Unit-2 CSS

CSS is the language we use to style a Web page.

What is CSS?

- CSS stands for Cascading Style Sheets
- CSS describes how HTML elements are to be displayed on screen, paper, or in other media
- CSS saves a lot of work. It can control the layout of multiple web pages all at once
- External stylesheets are stored in CSS files
- Cascading style sheets make it easy to apply and maintain styles on multiple topic files or Web pages. Once a stylesheet has been applied to a set of pages, changing a style on the stylesheet updates that style on each page the stylesheet is linked to.

CSS Basic Properties

Here are some basic CSS properties to work with.

- Text Properties
- List Properties
- Border Properties
- Font Properties

Text Properties

Property	Description	Values
color	Sets the color of a text	RGB, hex, keyword
line-height	Sets the distance between lines	normal, *number, length, %*
letter-spacing	Increase or decrease the space between characters	normal, *length*
text-align	Aligns the text in an element	left, right, center, justify
text-decoration	Adds decoration to text	none, underline, overline, line-through
text-indent	Indents the first line of text in an element	*length, %*
text-transform	Controls the letters in an element	none, capitalize, uppercase, lowercase

List Properties

Property	Description	Values
list-style	Sets all the properties for a list in one declaration	*list-style-type*, *list-style-position*, *list-style-image*, inherit
list-style-image	Specifies an image as the list-item marker	URL, none, inherit
list-style-position	Specifies where to place the list-item marker	inside, outside, inherit
list-style-type	Specifies the type of list-item marker	none, disc, circle, square, decimal, decimal-leading-zero, armenian, georgian, lower-alpha, upper-alpha, lower-greek, lower-latin, upper-latin, lower-roman, upper-roman, inherit

Border Properties

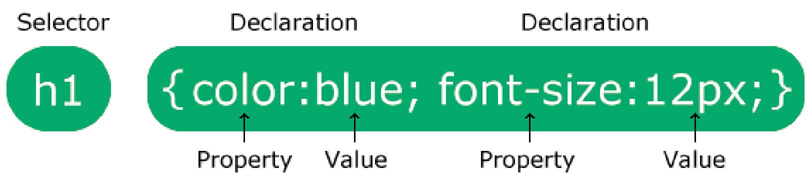
Property	Description	Values
border	Sets all the border properties in one declaration	*border-width*, *border-style*, *border-color*
border-bottom	Sets all the bottom border properties in one declaration	*border-bottom-width*, *border-bottom-style*, *border-bottom-color*
border-bottom-color	Sets the color of the bottom border	*border-color*
border-bottom-style	Sets the style of the bottom border	*border-style*
border-bottom-width	Sets the width of the bottom border	*border-width*
border-color	Sets the color of the four borders	*color_name*, *hex_number*, *rgb_number*, transparent, inherit

border-left	Sets all the left border properties in one declaration	*border-left-width, border-left-style, border-left-color*
border-left-color	Sets the color of the left border	*border-color*
border-left-style	Sets the style of the left border	*border-style*
border-left-width	Sets the width of the left border	*border-width*
border-right	Sets all the right border properties in one declaration	*border-right-width, border-right-style, border-right-color*
border-right-color	Sets the color of the right border	*border-color*
border-right-style	Sets the style of the right border	*border-style*
border-right-width	Sets the width of the right border	*border-width*
border-style	Sets the style of the four borders	none, hidden, dotted, dashed, solid, double, groove, ridge, inset, outset, inherit
border-top	Sets all the top border properties in one declaration	*border-top-width, border-top-style, border-top-color*
border-top-color	Sets the color of the top border	*border-color*
border-top-style	Sets the style of the top border	*border-style*
border-top-width	Sets the width of the top border	*border-width*
border-width	Sets the width of the four borders	thin, medium, thick, *length*, inherit

Font Properties

Property	Description	Values
font	Sets all the font properties in one declaration	*font-style*, *font-variant*, *font-weight*, *font-size/line-height*, *font-family*, caption, icon, menu, message-box, small-caption, status-bar, inherit
font-family	Specifies the font family for text	*family-name*, *generic-family*, inherit
font-size	Specifies the font size of text	xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger, *length*, *%*, inherit
font-style	Specifies the font style for text	normal, italic, oblique, inherit
font-variant	Specifies whether or not a text should be displayed in a small-caps font	normal, small-caps, inherit
font-weight	Specifies the weight of a font	normal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800, 900, inherit **Careful, many of these are not supported!**

CSS Syntax



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

Example

In this example all <p> elements will be center-aligned, with a red text color:

```
p {
  color: red;
  text-align: center;
}
```

Try it Yourself »

Example Explained

- p is a selector in CSS (it points to the HTML element you want to style: <p>).
- color is a property, and red is the property value
- text-align is a property, and center is the property value

CSS Selectors

A CSS selector selects the HTML element(s) you want to style.

CSS selectors are used to "find" (or select) the HTML elements you want to style.

We can divide CSS selectors into five categories:

- Simple selectors (select elements based on name, id, class)
- Combinator selectors (select elements based on a specific relationship between them)
- Pseudo-class selectors (select elements based on a certain state)
- Pseudo-elements selectors (select and style a part of an element)
- Attribute selectors (select elements based on an attribute or attribute value)

The CSS element Selector

The element selector selects HTML elements based on the element name.

Example

Here, all <p> elements on the page will be center-aligned, with a red text color:

```
p {
  text-align: center;
  color: red;
}
```

Try it Yourself »

The CSS id Selector

The id selector uses the id attribute of an HTML element to select a specific element.

The id of an element is unique within a page, so the id selector is used to select one unique element!

To select an element with a specific id, write a hash (#) character, followed by the id of the element.

Example

The CSS rule below will be applied to the HTML element with id="para1":

```
#para1 {
  text-align: center;
  color: red;
}
```

Try it Yourself »

Note: An id name cannot start with a number!

The CSS class Selector

The class selector selects HTML elements with a specific class attribute.

To select elements with a specific class, write a period (.) character, followed by the class name.

Example

In this example all HTML elements with class="center" will be red and center-aligned:

```
.center {
  text-align: center;
  color: red;
}
```

Try it Yourself »

You can also specify that only specific HTML elements should be affected by a class.

Example

In this example only <p> elements with class="center" will be red and center-aligned:

```
p.center {
  text-align: center;
  color: red;
}
```

Try it Yourself »

Example

In this example the <p> element will be styled according to class="center" and to class="large":

```
<p class="center large">This paragraph refers to two classes.</p>
```

Try it Yourself »

The CSS Universal Selector

The universal selector (*) selects all HTML elements on the page.

Example

The CSS rule below will affect every HTML element on the page:

```
* {
  text-align: center;
  color: blue;
}
```

Try it Yourself »

The CSS Grouping Selector

The grouping selector selects all the HTML elements with the same style definitions.

Look at the following CSS code (the h1, h2, and p elements have the same style definitions):

```
h1 {
  text-align: center;
  color: red;
}

h2 {
  text-align: center;
  color: red;
}

p {
  text-align: center;
  color: red;
}
```

It will be better to group the selectors, to minimize the code.

To group selectors, separate each selector with a comma.

Example

In this example we have grouped the selectors from the code above:

```
h1, h2, p {
  text-align: center;
  color: red;
}
```

Try it Yourself »

All CSS Simple Selectors

Selector	Example	Example description
#id	#firstname	Selects the element with id="firstname"
.class	.intro	Selects all elements with class="intro"
element.class	p.intro	Selects only <p> elements with class="intro"
*	*	Selects all elements
element	p	Selects all <p> elements
element,element,..	div, p	Selects all <div> elements and all <p> elements

How To Add CSS

When a browser reads a style sheet, it will format the HTML document according to the information in the style sheet.

Three Ways to Insert CSS

There are three ways of inserting a style sheet:

- External CSS
- Internal CSS
- Inline CSS

External CSS

With an external style sheet, you can change the look of an entire website by changing just one file!

Each HTML page must include a reference to the external style sheet file inside the <link> element, inside the head section.

Example

External styles are defined within the <link> element, inside the <head> section of an HTML page:

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="mystyle.css">
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

Try it Yourself »

An external style sheet can be written in any text editor, and must be saved with a .css extension.

The external .css file should not contain any HTML tags.

Here is how the "mystyle.css" file looks:

"mystyle.css"

```
body {
  background-color: lightblue;
}

h1 {
  color: navy;
  margin-left: 20px;
}
```

Note: Do not add a space between the property value and the unit:
Incorrect (space): `margin-left: 20 px;`
Correct (nospace): `margin-left: 20px;`

Internal CSS

An internal style sheet may be used if one single HTML page has a unique style.

The internal style is defined inside the <style> element, inside the head section.

Example

Internal styles are defined within the <style> element, inside the <head> section of an HTML page:

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

h1 {
  color: maroon;
  margin-left: 40px;
}
</style>
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

Try it Yourself »

Inline CSS

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

Example

Inline styles are defined within the "style" attribute of the relevant element:

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

<h1 style="color:blue;text-align:center;">This is a heading</h1>
<p style="color:red;">This is a paragraph.</p>

</body>
</html>
```

Try it Yourself »

CSS styling

CSS Backgrounds

The CSS background properties are used to add background effects for elements.

following CSS background properties:

- background-color
- background-image
- background-repeat
- background-attachment
- background-position
- background (shorthand property)

CSS background-color

The background-color property specifies the background color of an element.

Example

The background color of a page is set like this:

```
body {
  background-color: lightblue;
}
```

Try it Yourself »

With CSS, a color is most often specified by:

- a valid color name - like "red"
- a HEX value - like "#ff0000"
- an RGB value - like "rgb(255,0,0)"

Look at CSS Color Values for a complete list of possible color values.

Other Elements

You can set the background color for any HTML elements:

Example

Here, the <h1>, <p>, and <div> elements will have different background colors:

```
h1 {
  background-color: green;
}

div {
  background-color: lightblue;
}

p {
  background-color: yellow;
}
```

Try it Yourself »

Opacity / Transparency

The opacity property specifies the opacity/transparency of an element. It can take a value from 0.0 - 1.0. The lower value, the more transparent:

opacity 1

opacity 0.6

opacity 0.3

opacity 0.1

Example

