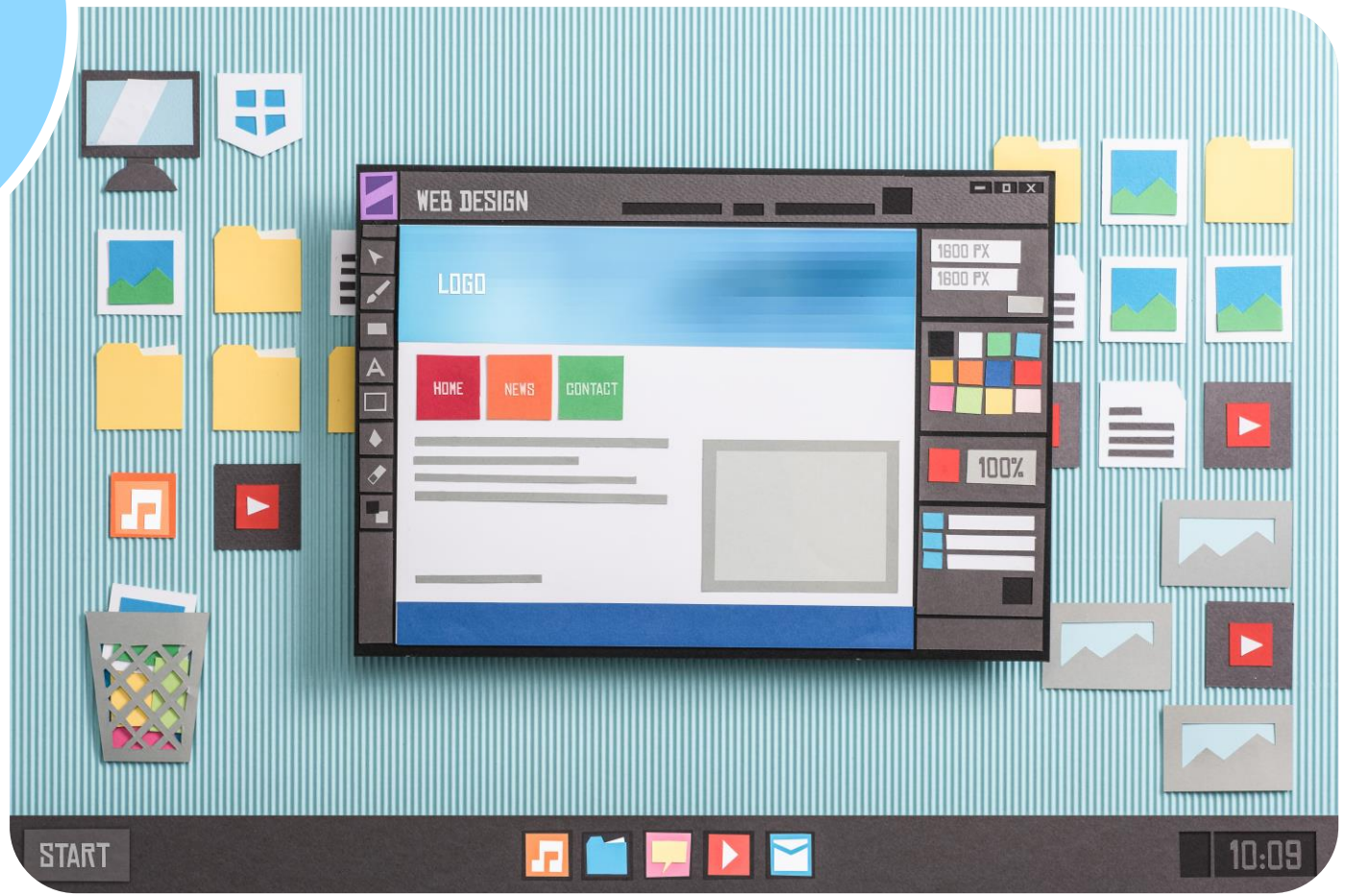


Objectives

- Styling the navigation bar
- CSS box model
- Positioning in CSS
- Animations in CSS

TIP OF THE DAY

Avoid using templates in the web development process.



Formatting lists

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.

```
<!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>
```

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.

