Styling using Pseudo class

Pseudo-classes are used to match content in a more extensive way since they can refer to elements not directly represented in the DOM. They refer to the state of the element and are used to style a special state of the element.

Focus Method:

he `focus()` method gives focus to an element (if it can be focused).he `focus()` method gives focus to an element (if it can be focused).

```
<!DOCTYPE html>
<html>
<style>
a:focus, a:active {
  color: red;
}
</style>

<body>
<h1>The Element Object</h1>
<h2>The focus() and blur() Methods</h2>

<a id="myAnchor" href="https://soa.ac.in">SOA</a>

<p>Click the buttons to give or remove focus from the link above.</p>

<input type="button" onclick="getfocus()" value="Get focus">
<input type="button" onclick="losefocus()" value="Lose focus">

<script>
function getfocus() {
  document.getElementById("myAnchor").focus();
}

function losefocus() {
  document.getElementById("myAnchor").blur();
}
</script>
```

```
</body>
</html>
```

Anchor Pseudo-classes

`a:hover` MUST come after `a:link` and `a:visited in` the CSS definition in order to be effective! `a:active` MUST come after `a:hover` in the CSS definition in order to be effective! Pseudo-class names are not case-sensitive.

```
<!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>
```

**Hover on **

Hover on

CSS - The :first-child Pseudo-class

The `:first-child` pseudo-class matches a specified element that is the first child of another element.

Match the first <p> element

In the following example, the selector matches any <p> element that is the first child of any element:

```
<!DOCTYPE html>
<html>
<head>
<style>
p:first-child {
  color: blue;
}
</style>
</head>
<body>

<p>This is some text.</p>
<p>This is some text.</p>

<div>
  <p>This is some text.</p>
  <p>This is some text.</p>
</div>
```

```
</body>
</html>
```

Match all <i> elements in all first child <p> elements

In the following example, the selector matches all <i> elements in <p> elements that are the first child of another element:

```
<!DOCTYPE html>
<html>
<head>
<style>
p:first-child i {
  color: blue;
}
</style>
</head>
<body>

<p>I am a <i>strong</i> person. I am a <i>strong</i> person.</p>
<p>I am a <i>strong</i> person. I am a <i>strong</i> person.</p>

<div>
  <p>I am a <i>strong</i> person. I am a <i>strong</i> person.</p>
  <p>I am a <i>strong</i> person. I am a <i>strong</i> person.</p>
</div>

</body>
</html>
```

CSS - The :lang Pseudo-class

The `:lang` pseudo-class allows you to define special rules for different languages.

In the example below, `:lang` defines the quotation marks for <q> elements with lang="no":

```
<!DOCTYPE html>
<html>
<head>
<style>
q:lang(no) {
  quotes: "~" "~";
}
</style>
</head>
<body>

<p>Some text <q lang="no">A quote in a paragraph</q> Some text.</p>
<p>In this example, :lang defines the quotation marks for q elements with lang="no":</p>

</body>
</html>
```

The `:active` selector is used to select and style the active link.

A link becomes active when you click on it.

Tip: The :active selector can be used on all elements, not only links.

Tip: Use the :link selector to style links to unvisited pages, the :visited selector to style links to visited pages, and the :hover selector to style links when you mouse over them.

```
<!DOCTYPE html>
<html>
<head>
<style>
a:active {
  background-color: yellow;
}
</style>
</head>
<body>

<h1>Demo of the :active selector</h1>

<a href="https://www.w3schools.com">W3schools.com</a><br>
<a href="https://www.w3schools.com/html/">HTML</a><br>
<a href="https://www.w3schools.com/css/">CSS</a>

<p><b>Note:</b> The :active selector styles the active link.</p>

</body>
</html>
```

The `:empty` selector matches every element that has no children (including text nodes).

```
<!DOCTYPE html>
<html>
<head>
<style>
p:empty {
```

```
  width: 100px;
  height: 20px;
  background: #ff0000;
}
</style>
</head>
<body>


<p></p>
<p>A paragraph.</p>
<p>Another paragraph.</p>


</body>
</html>
```

he `:valid` selector selects form elements with a value that validates according to the element's settings.