```
div {
  background-color: green;
  opacity: 0.3;
}
```

Try it Yourself »

Note: When using the opacity property to add transparency to the background of an element, all of its child elements inherit the same transparency. This can make the text inside a fully transparent element hard to read.

Transparency using RGBA

If you do not want to apply opacity to child elements, like in our example above, use **RGBA** color values. The following example sets the opacity for the background color and not the text:

100% opacity

60% opacity

30% opacity

10% opacity

You learned from our CSS Colors Chapter, that you can use RGB as a color value. In addition to RGB, you can use an RGB color value with an **alpha** channel (RGB**A**) - which specifies the opacity for a color.

An RGBA color value is specified with: rgba(red, green, blue, *alpha*). The *alpha* parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque).

Example

```
div {
  background: rgba(0, 128, 0, 0.3) /* Green background with 30% opacity */
}
```

Try it Yourself »

CSS Background Image

The background-image property specifies an image to use as the background of an element.

By default, the image is repeated so it covers the entire element.

Example

Set the background image for a page:

```
body {
  background-image: url("paper.gif");
}
```

Try it Yourself »

Example

This example shows a **bad combination** of text and background image. The text is hardly readable:

```
body {
  background-image: url("bgdesert.jpg");
}
```

Try it Yourself »

Note: When using a background image, use an image that does not disturb the text.

The background image can also be set for specific elements, like the <p> element:

Example

```
p {
  background-image: url("paper.gif");
}
```

Try it Yourself »

CSS Background Image Repeat

By default, the background-image property repeats an image both horizontally and vertically.

Some images should be repeated only horizontally or vertically, or they will look strange, like this:

Example

```
body {
  background-image: url("gradient_bg.png");
}
```

Try it Yourself »

If the image above is repeated only horizontally (background-repeat: repeat-x;), the background will look better:

Example

```
body {
  background-image: url("gradient_bg.png");
  background-repeat: repeat-x;
}
```

Try it Yourself »

Tip: To repeat an image vertically, set background-repeat: repeat-y;

CSS background-repeat: no-repeat

Showing the background image only once is also specified by the background-repeat property:

Example

Show the background image only once:

```
body {
  background-image: url("img_tree.png");
  background-repeat: no-repeat;
}
```

Try it Yourself »

In the example above, the background image is placed in the same place as the text. We want to change the position of the image, so that it does not disturb the text too much.

CSS background-position

The background-position property is used to specify the position of the background image.

Example

Position the background image in the top-right corner:

```
body {
  background-image: url("img_tree.png");
  background-repeat: no-repeat;
  background-position: right top;
}
```

Try it Yourself »

The CSS Background Repeat and Position Properties

CSS background-attachment

The background-attachment property specifies whether the background image should scroll or be fixed (will not scroll with the rest of the page):

Example

Specify that the background image should be fixed:

```
body {
  background-image: url("img_tree.png");
  background-repeat: no-repeat;
  background-position: right top;
  background-attachment: fixed;
}
```

Try it Yourself »

Example

Specify that the background image should scroll with the rest of the page:

```
body {
  background-image: url("img_tree.png");
  background-repeat: no-repeat;
  background-position: right top;
  background-attachment: scroll;
}
```

Try it Yourself »

The CSS Background Attachment Property

Property	Description
background-attachment	Sets whether a background image is fixed or scrolls with the rest of the page

CSS background - Shorthand property

To shorten the code, it is also possible to specify all the background properties in one single property. This is called a shorthand property.

Instead of writing:

```
body {
  background-color: #ffffff;
  background-image: url("img_tree.png");
  background-repeat: no-repeat;
  background-position: right top;
}
```

You can use the shorthand property background:

Example

Use the shorthand property to set the background properties in one declaration:

```
body {
  background: #ffffff url("img_tree.png") no-repeat right top;
}
```

Try it Yourself »

When using the shorthand property the order of the property values is:

- background-color
- background-image
- background-repeat
- background-attachment
- background-position

It does not matter if one of the property values is missing, as long as the other ones are in this order. Note that we do not use the background-attachment property in the examples above, as it does not have a value.

All CSS Background Properties

Property	Description
background	Sets all the background properties in one declaration
background-attachment	Sets whether a background image is fixed or scrolls with the rest of the page
background-clip	Specifies the painting area of the background
background-color	Sets the background color of an element
background-image	Sets the background image for an element
background-origin	Specifies where the background image(s) is/are positioned
background-position	Sets the starting position of a background image
background-repeat	Sets how a background image will be repeated

background-size	Specifies the size of the background image(s)

CSS 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)"

Look at CSS Color Values for a complete list of possible color values.

The default text color for a page is defined in the body selector.

Example

```
body {
  color: blue;
}

h1 {
  color: green;
}
```

Try it Yourself »

Text Color and Background Color

In this example, we define both the background-color property and the color property:

Example

```
body {
  background-color: lightgrey;
  color: blue;
}

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

div {
  background-color: blue;
  color: white;
}
```

Try it Yourself »

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!

Text Alignment and Text Direction

learn about the following properties:

- text-align
- text-align-last
- direction
- unicode-bidi
- vertical-align

Text Alignment

The text-align property is used to set the horizontal alignment of a text.

A text can be left or right aligned, centered, or justified.

The following example shows center aligned, and left and right aligned text (left alignment is default if text direction is left-to-right, and right alignment is default if text direction is right-to-left):

Example

```
h1 {
  text-align: center;
}

h2 {
  text-align: left;
}

h3 {
  text-align: right;
}
```

Try it Yourself »

When the text-align property is set to "justify", each line is stretched so that every line has equal width, and the left and right margins are straight (like in magazines and newspapers):

Example

```
div {
  text-align: justify;
}
```

Try it Yourself »

Text Align Last

The text-align-last property specifies how to align the last line of a text.

Example

Align the last line of text in three <p> elements:

```
p.a {
  text-align-last: right;
}

p.b {
  text-align-last: center;
}

p.c {
  text-align-last: justify;
}
```

Try it Yourself »

Text Direction

The direction and unicode-bidi properties can be used to change the text direction of an element:

Example

```
p {
  direction: rtl;
  unicode-bidi: bidi-override;
}
```

Try it Yourself »

Vertical Alignment

The vertical-align property sets the vertical alignment of an element.

Example

Set the vertical alignment of an image in a text:

```
img.a {
  vertical-align: baseline;
}

img.b {
  vertical-align: text-top;
}

img.c {
  vertical-align: text-bottom;
}

img.d {
  vertical-align: sub;
}

img.e {
  vertical-align: super;
}
```

Try it Yourself »

The CSS Text Alignment/Direction Properties

Property	Description
direction	Specifies the text direction/writing direction
text-align	Specifies the horizontal alignment of text
text-align-last	Specifies how to align the last line of a text
unicode-bidi	Used together with the direction property to set or return whether the text should be overridden to support multiple languages in the same document
vertical-align	Sets the vertical alignment of an element

Text Decoration

learn about the following properties:

- text-decoration-line
- text-decoration-color
- text-decoration-style
- text-decoration-thickness
- text-decoration

Add a Decoration Line to Text

The text-decoration-line property is used to add a decoration line to text.

Tip: You can combine more than one value, like overline and underline to display lines both over and under a text.

Example

```
h1 {
  text-decoration-line: overline;
}

h2 {
  text-decoration-line: line-through;
}

h3 {
  text-decoration-line: underline;
}

p {
  text-decoration-line: overline underline;
}
```

Try it Yourself »

Note: It is not recommended to underline text that is not a link, as this often confuses the reader.

Specify a Color for the Decoration Line

The text-decoration-color property is used to set the color of the decoration line.

Example

```
h1 {
  text-decoration-line: overline;
  text-decoration-color: red;
}
```

```
h2 {
  text-decoration-line: line-through;
  text-decoration-color: blue;
}

h3 {
  text-decoration-line: underline;
  text-decoration-color: green;
}

p {
  text-decoration-line: overline underline;
  text-decoration-color: purple;
}
```

Try it Yourself »

Specify a Style for the Decoration Line

The text-decoration-style property is used to set the style of the decoration line.

Example

```
h1 {
  text-decoration-line: underline;
  text-decoration-style: solid;
}

h2 {
  text-decoration-line: underline;
  text-decoration-style: double;
}

h3 {
  text-decoration-line: underline;
  text-decoration-style: dotted;
}

p.ex1 {
  text-decoration-line: underline;
  text-decoration-style: dashed;
}

p.ex2 {
  text-decoration-line: underline;
  text-decoration-style: wavy;
}

p.ex3 {
  text-decoration-line: underline;
  text-decoration-color: red;
  text-decoration-style: wavy;
}
```

Try it Yourself »

Specify the Thickness for the Decoration Line

The text-decoration-thickness property is used to set the thickness of the decoration line.

Example

```
h1 {
  text-decoration-line: underline;
  text-decoration-thickness: auto;
}

h2 {
  text-decoration-line: underline;
  text-decoration-thickness: 5px;
}

h3 {
  text-decoration-line: underline;
  text-decoration-thickness: 25%;
}

p {
  text-decoration-line: underline;
  text-decoration-color: red;
  text-decoration-style: double;
  text-decoration-thickness: 5px;
}
```

Try it Yourself »

The Shorthand Property

The text-decoration property is a shorthand property for:

- text-decoration-line (required)
- text-decoration-color (optional)
- text-decoration-style (optional)
- text-decoration-thickness (optional)

Example

```
h1 {
  text-decoration: underline;
}

h2 {
  text-decoration: underline red;
}

h3 {
  text-decoration: underline red double;
}

p {
  text-decoration: underline red double 5px;
}
```

Try it Yourself »

All CSS text-decoration Properties

Property	Description
text-decoration	Sets all the text-decoration properties in one declaration
text-decoration-color	Specifies the color of the text-decoration
text-decoration-line	Specifies the kind of text decoration to be used (underline, overline, etc.)
text-decoration-style	Specifies the style of the text decoration (solid, dotted, etc.)
text-decoration-thickness	Specifies the thickness of the text decoration line

Text Transformation

The text-transform property is used to specify uppercase and lowercase letters in a text.

It can be used to turn everything into uppercase or lowercase letters, or capitalize the first letter of each word:

Example

```
p.uppercase {
  text-transform: uppercase;
}

p.lowercase {
  text-transform: lowercase;
}

p.capitalize {
  text-transform: capitalize;
}
```

Try it Yourself »

The CSS Text Transformation Property

Property	Description
text-transform	Controls the capitalization of text

Text Spacing

In this chapter you will learn about the following properties:

- text-indent
- letter-spacing
- line-height
- word-spacing
- white-space

Text Indentation

The text-indent property is used to specify the indentation of the first line of a text:

Example

```
p {
  text-indent: 50px;
}
```

Try it Yourself »

Letter Spacing

The letter-spacing property is used to specify the space between the characters in a text.

The following example demonstrates how to increase or decrease the space between characters:

Example

```
h1 {
  letter-spacing: 5px;
}

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

Try it Yourself »

Line Height

The line-height property is used to specify the space between lines:

Example

```
p.small {
  line-height: 0.8;
}

p.big {
  line-height: 1.8;
}
```

Try it Yourself »

Word Spacing

The word-spacing property is used to specify the space between the words in a text.