Styling the navigation bar

A navigation bar is basically a styled unordered list

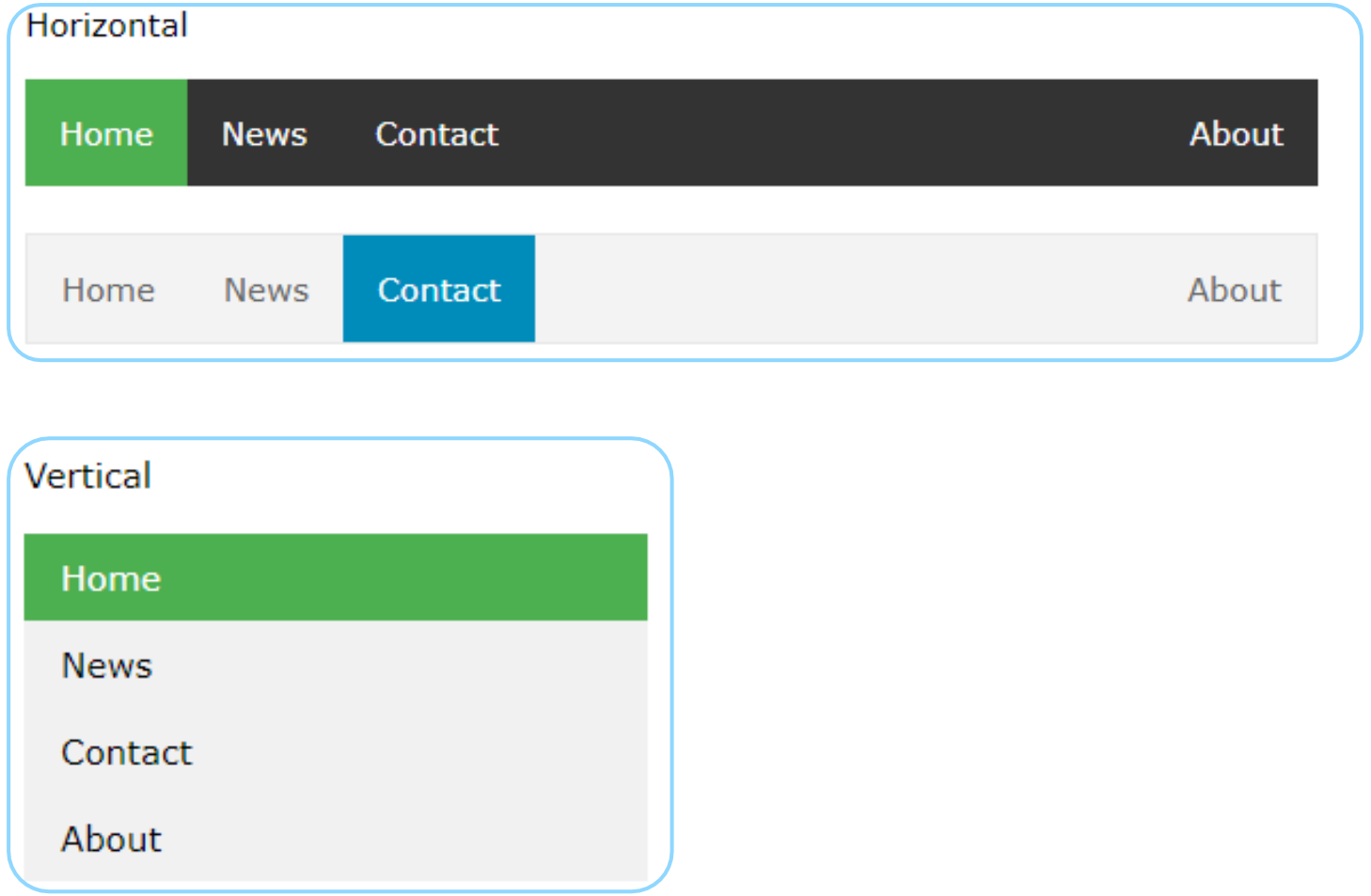
To remove list markers, use *list-style-type:none;*

The *display: inline;* property aligns list items horizontal

To create a vertical navigation bar, use *display: block;*

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

Inline

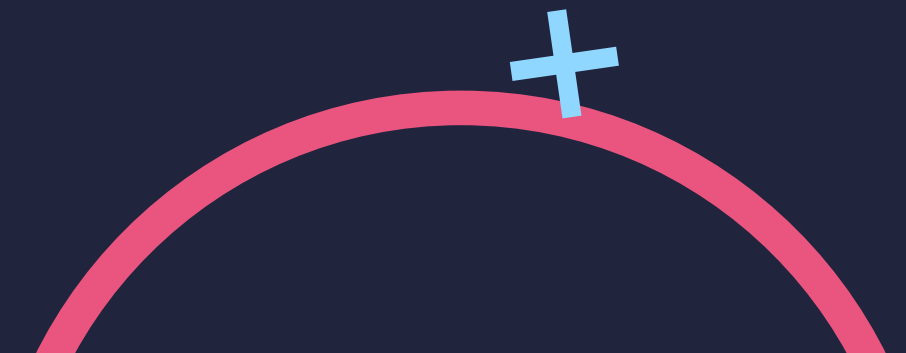
Block level elements

Can appear in the body of an HTML page.

Can contain other block level as well as inline elements.

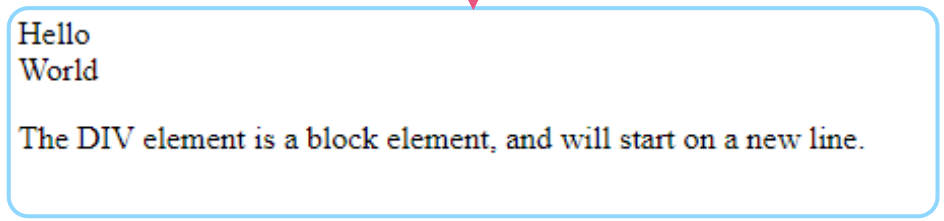
Block level elements begin on new lines.