Note: The :valid selector only works for form elements with limitations, such as input elements with min and max attributes, email fields with a legal email, or number fields with a numeric value, etc.

Tip: Use the :invalid selector to select form elements with a value that does not validate according to the element's settings.

```
<!DOCTYPE html>
<html>
<head>
<style>
input:valid {
  background-color: purple;
}
</style>
```

</head>

<body>

<h1>Demo of the :valid selector</h1>

<input type="email" value="support@example.com">

** Try typing an illegal e-mail address, to see the styling disappear. **

</body>

</html>

he `:checked` selector matches every checked <input> element (only for radio buttons and checkboxes) and <option> element.

```
<!DOCTYPE html>
<html>
<head>
<style>
input:checked {
  height: 50px;
  width: 50px;
}
</style>
</head>
<body>
```

```
<form action="">
  <input type="radio" checked="checked" value="male" name="gender"> Male<br>
  <input type="radio" value="female" name="gender"> Female<br>
  <input type="checkbox" checked="checked" value="Bike"> I have a bike<br>
  <input type="checkbox" value="Car"> I have a car
</form>



</body>
</html>
```

The `:link` selector is used to select unvisited links.

Note: The :link selector does not style links you have already visited.

Tip: Use the :visited selector to style links to visited pages, the :hover selector to style links when you mouse over them, and the :active selector to style links when you click on them.

```
<!DOCTYPE html>
<html>
<head>
<style>
a:link {
  background-color: yellow;
}
</style>
</head>
<body>

<h1>Demo of the :link selector</h1>

<p>The :link selector style links to pages you have not visited yet:</p>
```

```
<a href="https://www.w3schools.com">W3Sschools</a>
<a href="http://www.wikipedia.org">Wikipedia</a>

</body>
</html>
```

The `:in-range` selector selects all elements with a value that is within a specified range.

```
<!DOCTYPE html>
<html>
<head>
<style>
input:in-range {
  border: 2px solid yellow;
}
</style>
</head>
<body>

<h1>Demo of the :in-range selector</h1>

<input type="number" min="5" max="10" value="7">

<p>Try typing a number out of range (less than 5 or higher than 10), to see the styling disappear.</p>

</body>
</html>
```

The `:not(selector)` selector matches every element that is NOT the specified element/selector.

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
  color: #000000;
}

:not(p) {
  color: #ff0000;
}
</style>
</head>
<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>

<div>This is some text in a div element.</div>

<a href="https://www.w3schools.com" target="_blank">Link to W3Schools!</a>

</body>
</html>
```

The `:first-child` selector is used to select the specified selector, only if it is the first child of its parent.

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

```
<head>
<style>
p:first-of-type {
  background: red;
}
</style>
</head>
<body>

<p>The first paragraph.</p>
<p>The second paragraph.</p>
<p>The third paragraph.</p>
<p>The fourth paragraph.</p>

</body>
</html>
```

The `:first-of-type` selector matches every element that is the first child, of a particular type, of its parent.

```
<!DOCTYPE html>
<html>
<head>
<style>
p:first-of-type {
  background: red;
}
</style>
```

```
</head>
<body>

<p>The first paragraph.</p>
<p>The second paragraph.</p>
<p>The third paragraph.</p>
<p>The fourth paragraph.</p>

</body>
</html>
```

The `:lang()` selector is used to select elements with a lang attribute with the specified value.

```
<!DOCTYPE html>
<html>
<head>
<style>
p:lang(it) {
  background: yellow;
}
</style>
</head>
<body>

<h1>Demo of the :lang selector</h1>

<p>I live in Italy.</p>
<p lang="it">Ciao bella!</p>

</body>
</html>
```

The `:nth-child(n)` selector matches every element that is the *n*th child of its parent.

n can be a number, a keyword (odd or even), or a formula (like *an* + *b*).

Tip: Look at the :nth-of-type() selector to select the element that is the *n*th child, **of the same type (tag name)**, of its parent.