The following example demonstrates how to increase or decrease the space between words:

Example

```
p.one {
  word-spacing: 10px;
}

p.two {
  word-spacing: -2px;
}
```

Try it Yourself »

White Space

The white-space property specifies how white-space inside an element is handled.

This example demonstrates how to disable text wrapping inside an element:

Example

```
p {
  white-space: nowrap;
}
```

Try it Yourself »

The CSS Text Spacing Properties

Property	Description
letter-spacing	Specifies the space between characters in a text
line-height	Specifies the line height
text-indent	Specifies the indentation of the first line in a text-block
white-space	Specifies how to handle white-space inside an element
word-spacing	Specifies the space between words in a text

Text Shadow

The text-shadow property adds shadow to text.

In its simplest use, you only specify the horizontal shadow (2px) and the vertical shadow (2px):

Text shadow effect

Example

```
h1 {
  text-shadow: 2px 2px;
}
```

Try it Yourself »

Next, add a color (red) to the shadow:

Text shadow effect!

Example

```
h1 {
  text-shadow: 2px 2px red;
}
```

Try it Yourself »

Then, add a blur effect (5px) to the shadow:

Text shadow effect!

Example

```
h1 {
  text-shadow: 2px 2px 5px red;
}
```

Try it Yourself »

More Text Shadow Examples

Example 1

Text-shadow on a white text:

```
h1 {
  color: white;
  text-shadow: 2px 2px 4px #000000;
}
```

Try it Yourself »

Example 2

Text-shadow with red neon glow:

```
h1 {
  text-shadow: 0 0 3px #ff0000;
}
```

Try it Yourself »

Example 3

Text-shadow with red and blue neon glow:

```
h1 {
  text-shadow: 0 0 3px #ff0000, 0 0 5px #0000ff;
}
```

Try it Yourself »

Example 4

```
h1 {
  color: white;
  text-shadow: 1px 1px 2px black, 0 0 25px blue, 0 0 5px darkblue;
}
```

Try it Yourself »

Tip: Go to our CSS Fonts chapter to learn about how to change fonts, text size and the style of a text.

Tip: Go to our CSS Text Effects chapter to learn about different text effects.

CSS Fonts

Font Selection is Important

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

The right font can create a strong identity for your brand.

Using a font that is easy to read is important. The font adds value to your text. It is also important to choose the correct color and text size for the font.

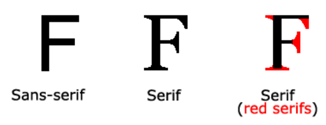
Generic Font Families

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.

All the different font names belong to one of the generic font families.

Difference Between Serif and Sans-serif Fonts



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

Some Font Examples

Generic Font Family	**Examples of Font Names**
Serif	Times New Roman, Georgia, Garamond
Sans-serif	Arial, Verdana, elvetica
Monospace	Courier New, Lucida Console, Monaco
Cursive	Brush Script MT, Lucida Handwriting
Fantasy	Copperplate, Papyrus

The CSS font-family Property

In CSS, we use the font-family property to specify the font of a text.

Note: If the font name is more than one word, it must be in quotation marks, like: "Times New Roman".

Tip: The font-family property should hold several font names as a "fallback" system, to ensure maximum compatibility between browsers/operating systems. Start with the font you want, and end with a generic family (to let the browser pick a similar font in the generic family, if no other fonts are available). The font names should be separated with comma. Read more about fallback fonts in the next chapter.

Example

Specify some different fonts for three paragraphs:

```
.p1 {
  font-family: "Times New Roman", Times, serif;
}

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

.p3 {
  font-family: "Lucida Console", "Courier New", monospace;
}
```

Try it Yourself »

What are Web Safe Fonts?

Web safe fonts are fonts that are universally installed across all browsers and devices.

Fallback Fonts

However, there are no 100% completely web safe fonts. There is always a chance that a font is not found or is not installed properly.

Therefore, it is very important to always use fallback fonts.

This means that you should add a list of similar "backup fonts" in the font-family property. If the first font does not work, the browser will try the next one, and the next one, and so on. Always end the list with a generic font family name.

Example

Here, there are three font types: Tahoma, Verdana, and sans-serif. The second and third fonts are backups, in case the first one is not found.

```
p {
font-family: Tahoma, Verdana, sans-serif;
}
```

Try it Yourself »

Best Web Safe Fonts for HTML and CSS

The following list are the best web safe fonts for HTML and CSS:

- Arial (sans-serif)
- Verdana (sans-serif)
- Tahoma (sans-serif)
- Trebuchet MS (sans-serif)
- Times New Roman (serif)
- Georgia (serif)
- Garamond (serif)
- Courier New (monospace)
- Brush Script MT (cursive)

Note: Before you publish your website, always check how your fonts appear on different browsers and devices, and always use fallback fonts!

Arial (sans-serif)

Arial is the most widely used font for both online and printed media. Arial is also the default font in Google Docs.

Arial is one of the safest web fonts, and it is available on all major operating systems.

Example

Arial Font of Sans-Serif

Arial font of Sans-Serif.

0 1 2 3 4 5 6 7 8 9

Try it Yourself »

Verdana (sans-serif)

Verdana is a very popular font. Verdana is easily readable even for small font sizes.

Example

Verdana Font of Sans-Serif

Verdana font of Sans-Serif.

0 1 2 3 4 5 6 7 8 9

Try it Yourself »

Tahoma (sans-serif)

The Tahoma font has less space between the characters.

Example

Tahoma Font of Sans-Serif

Tahoma font of Sans-Serif.

0 1 2 3 4 5 6 7 8 9

Try it Yourself »

Trebuchet MS (sans-serif)

Trebuchet MS was designed by Microsoft in 1996. Use this font carefully. Not supported by all mobile operating systems.

Try it Yourself »

Times New Roman (serif)

Times New Roman is one of the most recognizable fonts in the world. It looks professional and is used in many newspapers and "news" websites. It is also the primary font for Windows devices and applications.

Try it Yourself »

Georgia (serif)

Georgia is an elegant serif font. It is very readable at different font sizes, so it is a good candidate for mobile-responsive design.

Try it Yourself »

Garamond (serif)

Garamond is a classical font used for many printed books. It has a timeless look and good readability.

Example

Try it Yourself »

Courier New (monospace)

Courier New is the most widely used monospace serif font. Courier New is often used with coding displays, and many email providers use it as their default font. Courier New is also the standard font for movie screenplays.

Try it Yourself »

Brush Script MT (cursive)

The Brush Script MT font was designed to mimic handwriting. It is elegant and sophisticated, but can be hard to read. Use it carefully.

Example

Try it Yourself »

Font Style

The font-style property is mostly used to specify italic text.

This property has three values:

- normal - The text is shown normally
- italic - The text is shown in italics
- oblique - The text is "leaning" (oblique is very similar to italic, but less supported)

Example

```
p.normal {
  font-style: normal;
}

p.italic {
  font-style: italic;
}

p.oblique {
  font-style: oblique;
}
```

Try it Yourself »

Font Weight

The font-weight property specifies the weight of a font:

Example

```
p.normal {
  font-weight: normal;
}

p.thick {
  font-weight: bold;
}
```

Try it Yourself »

Font Variant

The font-variant property specifies whether or not a text should be displayed in a small-caps font.

In a small-caps font, all lowercase letters are converted to uppercase letters. However, the converted uppercase letters appears in a smaller font size than the original uppercase letters in the text.

Example

```
p.normal {
  font-variant: normal;
}

p.small {
  font-variant: small-caps;
}
```

Try it Yourself »

Font Size

The font-size property sets the size of the text.

Being able to manage the text size is important in web design. However, you should not use font size adjustments to make paragraphs look like headings, or headings look like paragraphs.

Always use the proper HTML tags, like <h1> - <h6> for headings and <p> for paragraphs.

The font-size value can be an absolute, or relative size.

Absolute size:

- Sets the text to a specified size
- Does not allow a user to change the text size in all browsers (bad for accessibility reasons)
- Absolute size is useful when the physical size of the output is known

Relative size:

- Sets the size relative to surrounding elements
- Allows a user to change the text size in browsers

Note: If you do not specify a font size, the default size for normal text, like paragraphs, is 16px (16px=1em).

Set Font Size With Pixels

Setting the text size with pixels gives you full control over the text size:

Example

```
h1 {
  font-size: 40px;
}

h2 {
  font-size: 30px;
}

p {
  font-size: 14px;
}
```

Try it Yourself »

Tip: If you use pixels, you can still use the zoom tool to resize the entire page.

Set Font Size With Em

To allow users to resize the text (in the browser menu), many developers use em instead of pixels.

1em is equal to the current font size. The default text size in browsers is 16px. So, the default size of 1em is 16px.

The size can be calculated from pixels to em using this formula: *pixels*/16=*em*

Example

```
h1 {
  font-size: 2.5em; /* 40px/16=2.5em */
}

h2 {
  font-size: 1.875em; /* 30px/16=1.875em */
}

p {
  font-size: 0.875em; /* 14px/16=0.875em */
}
```

Try it Yourself »

In the example above, the text size in em is the same as the previous example in pixels. However, with the em size, it is possible to adjust the text size in all browsers.

Unfortunately, there is still a problem with older versions of Internet Explorer. The text becomes larger than it should when made larger, and smaller than it should when made smaller.

Use a Combination of Percent and Em

The solution that works in all browsers, is to set a default font-size in percent for the <body> element:

Example

```
body {
  font-size: 100%;
}

h1 {
  font-size: 2.5em;
}

h2 {
  font-size: 1.875em;
}

p {
  font-size: 0.875em;
}
```

Try it Yourself »

Our code now works great! It shows the same text size in all browsers, and allows all browsers to zoom or resize the text!