Block level elements create larger structures



Block level elements

```html
<!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>
```



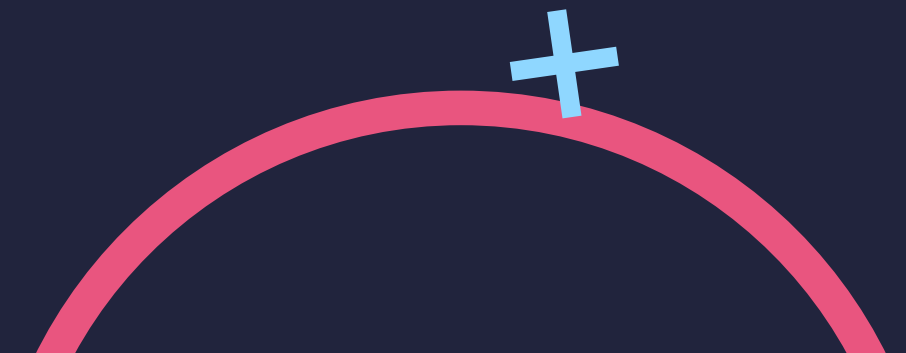
Inline level elements

Can appear in the body of an HTML page.

Can contain data and other inline elements.

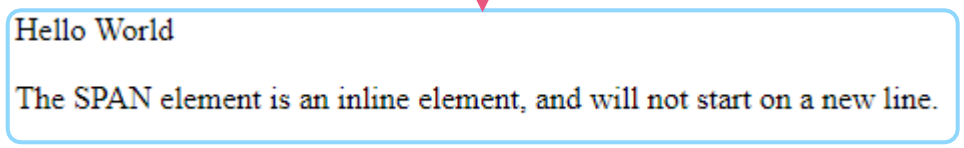
Inline elements do not begin on new lines.

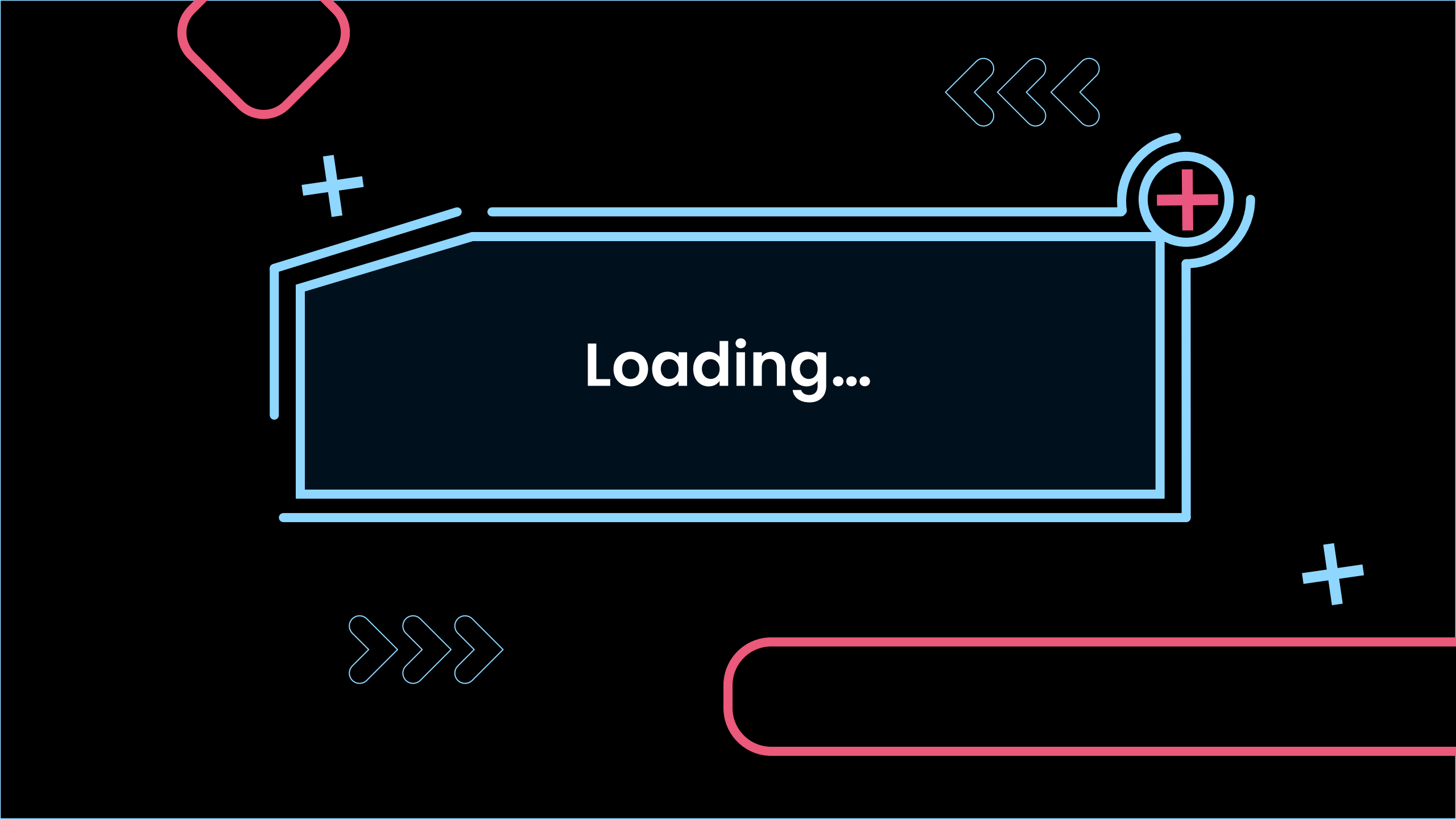
Inline elements create shorter structures