```
<!DOCTYPE html>
<html>
<head>
<style>
/* Selects the second element of div siblings */
div:nth-child(2) {
  background: red;
}

/* Selects the second li element in a list */
li:nth-child(2) {
  background: lightgreen;
}

/* Selects every third element among any group of siblings */
:nth-child(3) {
  background: yellow;
}
</style>
</head>
<body>

<div>
  <p>This is some text.</p>
</div>
```

```
<div>
  <p>This is some text.</p>
</div>

<div>
  <p>This is some text.</p>
</div>

<ul>
  <li>First list item</li>
  <li>Second list item</li>
  <li>Third list item</li>
  <li>Fourth list item</li>
  <li>Fifth list item</li>
</ul>

</body>
</html>
```

CSS :nth-last-child() Selector

```
<!DOCTYPE html>
<html>
<head>
<style>
p:nth-last-child(2) {
  background: red;
}
</style>
</head>
<body>
```

```
<p>The first paragraph.</p>

<p>The second paragraph.</p>

<p>The third paragraph.</p>

<p>The fourth paragraph.</p>


</body>

</html>
```

The `:nth-last-of-type(n)` selector matches every element that is the *n*th child, of a particular type, of its parent, counting from the last child.

n can be a number, a keyword, or a formula.

Tip: Look at the :nth-last-child() selector to select the element that is the *n*th child, **regardless of type**, of its parent, counting from the last child.

```
<!DOCTYPE html>

<html>

<head>

<style>

p:nth-last-of-type(2) {

  background: red;

}

</style>

</head>

<body>


<p>The first paragraph.</p>

<p>The second paragraph.</p>

<p>The third paragraph.</p>

<p>The fourth paragraph.</p>


</body>
```

</html>

The `:nth-of-type(n)` selector matches every element that is the *n*th child, of the same type (tag name), of its parent.

n can be a number, a keyword (odd or even), or a formula (like *an* + *b*).

Tip: Look at the :nth-child() selector to select the element that is the *n*th child, **regardless of type**, of its parent.

Version: CSS3

```
<!DOCTYPE html>
<html>
<head>
<style>
/* Selects the second element of div siblings */
div:nth-of-type(2) {
  background: red;
}

/* Selects the second li element in a list */
li:nth-of-type(2) {
  background: lightgreen;
}

/* Selects every third element among any group of siblings */
:nth-of-type(3) {
  background: yellow;
}
</style>
</head>
<body>

<div>
```

```
<p>This is some text.</p>
</div>

<div>
<p>This is some text.</p>
</div>

<div>
<p>This is some text.</p>
</div>

<ul>
  <li>First list item</li>
  <li>Second list item</li>
  <li>Third list item</li>
  <li>Fourth list item</li>
  <li>Fifth list item</li>
</ul>

</body>
</html>
```

The `:only-child` selector matches every element that is the only child of its parent.

```
<!DOCTYPE html>
<html>
<head>
<style>
p:only-child {
  background: red;
}
```

```
</style>
</head>
<body>

<div><p>This is a paragraph.</p></div>

<div><span>This is a span.</span><p>This is a paragraph.</p></div>

</body>
</html>
```

The `:only-of-type` selector matches every element that is the only child of its type, of its parent.

```
<!DOCTYPE html>
<html>
<head>
<style>
p:only-of-type {
  background: red;
}
</style>
</head>
<body>

<div><p>This is a paragraph.</p></div>

<div><p>This is a paragraph.</p><p>This is a paragraph.</p></div>

</body>
</html>
```

The `:nth-of-type(n)` selector matches every element that is the *n*th child, of the same type (tag name), of its parent.

n can be a number, a keyword (odd or even), or a formula (like *an* + *b*).

Tip: Look at the :nth-child() selector to select the element that is the *n*th child, **regardless of type**, of its parent.

```
<!DOCTYPE html>
<html>
<head>
<style>
/* Selects the second element of div siblings */
div:nth-of-type(2) {
  background: red;
}

/* Selects the second li element in a list */
li:nth-of-type(2) {
  background: lightgreen;
}

/* Selects every third element among any group of siblings */
:nth-of-type(3) {
  background: yellow;
}
</style>
</head>
<body>

<div>
<p>This is some text.</p>
</div>
```

```
<div>
<p>This is some text.</p>
</div>

<div>
<p>This is some text.</p>
</div>

<ul>
  <li>First list item</li>
  <li>Second list item</li>
  <li>Third list item</li>
  <li>Fourth list item</li>
  <li>Fifth list item</li>
</ul>

</body>
</html>
```

The `:nth-of-type(n)` selector matches every element that is the *n*th child, of the same type (tag name), of its parent.

n can be a number, a keyword (odd or even), or a formula (like *an* + *b*).

Tip: Look at the :nth-child() selector to select the element that is the *n*th child, **regardless of type**, of its parent.