All CSS Font Properties

Property	Description
font	Sets all the font properties in one declaration
font-family	Specifies the font family for text
font-size	Specifies the font size of text

font-style	Specifies the font style for text
font-variant	Specifies whether or not a text should be displayed in a small-caps font
font-weight	Specifies the weight of a font

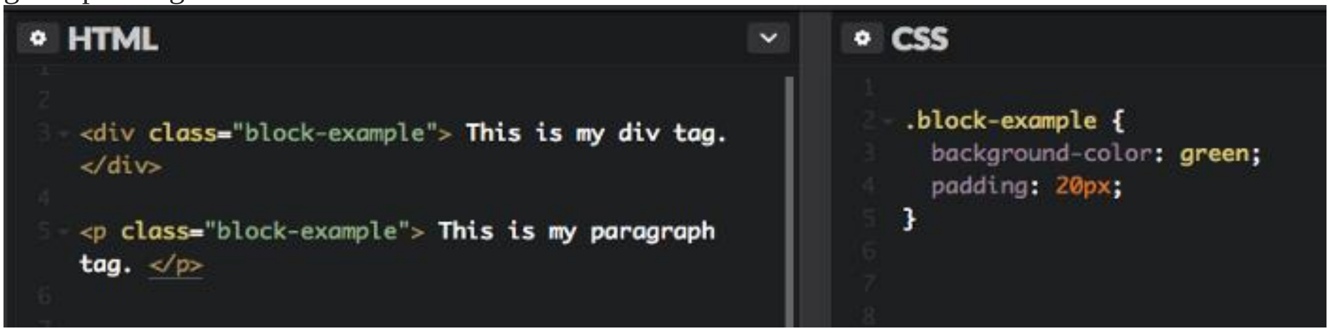
CSS Block Elements

CSS display properties: block, inline, and inline-block

Web browsers treat every element as a kind of box. However, CSS has two different types of boxes — block and inline.

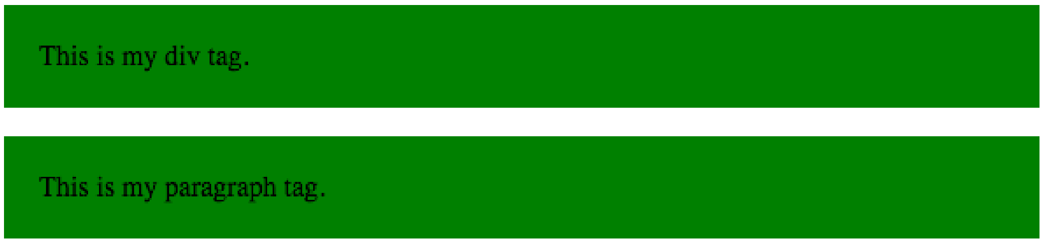
Block Elements-A block element always starts on a new line, and fills up the horizontal space left and right on the web page. You can add margins and padding on all four sides of any block element — top, right, left, and bottom.

Some examples of block elements are <**div**> and <**p**> tags. As shown below, I've also added green padding on all four sides of each block element.



```html
<div class="block-example"> This is my div tag.
</div>

<p class="block-example"> This is my paragraph
tag. </p>
```

```css
.block-example {
  background-color: green;
  padding: 20px;
}
```

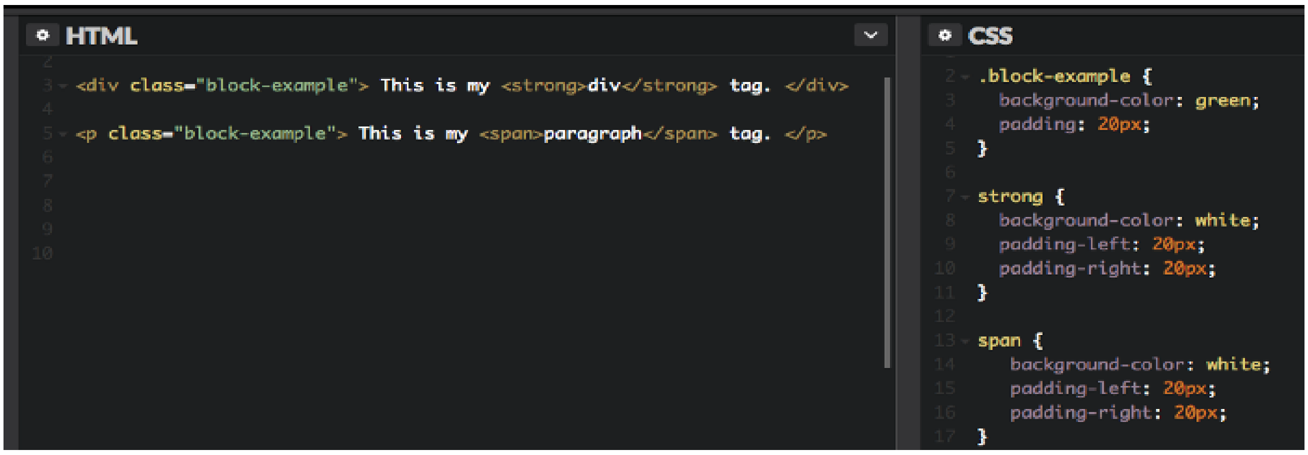


This is my div tag.

This is my paragraph tag.

Inline Elements-Inline elements don't start on a new line, they appear on the same line as the content and tags beside them. Some examples of inline elements are **<span>** , **<strong>**, and **<img>** tags.

When it comes to margins and padding, browsers treat inline elements differently. You can add space to the left and right on an inline element, but you cannot add height to the top or bottom padding or margin of an inline element.Inline elements can actually appear within block elements, as shown below. I've added white padding on the left and right side of each inline element.

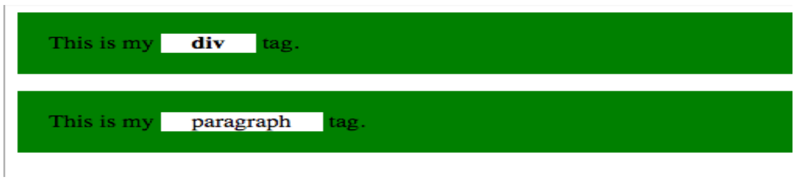


HTML

```
<div class="block-example"> This is my <strong>div</strong> tag. </div>

<p class="block-example"> This is my <span>paragraph</span> tag. </p>
```

CSS

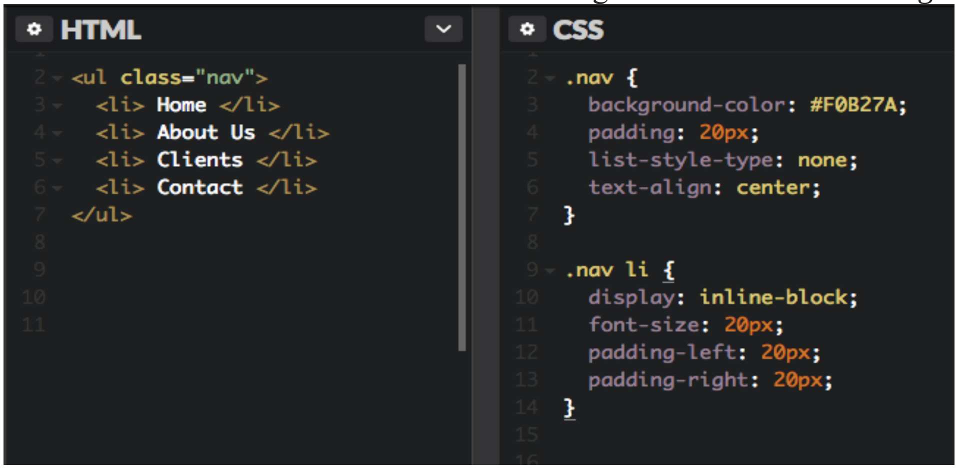
```
.block-example {
  background-color: green;
  padding: 20px;
}

strong {
  background-color: white;
  padding-left: 20px;
  padding-right: 20px;
}

span {
   background-color: white;
   padding-left: 20px;
   padding-right: 20px;
}
```



This is my **div** tag.

This is my paragraph tag.

Inline-Block-Inline-block elements are similar to inline elements, except they can have padding and margins added on all four sides. You'll have to declare display: inline-block in your CSS code.One common use for using inline-block is for creating navigation links horizontally, as shown below. I've created a horizontal navigation menu with an orange background color.



HTML

```
<ul class="nav">
  <li> Home </li>
  <li> About Us </li>
  <li> Clients </li>
  <li> Contact </li>
</ul>
```

CSS

```
.nav {
  background-color: #F0B27A;
  padding: 20px;
  list-style-type: none;
  text-align: center;
}

.nav li {
  display: inline-block;
  font-size: 20px;
  padding-left: 20px;
  padding-right: 20px;
}
```

CSS Lists

Unordered Lists:

- ○ Coffee
- ○ Tea
- ○ Coca Cola

- ▪ Coffee
- ▪ Tea
- ▪ Coca Cola

Ordered Lists:

1. Coffee
2. Tea
3. Coca Cola

I. Coffee
II. Tea
III. Coca Cola

HTML Lists and CSS List Properties

In HTML, there are two main types of lists:

- unordered lists (<ul>) - the list items are marked with bullets
- ordered lists (<ol>) - the list items are marked with numbers or letters

The CSS list properties allow you to:

- Set different list item markers for ordered lists
- Set different list item markers for unordered lists
- Set an image as the list item marker
- Add background colors to lists and list items

Different List Item Markers

The list-style-type property specifies the type of list item marker.

The following example shows some of the available list item markers:

Example

```
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;
}
```

Try it Yourself »

Note: Some of the values are for unordered lists, and some for ordered lists.

An Image as The List Item Marker

The list-style-image property specifies an image as the list item marker:

Example

```
ul {
  list-style-image: url('sqpurple.gif');
}
```

Try it Yourself »

Position the List Item Markers

The list-style-position property specifies the position of the list-item markers (bullet points).

"list-style-position: outside;" means that the bullet points will be outside the list item. The start of each line of a list item will be aligned vertically. This is default:

- Coffee - A brewed drink prepared from roasted coffee beans...
- Tea
- Coca-cola

"list-style-position: inside;" means that the bullet points will be inside the list item. As it is part of the list item, it will be part of the text and push the text at the start:

- Coffee - A brewed drink prepared from roasted coffee beans...
- Tea
- Coca-cola

Example

```
ul.a {
  list-style-position: outside;
}

ul.b {
  list-style-position: inside;
}
```

Try it Yourself »

Remove Default Settings

The list-style-type:none property can also be used to remove the markers/bullets. Note that the list also has default margin and padding. To remove this, add margin:0 and padding:0 to <ul> or <ol>:

Example

```
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
}
```

Try it Yourself »

List - Shorthand property

The list-style property is a shorthand property. It is used to set all the list properties in one declaration:

Example

```
ul {
  list-style: square inside url("sqpurple.gif");
}
```

Try it Yourself »

When using the shorthand property, the order of the property values are:

- list-style-type (if a list-style-image is specified, the value of this property will be displayed if the image for some reason cannot be displayed)
- list-style-position (specifies whether the list-item markers should appear inside or outside the content flow)
- list-style-image (specifies an image as the list item marker)

If one of the property values above is missing, the default value for the missing property will be inserted, if any.