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>
```

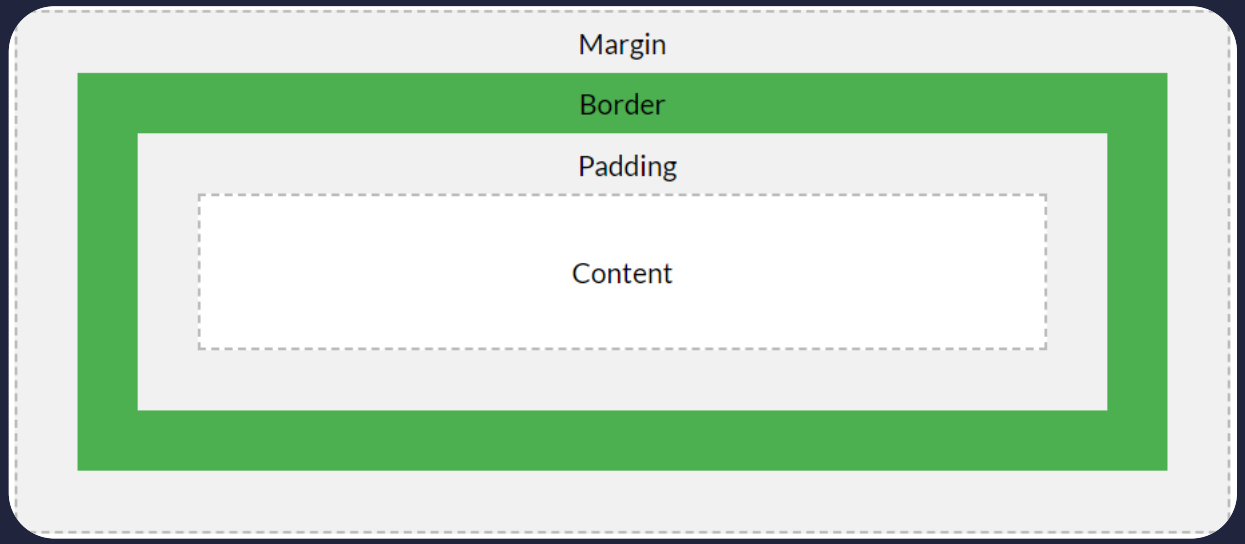



Loading...

CSS box model

The CSS box model is essentially a box that wraps around every HTML element. It consists of:

- padding
- borders
- margins
- content



(Source:w3schools)

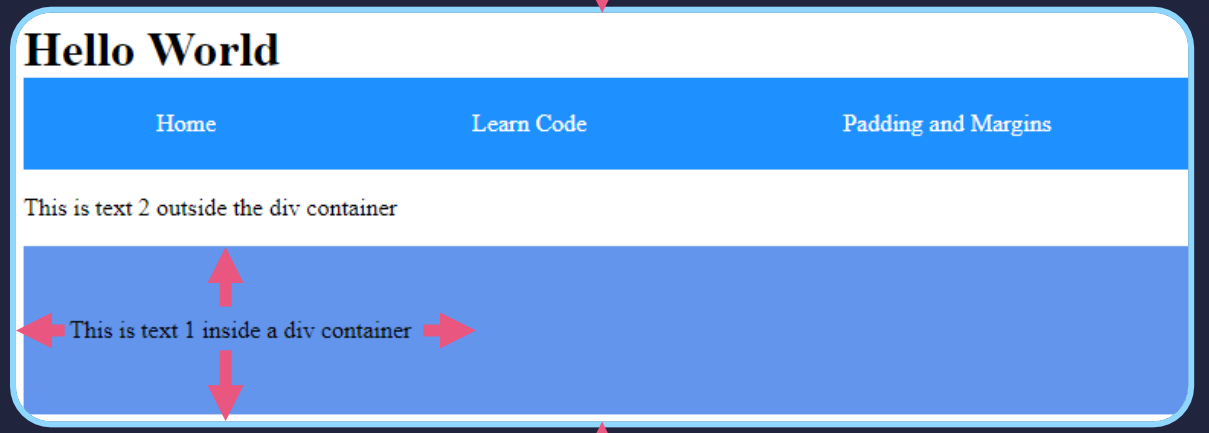
Padding, borders, margins and content

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

Padding

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;


}
```

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;


}
```

Margin

CSS

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

}

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


}
```

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>
```

Positioning in CSS

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	Relative	Fixed	Sticky	Absolute
Elements are positioned static by default.	Positioned relative to its normal position.	Always stays in the same place even if the page is scrolled.	Positioned based on the user's scroll position.	Positioned relative to the nearest positioned ancestor.

Static positioning

```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 positioning

```html
<!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>
```