```
<!DOCTYPE html>
<html>
<head>
<style>
/* Selects the second element of div siblings */
div:nth-of-type(2) {
```

```
  background: red;
}

/* Selects the second li element in a list */
li:nth-of-type(2) {
  background: lightgreen;
}

/* Selects every third element among any group of siblings */
:nth-of-type(3) {
  background: yellow;
}
</style>
</head>
<body>

<div>
<p>This is some text.</p>
</div>

<div>
<p>This is some text.</p>
</div>

<div>
<p>This is some text.</p>
</div>

<ul>
```

```
  <li>First list item</li>
  <li>Second list item</li>
  <li>Third list item</li>
  <li>Fourth list item</li>
  <li>Fifth list item</li>
</ul>


</body>
</html>
```

Styling with Pseudo-elements

A CSS pseudo-element is used to style specified parts of an element. Pseudo-elements can be used to refer to content that does not exist in the source code, but the content is generated afterward.

For example, first-line can be used to change the font of the first line of a paragraph. Check the following example:

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

As a rule, double colons (::) should be used instead of a single colon (:). This distinguishes pseudo-classes from pseudo-elements.

CSS inheritance

HTML elements can inherit CSS styles from their parent elements. This is called CSS inheritance. Not all styles are inherited from a parent or ancestor element. Generally, styles that apply to text are inherited, whereas borders, margins and paddings and similar styles are not inherited. HTML elements can inherit styles from more remote ancestors too, and not just from their direct parents.

As in this example:

This text inherits the font-size of the parent div element.

This text inherits the font-size of the parent div element.

In this example the CSS property font-family is set on the body element. This CSS property is then inherited by the div and p elements. Now that we know the different ways we can define CSS rule sets, we will learn about another important aspect of how the rules get applied - Specificity.

CSS precedence and specificity

- When multiple CSS rules are applied on the same HTML elements, and those CSS rules set some of the same CSS properties, finally, which styles end up being applied to the HTML element, is determined by CSS precedence. In the following example, the property font-size is being applied to multiple elements:

- Let's understand how the precedence gets applied here. CSS precedence rules When the browser needs to resolve what styles to apply to a given HTML element, it uses a set of CSS precedence rules. Given these rules, the browser can determine what styles to apply.
- The rules are: Use of !important after CSS properties Specificity of CSS rule selectors Sequence of declaration Note that CSS precedence happens at CSS property level. Thus, if two CSS rules target the same HTML element, and the first CSS rule takes precedence over the second, then all CSS properties specified in the first CSS rule takes precedence over the CSS properties declared in the second rule.

- However, if the second CSS rule contains CSS properties that are not specified in the first CSS rule, then these are still applied. The CSS rules are combined—not overriding each other. Use of !important after CSS properties.

- The !important instruction has the highest precedence of all precedence factors, which should be used carefully: If you need a certain CSS property to take precedence over all other CSS rules setting the same CSS property for the same HTML elements, you can add the instruction !important after the CSS property when you declare it.
- It is not a good practice to use !important as it overrides the normal CSS rules flow and specificity. Try to avoid the use of !important. Try to make better use of CSS cascading properties and increasing specificity by using more specific rules:

- The specificity of a CSS rule depends on its selector.
- If the CSS selector is more specific, the CSS property declarations will get greater precedence inside the CSS rule for the selector. The more specifically or uniquely, a CSS selector points to an HTML element, the higher will be its specificity.

- The specificity is determined by the type of CSS selector. Here is a list of CSS selector specificity: Inherited styles: Lowest specificity of all selectors, since inherited style targets the element's parent, and not the HTML element itself Universal selector *: Lowest specificity of all directly targeted selectors Element selector: Higher specificity than universal selector and inherited styles Attribute selector: Higher specificity than element selector Class selector: Higher specificity than an attribute, element, and universal selectors ID selector: Higher specificity than a class selector Inline style using style attribute: Stronger specificity than ID selector Universal selector (*), combinators (+, >, ~, ' '), and negation pseudoclass (:not) have no effect on specificity. The following code shows how t

How to calculate specificity?

- A selector’s specificity is calculated as follows: Count the number of ID attributes in the selector (= a) Count the number of other attributes and pseudo-classes in the selector (= b) Count the number of element or pseudo-element names in the selector (= c) On concatenating the above three numbers, a-b-c gives the specificity for the selector.
- The selector with the highest specificity gets applied.
- CSS units Some of the CSS properties need units of measurement, so in order to be able to use and apply the values correctly, we should know the units which can be used.
- The main CSS units can be categorized as follows. Absolute lengths Absolute values are applied without any relation to user settings. For example, px, cm, mm, inches, and so on If browser default settings are changed, it does not impact the absolute px size of fonts px is most commonly used for specifying the absolute length in web development

Styling text:

CSS Colors

CSS Color property is used to set the color of HTML elements. This property is used to set font color, background color, etc. The color of an element can be defined in the following ways:

- Built-In Color
- RGB Format
- RGBA Format
- Hexadecimal Notation
- HSL
- HSLA

Built-In Color: These are a set of predefined colors which are used by its name. For example: red, blue, green etc.

Syntax:

```
h1 {
    color: color-name;
}
```

CSS text-align-last Property

- The text-align-last property in CSS is used to set the last line of the paragraph just before the line break. The line break may be due to the natural ending of a paragraph, or it may be due to the use of
 tag. The text-align-last property sets the alignment of all the last lines occurring in the element in which the text-align-last property is applied. For example, if the text-align-last property is applied to the <p> element then, all the last lines of the <p> element will be affected by the text-align-last property.

  Syntax:

```
text-align-last: auto|start|end|left|right|center|justify|
    initial|inherit;
```

Default Value: Its default value is auto.

```
<!DOCTYPE html>
<html>

<head>
     <title>
          text-align-last property
     </title>

     <!-- CSS style to text-align-last property -->
     <style>
          p {
               text-align-last: right;
               font-family: sans-serif;
               border: 1px solid black;
          }
     </style>
</head>

<body>

     <h2 style="text-align:center">
          text-align-last: right;
     </h2>

     <!-- text-align-last: left; property -->
     <p>
          Prepare for the Recruitment drive of product
          based companies like Microsoft, Amazon, Adobe
          etc with a free online placement preparation
          course. The course focuses on various MCQ's
          & Coding question likely to be asked in the
          interviews & make your upcoming placement
          season efficient and successful.
     </p>

     <!-- text-align-last: right; property -->
     <p> GeeksForGeeks: A computer science portal</p>

</body>

</html>
```

textdecration

<!DOCTYPE html>
<html>
<body>

```
<p id="demo">This is an example paragraph.</p>

<button type="button" onclick="myFunction()">Decorate text</button>

<script>
function myFunction() {
  document.getElementById("demo").style.textDecoration = "underline overline";
}
</script>

</body>
</html>
```

The `text-indent` property specifies the indentation of the first line in a text-block.

Note: Negative values are allowed. The first line will be indented to the left if the value is negative.

```
<!DOCTYPE html>

<html>

<head>

<style>

div.a {

  text-indent: 50px;

}


div.b {

  text-indent: -2em;

}


div.c {

  text-indent: 30%;

}

</style>

</head>
```

<body>

<h1>The text-indent Property</h1>

<h2>text-indent: 50px:</h2>

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut.

<h2>text-indent: -2em:</h2>

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut.

<h2>text-indent: 30%:</h2>

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut.

</body>

</html>

The `line-height` property specifies the height of a line.

Note: Negative values are not allowed.

<!DOCTYPE html>

<html>

```
<head>
<style>
div.a {
  line-height: normal;
}

div.b {
  line-height: 1.6;
}

div.c {
  line-height: 80%;
}

div.d {
  line-height: 200%;
}
</style>
</head>
<body>

<h1>The line-height Property</h1>

<h2>line-height: normal (default):</h2>
<div class="a">This is a paragraph with a standard line-height.<br>
The standard line height in most browsers is about 110% to 120%.</div>