Styling List With Colors

We can also style lists with colors, to make them look a little more interesting.

Anything added to the <ol> or <ul> tag, affects the entire list, while properties added to the <li> tag will affect the individual list items:

Example

```
ol {
  background: #ff9999;
  padding: 20px;
}

ul {
  background: #3399ff;
  padding: 20px;
}

ol li {
  background: #ffe5e5;
  color: darkred;
  padding: 5px;
  margin-left: 35px;
}

ul li {
  background: #cce5ff;
  color: darkblue;
  margin: 5px;
}
```

Result:

1. Coffee
2. Tea
3. Coca Cola

- Coffee
- Tea
- Coca Cola

Try it Yourself »

All CSS List Properties

Property	Description
list-style	Sets all the properties for a list in one declaration
list-style-image	Specifies an image as the list-item marker

list-style-position	Specifies the position of the list-item markers (bullet points)
list-style-type	Specifies the type of list-item marker

CSS Tables

Try it Yourself »

Table Borders

To specify table borders in CSS, use the border property.

The example below specifies a solid border for <table>, <th>, and <td> elements:

Firstname	Lastname
Peter	Griffin
Lois	Griffin

Example

```
table, th, td {
  border: 1px solid;
}
```

Try it Yourself »

Full-Width Table

The table above might seem small in some cases. If you need a table that should span the entire screen (full-width), add width: 100% to the <table> element:

Firstname	Lastname
Peter	Griffin
Lois	Griffin

Example

```
table {
  width: 100%;
}
```

Try it Yourself »

Double Borders

Notice that the table in the examples above have double borders. This is because both the table and the <th> and <td> elements have separate borders.

To remove double borders, take a look at the example below.

Collapse Table Borders

The border-collapse property sets whether the table borders should be collapsed into a single border:

Firstname	Lastname
Peter	Griffin
Lois	Griffin

Example

```
table {
  border-collapse: collapse;
}
```

Try it Yourself »

If you only want a border around the table, only specify the border property for <table>:

Firstname	Lastname
Peter	Griffin
Lois	Griffin

Example

```
table {
  border: 1px solid;
}
```

Try it Yourself »

CSS Table Size

Table Width and Height

The width and height of a table are defined by the width and height properties.

The example below sets the width of the table to 100%, and the height of the <th> elements to 70px:

Firstname	Lastname	Savings
Peter	Griffin	$100
Lois	Griffin	$150
Joe	Swanson	$300

Example

```
table {
  width: 100%;
}

th {
  height: 70px;
}
```

Try it Yourself »

To create a table that should only span half the page, use width: 50%:

Firstname	Lastname	Savings
Peter	Griffin	$100
Lois	Griffin	$150
Joe	Swanson	$300

Example

```
table {
  width: 50%;
}
```

Try it Yourself »

CSS Table Alignment

Horizontal Alignment

The text-align property sets the horizontal alignment (like left, right, or center) of the content in <th> or <td>.

By default, the content of <th> elements are center-aligned and the content of <td> elements are left-aligned.

To center-align the content of <td> elements as well, use text-align: center:

Firstname	Lastname	Savings
Peter	Griffin	$100
Lois	Griffin	$150
Joe	Swanson	$300

Example

```
td {
  text-align: center;
}
```

Try it Yourself »

To left-align the content, force the alignment of <th> elements to be left-aligned, with the text-align: left property:

Firstname	Lastname	Savings
Peter	Griffin	$100
Lois	Griffin	$150
Joe	Swanson	$300

Example

```
th {
  text-align: left;
}
```

Try it Yourself »

Vertical Alignment

The vertical-align property sets the vertical alignment (like top, bottom, or middle) of the content in <th> or <td>.

By default, the vertical alignment of the content in a table is middle (for both <th> and <td> elements).

The following example sets the vertical text alignment to bottom for <td> elements:

Firstname	Lastname	Savings
Peter	Griffin	$100
Lois	Griffin	$150
Joe	Swanson	$300

Example

```
td {
  height: 50px;
  vertical-align: bottom;
}
```

Try it Yourself »

CSS Table Style

Table Padding

To control the space between the border and the content in a table, use the padding property on <td> and <th> elements:

First Name	Last Name	Savings
Peter	Griffin	$100
Lois	Griffin	$150
Joe	Swanson	$300

Example

```
th, td {
  padding: 15px;
  text-align: left;
}
```

Try it Yourself »

Horizontal Dividers

First Name	Last Name	Savings
Peter	Griffin	$100
Lois	Griffin	$150
Joe	Swanson	$300

Add the border-bottom property to <th> and <td> for horizontal dividers:

Example

```
th, td {
  border-bottom: 1px solid #ddd;
}
```

Try it Yourself »

Hoverable Table

Use the :hover selector on <tr> to highlight table rows on mouse over:

First Name	Last Name	Savings
Peter	Griffin	$100

Lois	Griffin	$150
Joe	Swanson	$300

Example

```
tr:hover {background-color: coral;}
```

Try it Yourself »

Striped Tables

First Name	Last Name	Savings
Peter	Griffin	$100
Lois	Griffin	$150
Joe	Swanson	$300

For zebra-striped tables, use the nth-child() selector and add a background-color to all even (or odd) table rows:

Example

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

Try it Yourself »

Table Color

The example below specifies the background color and text color of <th> elements:

First Name	Last Name	Savings
Peter	Griffin	$100
Lois	Griffin	$150
Joe	Swanson	$300

Example

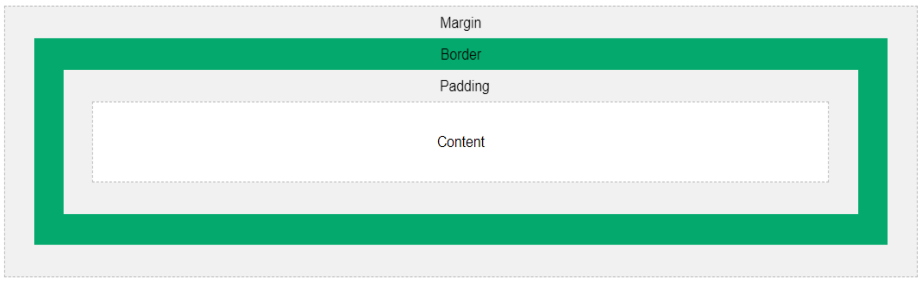
```
th {
  background-color: #04AA6D;
  color: white;
}
```

Try it Yourself »

CSS Box Model

In CSS, the term "box model" is used when talking about design and layout.

The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:



Explanation of the different parts:

- **Content** - The content of the box, where text and images appear
- **Padding** - Clears an area around the content. The padding is transparent
- **Border** - A border that goes around the padding and content
- **Margin** - Clears an area outside the border. The margin is transparent

The box model allows us to add a border around elements, and to define space between elements.

Example

Demonstration of the box model:

```
div {
  width: 300px;
  border: 15px solid green;
  padding: 50px;
  margin: 20px;
}
```

Try it Yourself »

Width and Height of an Element

In order to set the width and height of an element correctly in all browsers, you need to know how the box model works.

Important: When you set the width and height properties of an element with CSS, you just set the width and height of the **content area**. To calculate the full size of an element, you must also add padding, borders and margins.

Example

This <div> element will have a total width of 350px:

```
div {
  width: 320px;
  padding: 10px;
  border: 5px solid gray;
  margin: 0;
}
```

Try it Yourself »

Here is the calculation:

320px (width)
+ 20px (left + right padding)
+ 10px (left + right border)

+ 0px (left + right margin)
= 350px

The total width of an element should be calculated like this:

Total element width = width + left padding + right padding + left border + right border + left margin + right margin

The total height of an element should be calculated like this:

Total element height = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin

CSS Margins

CSS Margins

The CSS margin properties are used to create space around elements, outside of any defined borders.

With CSS, you have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).

Margin - Individual Sides

CSS has properties for specifying the margin for each side of an element:

- margin-top
- margin-right
- margin-bottom
- margin-left

All the margin properties can have the following values:

- auto - the browser calculates the margin
- *length* - specifies a margin in px, pt, cm, etc.
- *%* - specifies a margin in % of the width of the containing element
- inherit - specifies that the margin should be inherited from the parent element

Tip: Negative values are allowed.

Example

Set different margins for all four sides of a <p> element:

```
p {
  margin-top: 100px;
  margin-bottom: 100px;
  margin-right: 150px;
  margin-left: 80px;
}
```

Try it Yourself »

Margin - Shorthand Property

To shorten the code, it is possible to specify all the margin properties in one property.

The margin property is a shorthand property for the following individual margin properties:

- margin-top
- margin-right
- margin-bottom
- margin-left

So, here is how it works:

If the margin property has four values:

- **margin: 25px 50px 75px 100px;**
    - top margin is 25px
    - right margin is 50px
    - bottom margin is 75px
    - left margin is 100px

Example

Use the margin shorthand property with four values:

```
p {
  margin: 25px 50px 75px 100px;
}
```

Try it Yourself »

If the margin property has three values:

- **margin: 25px 50px 75px;**
    - top margin is 25px
    - right and left margins are 50px
    - bottom margin is 75px

Example

Use the margin shorthand property with three values:

```
p {
  margin: 25px 50px 75px;
}
```

Try it Yourself »

If the margin property has two values:

- **margin: 25px 50px;**
  - top and bottom margins are 25px
  - right and left margins are 50px

Example

Use the margin shorthand property with two values:

```
p {
  margin: 25px 50px;
}
```

Try it Yourself »

If the margin property has one value:

- **margin: 25px;**
  - all four margins are 25px

Example

Use the margin shorthand property with one value:

```
p {
  margin: 25px;
}
```

Try it Yourself »

The auto Value

You can set the margin property to auto to horizontally center the element within its container.

The element will then take up the specified width, and the remaining space will be split equally between the left and right margins.