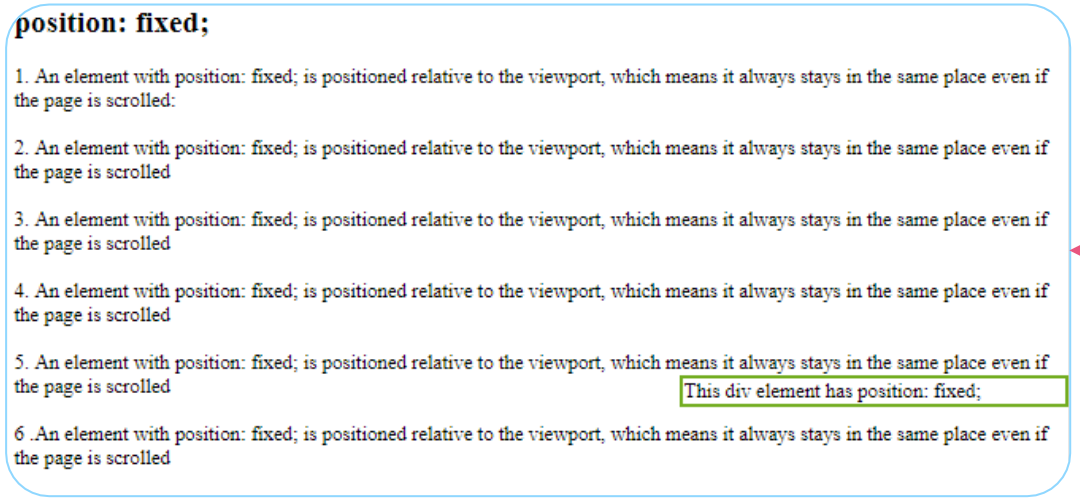
position: relative;

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

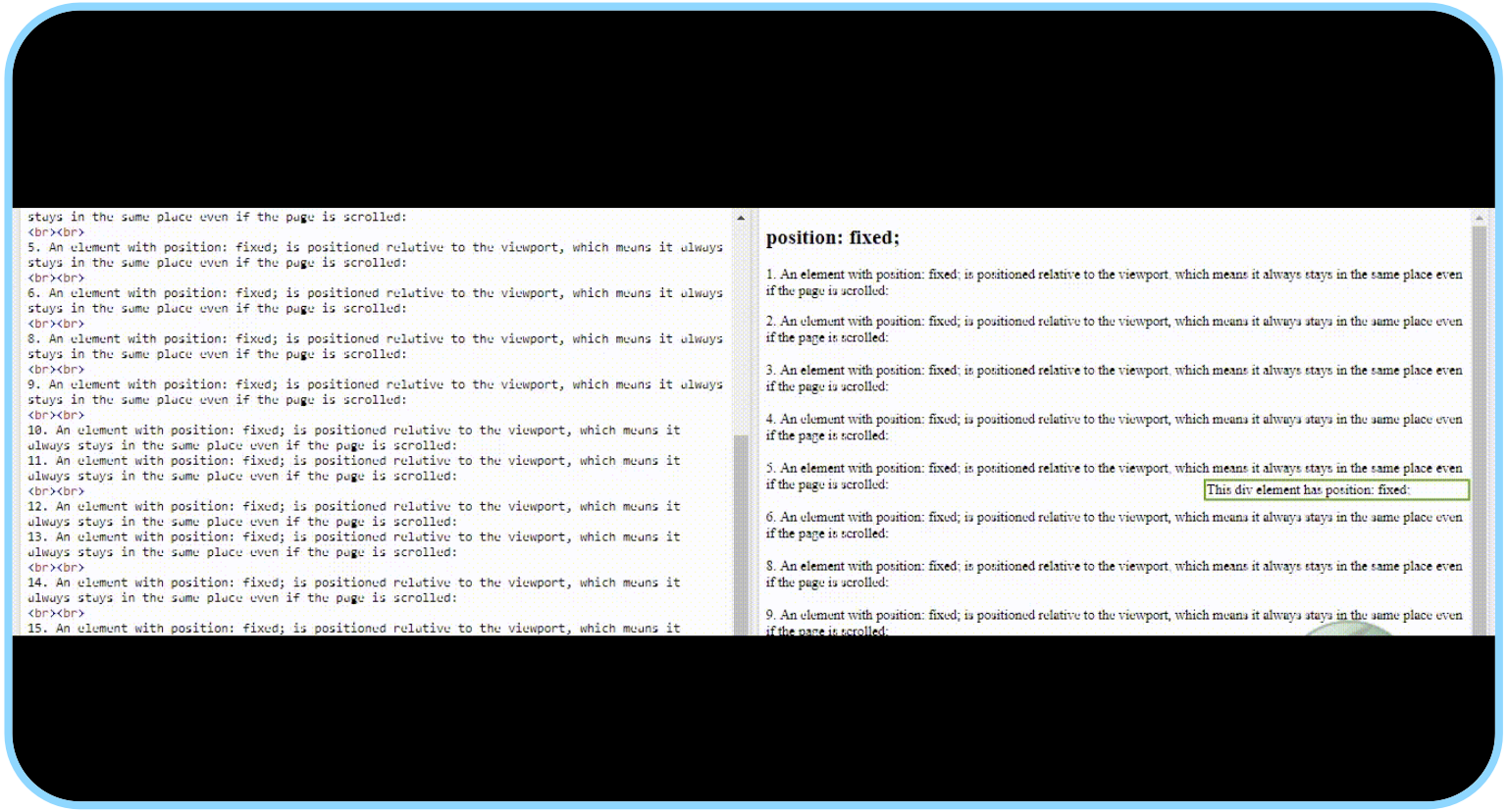
This div element has position: relative;

Fixed positioning

```
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>
```