<h2>line-height: 1.6 (recommended):</h2>
<div class="b">This is a paragraph with the recommended line-height.<br>
```

```
The line height is here set to 1.6. This is a unitless value;<br>
meaning that the line height will be relative to the font size.</div>

<h2>line-height: 80%:</h2>
<div class="c">This is a paragraph with a smaller line-height.<br>
The line height is here set to 80%.</div>

<h2>line-height: 200%:</h2>
<div class="d">This is a paragraph with a bigger line-height.<br>
The line height is here set to 200%.</div>

</body>
</html>
```

The `word-spacing` property increases or decreases the white space between words.

```
<!DOCTYPE html>
<html>
<head>
<style>
p.a {
  word-spacing: normal;
}

p.b {
  word-spacing: 30px;
}

p.c {
  word-spacing: 1cm;
}
```

```
</style>
</head>
<body>

<h1>The word-spacing Property</h1>

<h2>word-spacing: normal:</h2>
<p class="a">This is some text. This is some text.</p>

<h2>word-spacing: 30px:</h2>
<p class="b">This is some text. This is some text.</p>

<h2>word-spacing: 1cm:</h2>
<p class="c">This is some text. This is some text.</p>

</body>
</html>
```

The `letter-spacing` property increases or decreases the space between characters in a text.

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  letter-spacing: 3px;
}

h2 {
  letter-spacing: 2px;
}
```

```
h3 {
  letter-spacing: -1px;
}
</style>
</head>
<body>

<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>

</body>
</html>
```

The direction property sets or returns the text direction (reading order) of an element's content.

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

<p id="demo">This is an example paragraph.</p>

<button type="button" onclick="myFunction()">Let text flow from right to left</button>

<script>
function myFunction() {
  document.getElementById("demo").style.direction = "rtl";
}
</script>
```

</body>

</html>

The background property is a shorthand property for:

- background-color
- background-image
- background-position
- background-size
- background-repeat
- background-origin
- background-clip
- background-attachment

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background: purple url("img_tree.gif") no-repeat fixed center;
}
</style>
</head>
<body>

<h1>The background Property</h1>

<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
```

```
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>
<p>This is some text</p>

</body>
</html>
```

background-color Specifies the background color to be used

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-color: coral;
```

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

<h1>The background-color Property</h1>

<p>
To add a background color you can use color names or values. Values can be of type HEX, RGB, RGBA, HSL, or HSLA.
</p>

</body>
</html>
```

background-origin Specifies the positioning area of the background images

```
<!DOCTYPE html>
<html>
<head>
<style>
#myDIV {
  height:200px;
  border: 10px dashed black;
  padding: 25px;
  background-origin: border-box;
  background-image: url("img_tree.gif");
}
</style>
</head>
<body>
```

```
<h1>The background-origin Property</h1>

<div id="myDIV">
This demo shows how to specify where the background image should be positioned.
</div>

</body>
</html>
```

background-position Specifies the position of the background images

```
<!DOCTYPE html>
<html>
<head>
<style>
#myDIV {
  height:200px;
  border: 10px dashed black;
  padding: 25px;
  background-attachment: fixed;
  background-position: center;
  background-image: url("img_tree.gif");
}
</style>
</head>
<body>

<h1>The background-position Property</h1>

<div id="myDIV">
```

This demo shows how to specify where the background image should be positioned.

</body>

</html>

background-repeat Specifies how to repeat the background images

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-repeat: repeat-y;
  background-image: url("img_flwr.gif");
}
</style>
</head>
<body>

<h1>The background-repeat Property</h1>

<p>
This demo shows how to specify how the background image can be repeated.
</p>

</body>
</html>
```

background-image Specifies ONE or MORE background images to be used

```
<!DOCTYPE html>
```

```
<html>
<head>
<style>
body {
  background-image: url("paper.gif");
}
</style>
</head>
<body>

<h1>The background-image Property</h1>

<p>
This demo shows some different techiques on how to set a background image of a HTML element.
</p>