Example

Use margin: auto:

```
div {
  width: 300px;
  margin: auto;
  border: 1px solid red;
}
```

Try it Yourself »

The inherit Value

This example lets the left margin of the <p class="ex1"> element be inherited from the parent element (<div>):

Example

Use of the inherit value:

```
div {
  border: 1px solid red;
  margin-left: 100px;
}

p.ex1 {
  margin-left: inherit;
}
```

Try it Yourself »

All CSS Margin Properties

Property	Description

margin	A shorthand property for setting all the margin properties in one declaration
margin-bottom	Sets the bottom margin of an element
margin-left	Sets the left margin of an element
margin-right	Sets the right margin of an element
margin-top	Sets the top margin of an element

CSS Borders

The CSS border properties allow you to specify the style, width, and color of an element's border.

CSS Border Style

The border-style property specifies what kind of border to display.

The following values are allowed:

- dotted - Defines a dotted border
- dashed - Defines a dashed border
- solid - Defines a solid border
- double - Defines a double border
- groove - Defines a 3D grooved border. The effect depends on the border-color value
- ridge - Defines a 3D ridged border. The effect depends on the border-color value
- inset - Defines a 3D inset border. The effect depends on the border-color value
- outset - Defines a 3D outset border. The effect depends on the border-color value
- none - Defines no border
- hidden - Defines a hidden border

The border-style property can have from one to four values (for the top border, right border, bottom border, and the left border).

Example

Demonstration of the different border styles:

```
p.dotted {border-style: dotted;}
p.dashed {border-style: dashed;}
p.solid {border-style: solid;}
p.double {border-style: double;}
p.groove {border-style: groove;}
p.ridge {border-style: ridge;}
p.inset {border-style: inset;}
p.outset {border-style: outset;}
p.none {border-style: none;}
p.hidden {border-style: hidden;}
p.mix {border-style: dotted dashed solid double;}
```

Result:

A dotted border.

A dashed border.

A solid border.

A double border.

A groove border. The effect depends on the border-color value.

A ridge border. The effect depends on the border-color value.

An inset border. The effect depends on the border-color value.

An outset border. The effect depends on the border-color value.

No border.

A hidden border.

A mixed border.

Try it Yourself »

CSS Padding

Padding is used to create space around an element's content, inside of any defined borders.

This element has a padding of 70px.

Try it Yourself »

CSS Padding

The CSS padding properties are used to generate space around an element's content, inside of any defined borders.

With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).

Padding - Individual Sides

CSS has properties for specifying the padding for each side of an element:

- padding-top
- padding-right
- padding-bottom
- padding-left

All the padding properties can have the following values:

- *length* - specifies a padding in px, pt, cm, etc.
- *%* - specifies a padding in % of the width of the containing element
- inherit - specifies that the padding should be inherited from the parent element

Note: Negative values are not allowed.

Example

Set different padding for all four sides of a <div> element:

```
div {
  padding-top: 50px;
  padding-right: 30px;
  padding-bottom: 50px;
  padding-left: 80px;
}
```

Try it Yourself »

Padding - Shorthand Property

To shorten the code, it is possible to specify all the padding properties in one property.

The padding property is a shorthand property for the following individual padding properties:

- padding-top
- padding-right
- padding-bottom
- padding-left

So, here is how it works:

If the padding property has four values:

- **padding: 25px 50px 75px 100px;**
    - top padding is 25px
    - right padding is 50px
    - bottom padding is 75px
    - left padding is 100px

Example

Use the padding shorthand property with four values:

```
div {
  padding: 25px 50px 75px 100px;
}
```

Try it Yourself »

If the padding property has three values:

- **padding: 25px 50px 75px;**
    - top padding is 25px
    - right and left paddings are 50px
    - bottom padding is 75px

Example

Use the padding shorthand property with three values:

```
div {
  padding: 25px 50px 75px;
}
```

Try it Yourself »

If the padding property has two values:

- **padding: 25px 50px;**
  - top and bottom paddings are 25px
  - right and left paddings are 50px

Example

Use the padding shorthand property with two values:

```
div {
  padding: 25px 50px;
}
```

Try it Yourself »

If the padding property has one value:

- **padding: 25px;**
  - all four paddings are 25px

Example

Use the padding shorthand property with one value:

```
div {
  padding: 25px;
}
```

Try it Yourself »

Padding and Element Width

The CSS width property specifies the width of the element's content area. The content area is the portion inside the padding, border, and margin of an element (the box model).

So, if an element has a specified width, the padding added to that element will be added to the total width of the element. This is often an undesirable result.

Example

Here, the <div> element is given a width of 300px. However, the actual width of the <div> element will be 350px (300px + 25px of left padding + 25px of right padding):

```
div {
  width: 300px;
  padding: 25px;
}
```

Try it Yourself »

To keep the width at 300px, no matter the amount of padding, you can use the box-sizing property. This causes the element to maintain its actual width; if you increase the padding, the available content space will decrease.

Example

Use the box-sizing property to keep the width at 300px, no matter the amount of padding:

```
div {
  width: 300px;
  padding: 25px;
  box-sizing: border-box;
}
```

Try it Yourself »

All CSS Padding Properties

Property	Description
padding	A shorthand property for setting all the padding properties in one declaration
padding-bottom	Sets the bottom padding of an element
padding-left	Sets the left padding of an element
padding-right	Sets the right padding of an element
padding-top	Sets the top padding of an element

CSS Grouping: Grouping is used to select the multiple elements together to apply the common styling properties to them. For this reason, it helps to reduce the length of the code that has multiple selectors with the same properties. This makes code easy to read. Page load times and development time for code are reduced as well when using grouping.
Instead of writing this long code, specifying the same properties to different selectors:

```
h1 {
  padding: 5px;
  color: grey;
}
p {
  padding: 5px;
  color: grey;}
```

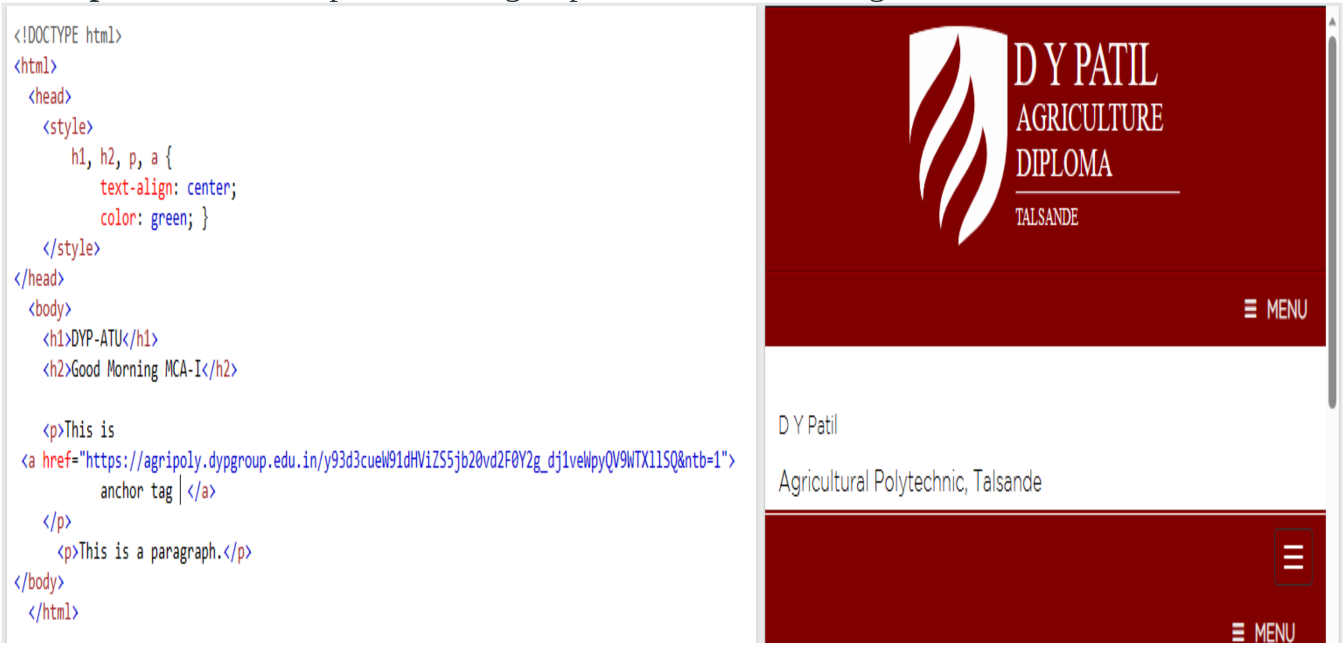
We can group them and write like this & we need the comma(**,**) to group the various selectors.

```
h1, p {
 padding: 5px;
 color: grey;
}
```

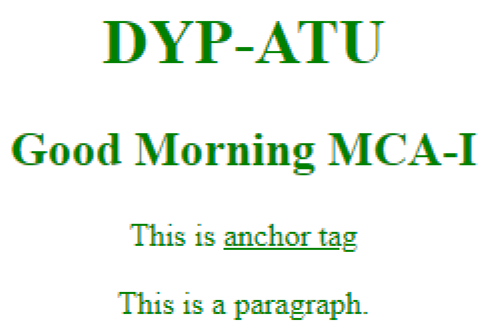
Approach:

- Add <style>tag inside <head> tag.
- Add various tags inside <body> tag with content.
- Inside <style> tag, we can group our selectors containing the same properties.

Example: In this example, we will group various selectors together.



```
<!DOCTYPE html>
<html>
  <head>
    <style>
        h1, h2, p, a {
            text-align: center;
            color: green; }
    </style>
</head>
  <body>
    <h1>DYP-ATU</h1>
    <h2>Good Morning MCA-I</h2>

    <p>This is
 <a href="https://agripoly.dypgroup.edu.in/y93d3cueW91dHViZS5jb20vd2F0Y2g_dj1veWpyQV9WTX11SQ&ntb=1">
            anchor tag </a>
    </p>
      <p>This is a paragraph.</p>
</body>
  </html>
```



DYP-ATU

Good Morning MCA-I

This is anchor tag

This is a paragraph.

CSS - Dimension

You have seen the border that surrounds every box ie. element, the padding that can appear inside each box and the margin that can go around them. In this tutorial we will learn how we can change the dimensions of boxes.

We have the following properties that allow you to control the dimensions of a box.

- The **height** property is used to set the height of a box.
- The **width** property is used to set the width of a box.
- The **line-height** property is used to set the height of a line of text.
- The **max-height** property is used to set a maximum height that a box can be.
- The **min-height** property is used to set the minimum height that a box can be.
- The **max-width** property is used to set the maximum width that a box can be.
- The **min-width** property is used to set the minimum width that a box can be.