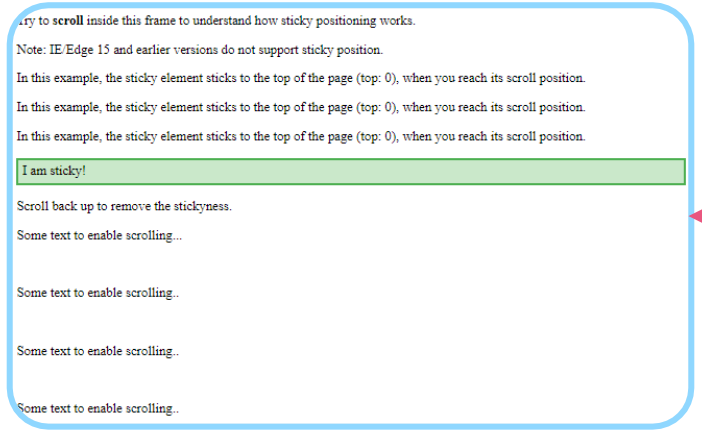


Fixed positioning

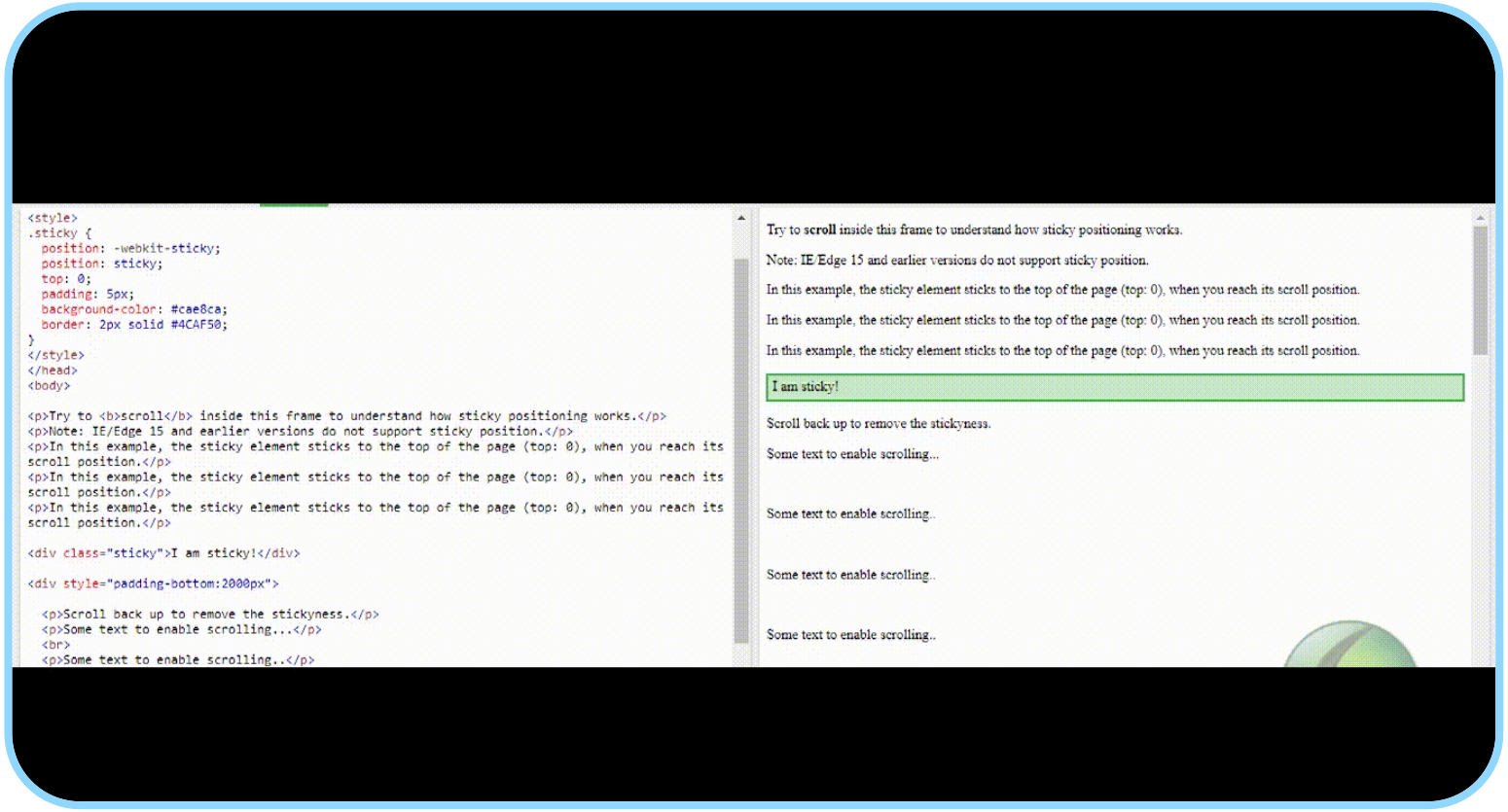


Sticky positioning

```
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>
```

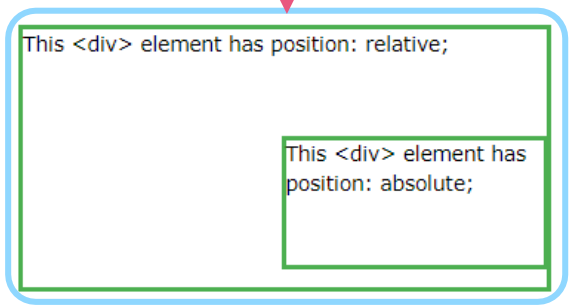


Sticky positioning



Absolute positioning

```
<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>
```