</body>
</html>
```

background-clip Specifies the painting area of the background images

```
<!DOCTYPE html>
<html>
<head>
<style>
#myDIV {
  border: 10px dotted black;
  padding: 15px;
  background: lightblue;
  background-clip: border-box;
```

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

<h1>The background-clip Property</h1>

<div id="myDIV">
  <p>
    <em>Content goes here.</em>
  </p>
</div>

</body>
</html>
```

CSS Flexbox Layout Module

Before the Flexbox Layout module, there were four layout modes:

- Block, for sections in a webpage
- Inline, for text
- Table, for two-dimensional table data
- Positioned, for explicit position of an element

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  flex-wrap: nowrap;
  background-color: DodgerBlue;
```

```
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>
<h1>Flexible Boxes</h1>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
</div>

<p>Try to resize the browser window.</p>
<p>A container with "flex-wrap: nowrap;" will never wrap its items.</p>
<p><strong>Note:</strong> Flexbox is not supported in Internet Explorer 10 or earlier versions.</p>
```

```
</body>
</html>
```

The flex container becomes flexible by setting the `display` property to `flex`:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  margin: 10px;
  padding: 20px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>Create a Flex Container</h1>

<div class="flex-container">
  <div>1</div>
```

```
  <div>2</div>
  <div>3</div>
</div>

<p>A Flexible Layout must have a parent element with the <em>display</em> property set to <em>flex</em>.</p>

<p>Direct child elements(s) of the flexible container automatically becomes flexible items.</p>

</body>
</html>
```

The flex-direction Property

The `flex-direction` property defines in which direction the container wants to stack the flex items.

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  flex-direction: column;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
```

```
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The flex-direction Property</h1>

<p>The "flex-direction: column;" stacks the flex items vertically (from top to bottom):</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>
```

column-reverse

The `column-reverse` value stacks the flex items vertically (but from bottom to top):

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
```

```
  display: flex;
  flex-direction: column-reverse;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The flex-direction Property</h1>

<p>The "flex-direction: column-reverse;" stacks the flex items vertically (but from bottom to top):</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>
```

The `row` value stacks the flex items horizontally (from left to right):

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  flex-direction: row;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The flex-direction Property</h1>

<p>The "flex-direction: row;" stacks the flex items horizontally (from left to right):</p>

<div class="flex-container">
  <div>1</div>
```

```
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>
```

The `row-reverse` value stacks the flex items horizontally (but from right to left):

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  flex-direction: row-reverse;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>
```

```
<h1>The flex-direction Property</h1>

<p>The "flex-direction: row-reverse;" stacks the flex items horizontally (but from right to left):</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>
```

The flex-wrap Property

The `flex-wrap` property specifies whether the flex items should wrap or not.

The examples below have 12 flex items, to better demonstrate the `flex-wrap` property.

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  flex-wrap: wrap;
  background-color: DodgerBlue;
}
```

```
.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The flex-wrap Property</h1>

<p>The "flex-wrap: wrap;" specifies that the flex items will wrap if necessary:</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
```

```
</div>

<p>Try resizing the browser window.</p>

</body>
</html>
```

The `nowrap` value specifies that the flex items will not wrap (this is default):

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  flex-wrap: nowrap;
  background-color: DodgerBlue;
}

.flex-container>div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>
```

```
<h1>The flex-wrap Property</h1>

<p>The "flex-wrap: nowrap;" specifies that the flex items will not wrap (this is default):</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
</div>

<p>Try resizing the browser window.</p>

</body>
</html>
```

The `wrap-reverse` value specifies that the flexible items will wrap if necessary, in reverse order:

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

```
<head>
<style>
.flex-container {
  display: flex;
  flex-wrap: wrap-reverse;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The flex-wrap Property</h1>

<p>The "flex-wrap: wrap-reverse;" specifies that the flex items will wrap if necessary, in reverse order:</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
```

```
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
</div>

<p>Try resizing the browser window.</p>

</body>
</html>
```

The flex-flow Property

The `flex-flow` property is a shorthand property for setting both the `flex-direction` and `flex-wrap` properties.