The Height and Width Properties

The *height* and *width* properties allow you to set the height and width for boxes. They can take values of a length, a percentage, or the keyword auto.

```
<html>
  <head>
  </head>

  <body>
    <p style = "width:400px; height:100px; border:1px solid red; padding:5px; margin:10px;">
      This paragraph is 400pixels wide and 100 pixels high
    </p>
  </body>
</html>
```

The line-height Property

The *line-height* property allows you to increase the space between lines of text. The value of the line-height property can be a number, a length, or a percentage.

Here is an example −

```
<html>
  <head>
  </head>

  <body>
    <p style = "width:400px; height:100px; border:1px solid red; padding:5px; margin:10px; line-
height:30px;">
        This paragraph is 400pixels wide and 100 pixels high and here line height is 30pixels.
        This paragraph is 400 pixels wide and 100 pixels high and here line height is 30pixels.
    </p>
  </body>
</html>
```

The max-height Property

The *max-height* property allows you to specify maximum height of a box. The value of the max-height property can be a number, a length, or a percentage.

NOTE − This property does not work in either Netscape 7 or IE 6.

```
<html>
  <head>
  </head>
  <body>
    <p style = "width:400px; max-height:10px; border:1px solid red; padding:5px;
margin:10px;">
        This paragraph is 400px wide and max height is 10px
        This paragraph is 400px wide and max height is 10px
        This paragraph is 400px wide and max height is 10px
        This paragraph is 400px wide and max height is 10px
    </p>
    <br>
    <br>
    <br>
    <img alt = "logo" src = "/css/images/logo.png" width = "195" height = "84" />
  </body>
</html>
```

The min-height Property

The *min-height* property allows you to specify minimum height of a box. The value of the min-height property can be a number, a length, or a percentage.

NOTE − This property does not work in either Netscape 7 or IE 6.

```
<html>
  <head>
  </head>

  <body>
    <p style = "width:400px; min-height:200px; border:1px solid red; padding:5px;
margin:10px;">
```

```
      This paragraph is 400px wide and min height is 200px
      This paragraph is 400px wide and min height is 200px
      This paragraph is 400px wide and min height is 200px
      This paragraph is 400px wide and min height is 200px
    </p>
    <img alt = "logo" src = "/css/images/logo.png" width = "95" height = "84" />
  </body>
</html>
```

The max-width Property

The *max-width* property allows you to specify maximum width of a box. The value of the max-width property can be a number, a length, or a percentage.

NOTE − This property does not work in either Netscape 7 or IE 6.

```
<html>
  <head>
  </head>

  <body>
    <p style = "max-width:100px; height:200px; border:1px solid red; padding:5px; 
margin:10px;">
      This paragraph is 200px high and max width is 100px
      This paragraph is 200px high and max width is 100px
      This paragraph is 200px high and max width is 100px
      This paragraph is 200px high and max width is 100px
      This paragraph is 200px high and max width is 100px
    </p>
    <img alt = "logo" src = "/images/css.gif" width = "95" height = "84" />
  </body>
</html>
```

The min-width Property

The *min-width* property allows you to specify minimum width of a box. The value of the min-width property can be a number, a length, or a percentage.

NOTE − This property does not work in either Netscape 7 or IE 6.

```
<html>
  <head>
  </head>

  <body>
    <p style = "min-width:400px; height:100px; border:1px solid red; padding:5px; 
margin:10px;">
      This paragraph is 100px high and min width is 400px
      This paragraph is 100px high and min width is 400px
    </p>
    <img alt = "logo" src = "/css/images/css.gif" width = "95" height = "84" />
  </body>
```

```
</html>
```

CSS Layout – The position Property

The position property specifies the type of positioning method used for an element (static, relative, fixed, absolute or sticky).

The position Property

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

There are five different position values:

- static
- relative
- fixed
- absolute
- 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. They also work differently depending on the position value.

position: static;

HTML elements are positioned static by default.

Static positioned elements are not affected by the top, bottom, left, and right properties.

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;

Here is the CSS that is used:

Example

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

Try it Yourself »

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;
}
```

Try it Yourself »

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.

A fixed element does not leave a gap in the page where it would normally have been located.

Notice the fixed element in the lower-right corner of the page. Here is the CSS that is used:

Example

```
div.fixed {
  position: fixed;
  bottom: 0;
  right: 0;
  width: 300px;
  border: 3px solid #73AD21;
}
```

Try it Yourself »

This <div> element has position: fixed;

position: absolute;

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

However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

Note: Absolute positioned elements are removed from the normal flow, and can overlap elements.

Here is a simple example:

This <div> element has position: relative;

This <div> element has position: absolute;

Here is the CSS that is used:

Example

```
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;
}
```

Try it Yourself »

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).

Note: Internet Explorer does not support sticky positioning. Safari requires a -webkit- prefix (see example below). You must also specify at least one of top, right, bottom or left for sticky positioning to work.

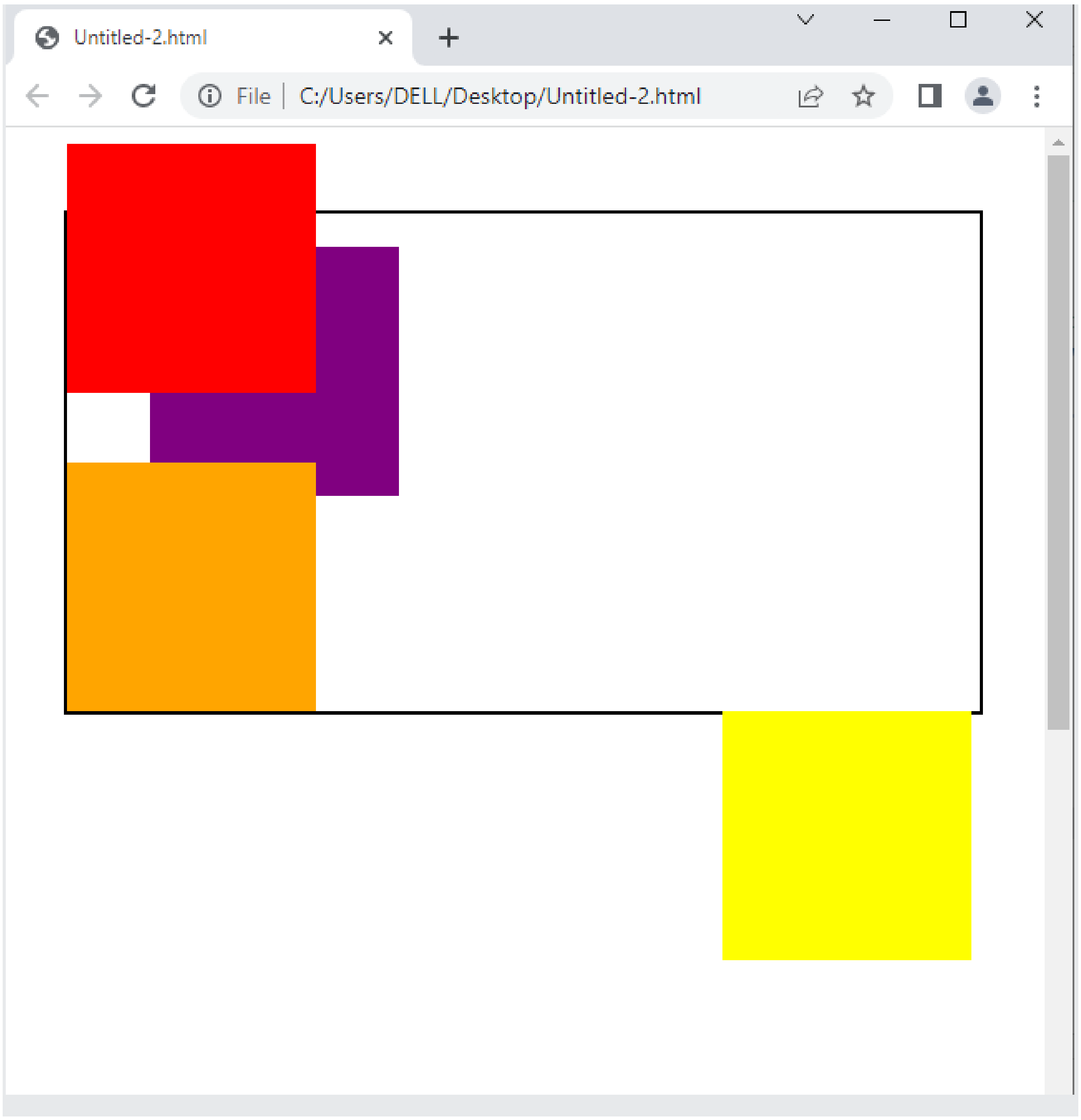
In this example, the sticky element sticks to the top of the page (top: 0), when you reach its scroll position.

Example

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

Try it Yourself »

```
<html>
  <head>
    <style>
      body{height: 200vh;}

      .container{width:550px;border:2px solid black;margin:50px auto;position:relative;}
      .box{width:150px;height: 150px;}
      .purple{background-color:purple;position:relative;left:50px;top:20px;}
      .yellow{background-color:yellow;position:absolute;right:5px;}
      .red{background-color:red;position:fixed;top:10px;}
      .orange{background-color:orange;position:sticky;top:30px;}
    </style>
  </head>

  <body>
<div class="container">
  <div class="box purple"></div>
  <div class="box orange"></div>
  <div class="box yellow"></div>
  <div class="box red"></div>