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

CSS float

```
<!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>
```

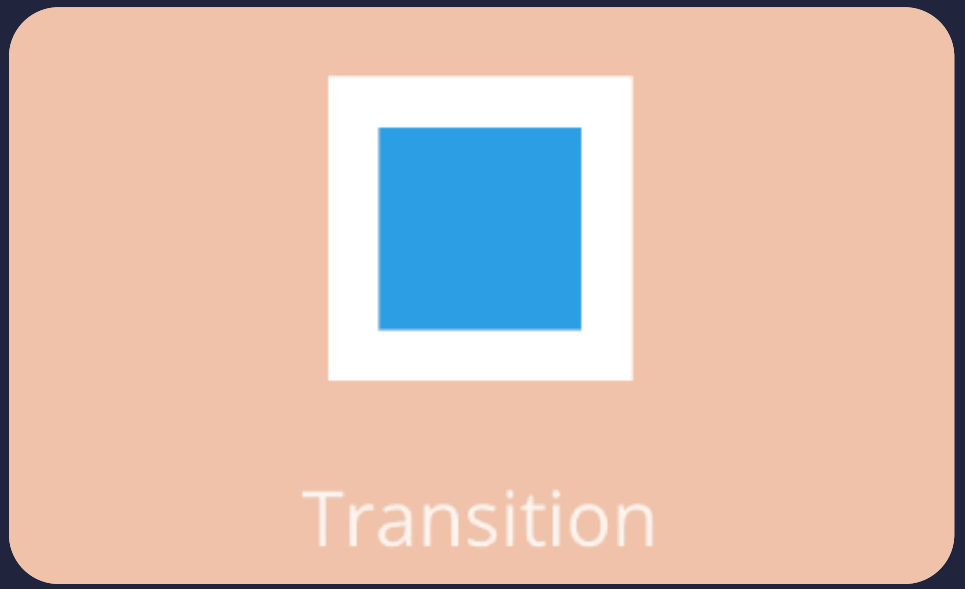
In this example, the image will float to the left in the paragraph, and the text in the paragraph will wrap around the image.

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

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 or not 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.s