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  flex-flow: row wrap;
  background-color: DodgerBlue;
}
```

```
.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>
<h1>The flex-flow Property</h1>

<p>The flex-flow property is a shorthand property for the flex-direction and the flex-wrap properties.</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
</div>
```

```
<p>Try resizing the browser window.</p>

</body>
</html>
```

The justify-content Property

The `justify-content` property is used to align the flex items:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  justify-content: center;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>
```

```
<h1>The justify-content Property</h1>

<p>The "justify-content: center;" aligns the flex items at the center of the container:</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>
```

The `center` value aligns the flex items at the center of the container:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  justify-content: center;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
```

```
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The justify-content Property</h1>

<p>The "justify-content: center;" aligns the flex items at the center of the container:</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>
```

The `flex-start` value aligns the flex items at the beginning of the container (this is default):

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
```

```
  justify-content: flex-start;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The justify-content Property</h1>

<p>The "justify-content: flex-start;" aligns the flex items at the beginning of the container (this is default):</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>
```

The `flex-end` value aligns the flex items at the end of the container:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  justify-content: flex-end;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The justify-content Property</h1>

<p>The "justify-content: flex-end;" aligns the flex items at the end of the container:</p>

<div class="flex-container">
  <div>1</div>
```

```
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>
```

The `space-around` value displays the flex items with space before, between, and after the lines:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  justify-content: space-around;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>
```

```
<h1>The justify-content Property</h1>

<p>The "justify-content: space-around;" displays the flex items with space before, between, and after the lines:</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>
```

The `space-between` value displays the flex items with space between the lines:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  justify-content: space-between;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
```

```
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The justify-content Property</h1>

<p>The "justify-content: space-between;" displays the flex items with space between the lines:</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>
```

The align-items Property

The `align-items` property is used to align the flex items.

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
```

```
  display: flex;
  height: 200px;
  align-items: center;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The align-items Property</h1>

<p>The "align-items: center;" aligns the flex items in the middle of the container:</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>
```

The `flex-end` value displays the flex lines at the end of the container:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  height: 600px;
  flex-wrap: wrap;
  align-content: flex-end;
  overflow: scroll;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The align-content Property</h1>
```

```
<p>The "align-content: flex-end;" displays the flex lines at the end of the container:</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
</div>

</body>
</html>
```

The `flex-start` value displays the flex lines at the start of the container:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  height: 600px;
  flex-wrap: wrap;
```

```
  align-content: flex-start;
  overflow: scroll;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The align-content Property</h1>

<p>The "align-content: flex-start;" displays the flex lines at the start of the container:</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
```

```
  <div>8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
</div>

</body>
</html>
```

The `center` value displays the flex lines in the middle of the container:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  height: 600px;
  flex-wrap: wrap;
  align-content: center;
  overflow: scroll;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
```

```
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The align-content Property</h1>

<p>The "align-content: center;" displays the flex lines in the middle of the container:</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
</div>

</body>
</html>
```

The `stretch` value stretches the flex lines to take up the remaining space (this is default):

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  height: 600px;
  flex-wrap: wrap;
  align-content: stretch;
  overflow: scroll;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The align-content Property</h1>
```

The "align-content: stretch;" stretches the flex lines to take up the remaining space (this is default):

```
<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
</div>

</body>
</html>
```

The `space-around` value displays the flex lines with space before, between, and after them:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  height: 600px;
```

```
  flex-wrap: wrap;
  align-content: space-around;
  overflow: scroll;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The align-content Property</h1>

<p>The "align-content: space-around;" displays the flex lines with space before, between, and after them:</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
```

```
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
</div>

</body>
</html>
```

The `space-between` value displays the flex lines with equal space between them:

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  height: 600px;
  flex-wrap: wrap;
  align-content: space-between;
  overflow: scroll;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
```

```
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The align-content Property</h1>

<p>The "align-content: space-between;" displays the flex lines with equal space between them:</p>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
</div>

</body>
</html>
```

The `align-content` property is used to align the flex lines.

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  height: 600px;
  flex-wrap: wrap;
  align-content: space-between;
  overflow: scroll;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>The align-content Property</h1>
```

```
<p>The "align-content: space-between;" displays the flex lines with equal space between them:</p>


<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
</div>

</body>
</html>
```