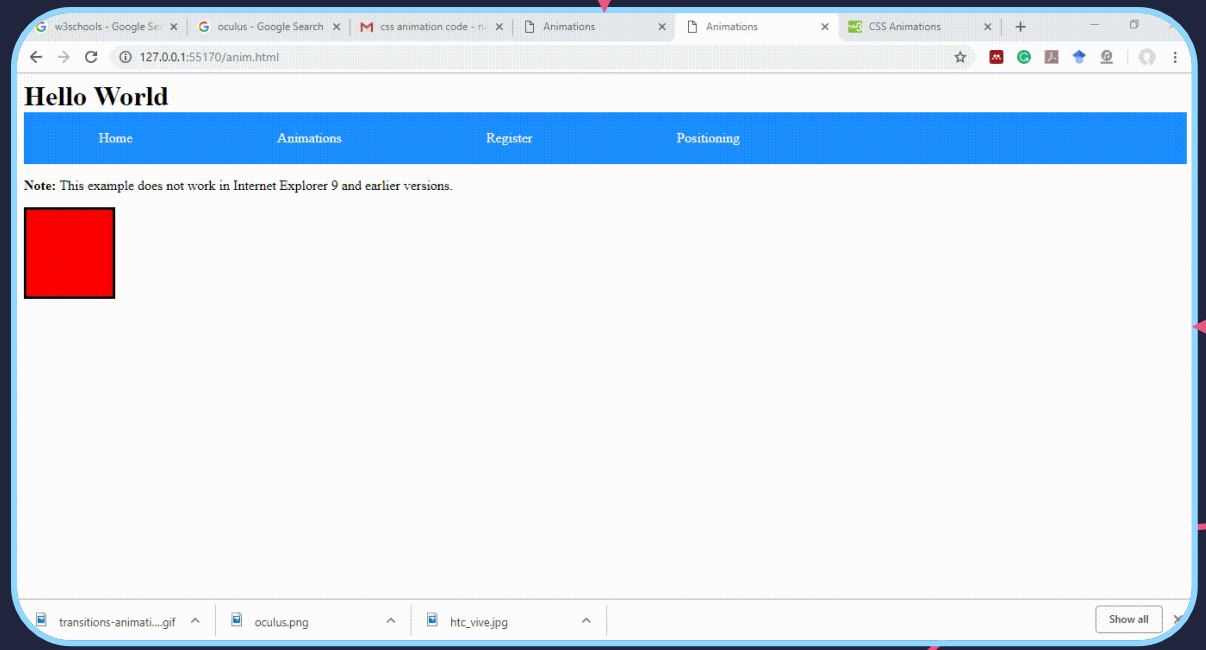
Animating colours and borders

HTML

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

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;}
}
```



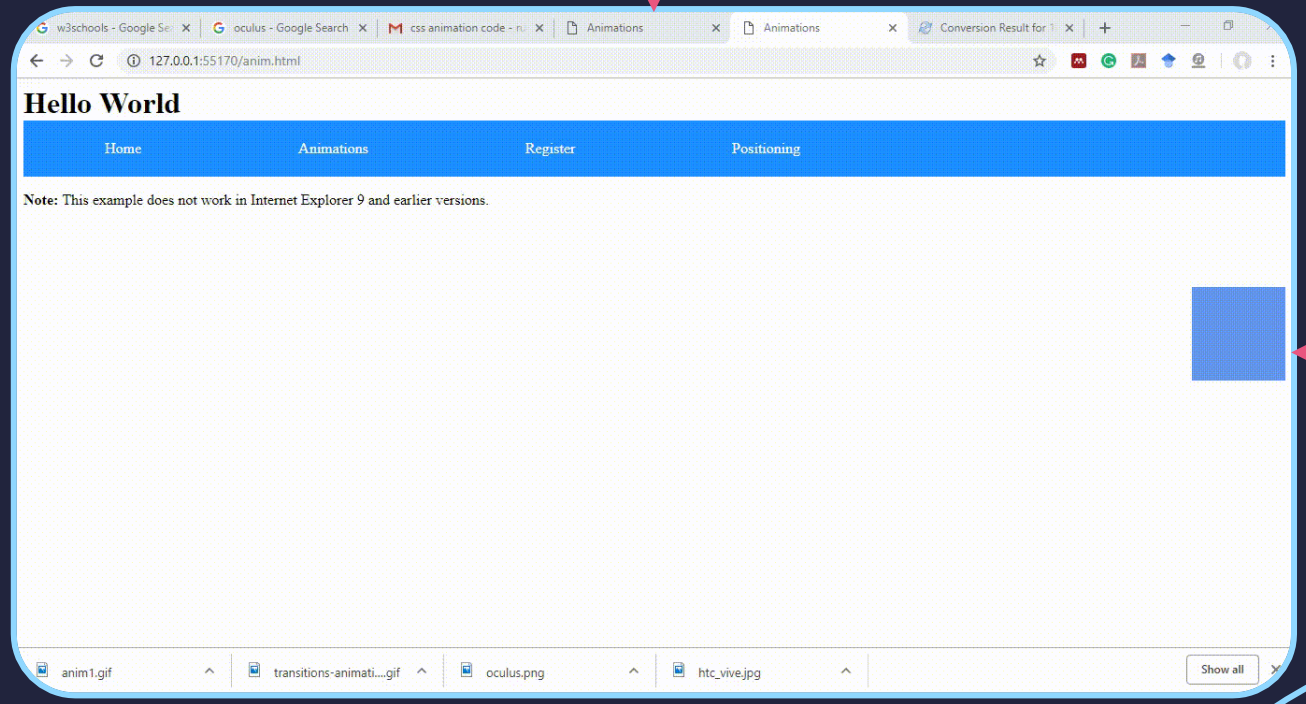
Animating size

HTML

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

CSS

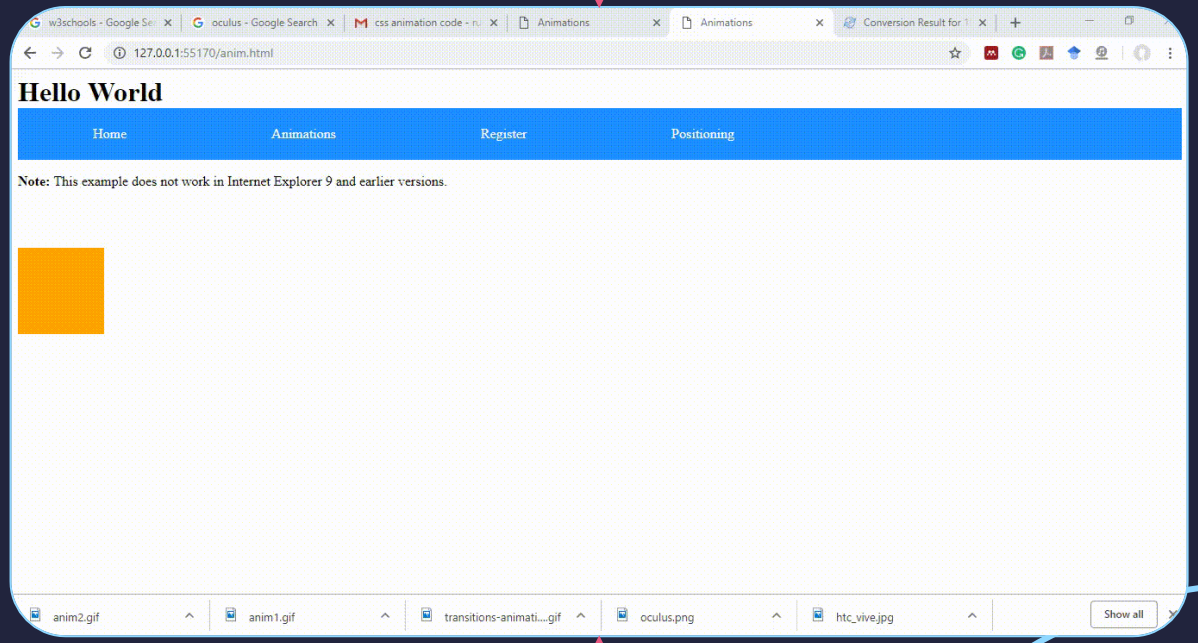
```
#square2 {
width: 100px;
height: 100px;
margin-top: 5%;
float:right;
background-color: cornflowerblue;
animation-name: square-anim2;
animation-duration: 5s;
animation-iteration-count: 3;
}

@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%;}

}
```