  </body>
</html>
```



CSS Layout - float and clear

The CSS float property specifies how an element should float.

The CSS clear property specifies what elements can float beside the cleared element and on which side.

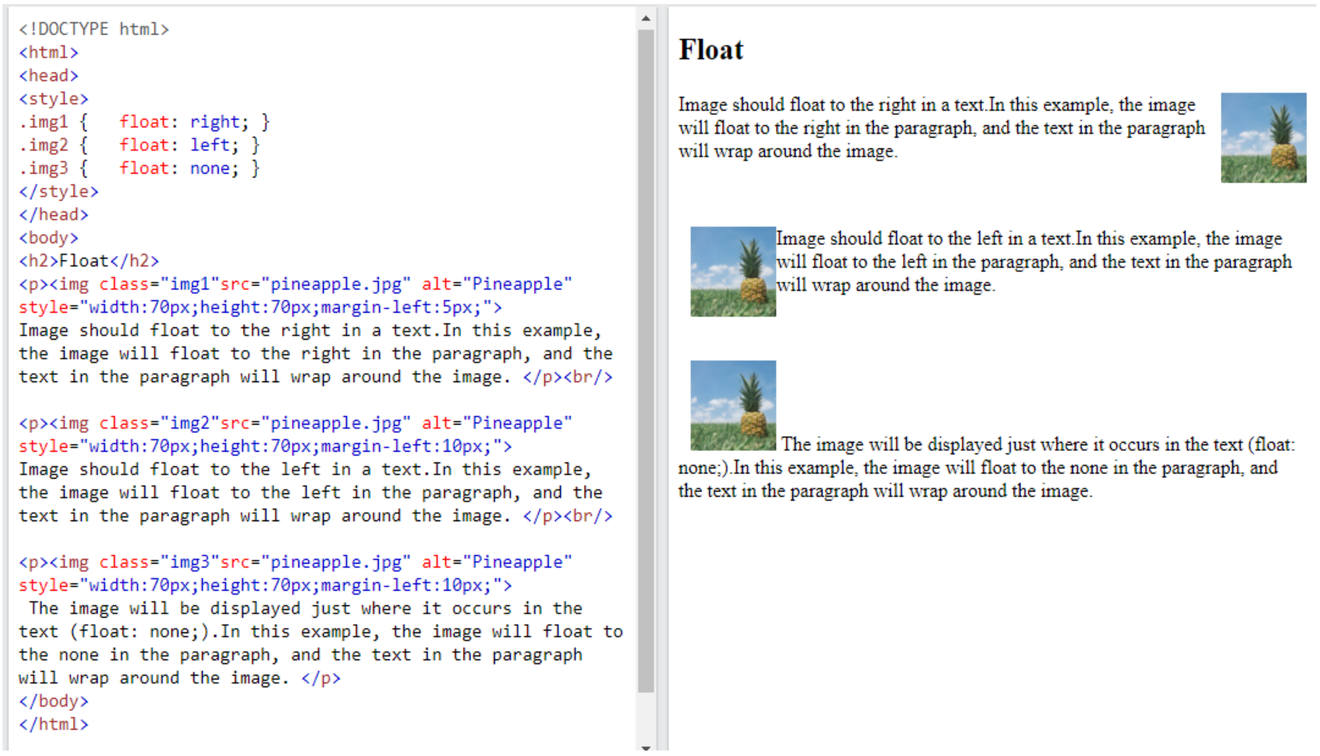
The float Property

The float property is used for positioning and formatting content e.g. let an image float left to the text in a container.

The float property can have one of the following values:

- 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

In its simplest use, the float property can be used to wrap text around images.



```
<!DOCTYPE html>
<html>
<head>
<style>
.img1 {   float: right; }
.img2 {   float: left; }
.img3 {   float: none; }
</style>
</head>
<body>
<h2>Float</h2>
<p><img class="img1"src="pineapple.jpg" alt="Pineapple"
style="width:70px;height:70px;margin-left:5px;">
Image should float to the right in a text.In this example,
the image will float to the right in the paragraph, and the
text in the paragraph will wrap around the image. </p><br/>

<p><img class="img2"src="pineapple.jpg" alt="Pineapple"
style="width:70px;height:70px;margin-left:10px;">
Image should float to the left in a text.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><br/>

<p><img class="img3"src="pineapple.jpg" alt="Pineapple"
style="width:70px;height:70px;margin-left:10px;">
 The image will be displayed just where it occurs in the
text (float: none;).In this example, the image will float to
the none in the paragraph, and the text in the paragraph
will wrap around the image. </p>
</body>
</html>
```

Float

Image should float to the right in a text.In this example, the image will float to the right in the paragraph, and the text in the paragraph will wrap around the image.

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

The image will be displayed just where it occurs in the text (float: none;).In this example, the image will float to the none in the paragraph, and the text in the paragraph will wrap around the image.

CSS Layout - clear

The clear Property

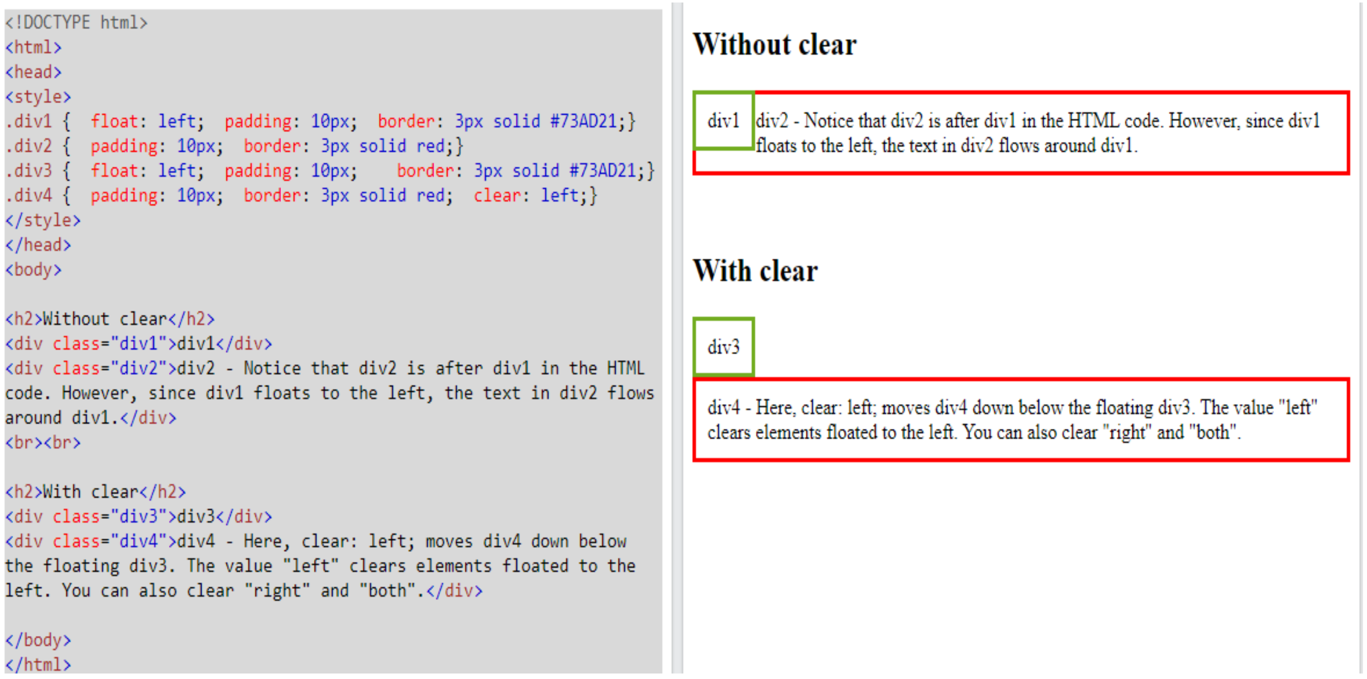
When we use the float property, and we want the next element below (not on right or left), we will have to use the clear property.

The clear property specifies what should happen with the element that is next to a floating element.

The clear property can have one of the following values:

- 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



```html
<!DOCTYPE html>
<html>
<head>
<style>
.div1 {  float: left;  padding: 10px;  border: 3px solid #73AD21;}
.div2 {  padding: 10px;  border: 3px solid red;}
.div3 {  float: left;  padding: 10px;    border: 3px solid #73AD21;}
.div4 {  padding: 10px;  border: 3px solid red;  clear: left;}
</style>
</head>
<body>

<h2>Without clear</h2>
<div class="div1">div1</div>
<div class="div2">div2 - Notice that div2 is after div1 in the HTML
code. However, since div1 floats to the left, the text in div2 flows
around div1.</div>
<br><br>

<h2>With clear</h2>
<div class="div3">div3</div>
<div class="div4">div4 - Here, clear: left; moves div4 down below
the floating div3. The value "left" clears elements floated to the
left. You can also clear "right" and "both".</div>

</body>
</html>
```

Without clear

div1 div2 - Notice that div2 is after div1 in the HTML code. However, since div1 floats to the left, the text in div2 flows around div1.

With clear

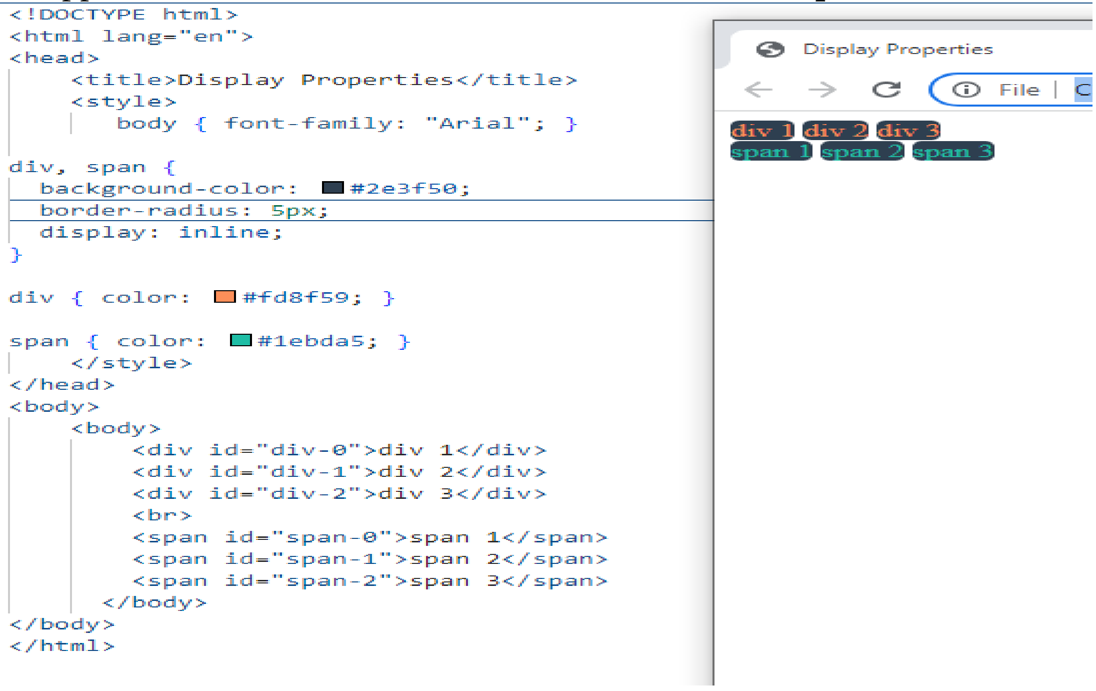
div3

div4 - Here, clear: left; moves div4 down below the floating div3. The value "left" clears elements floated to the left. You can also clear "right" and "both".

Display Property in CSS

CSS Display: inline

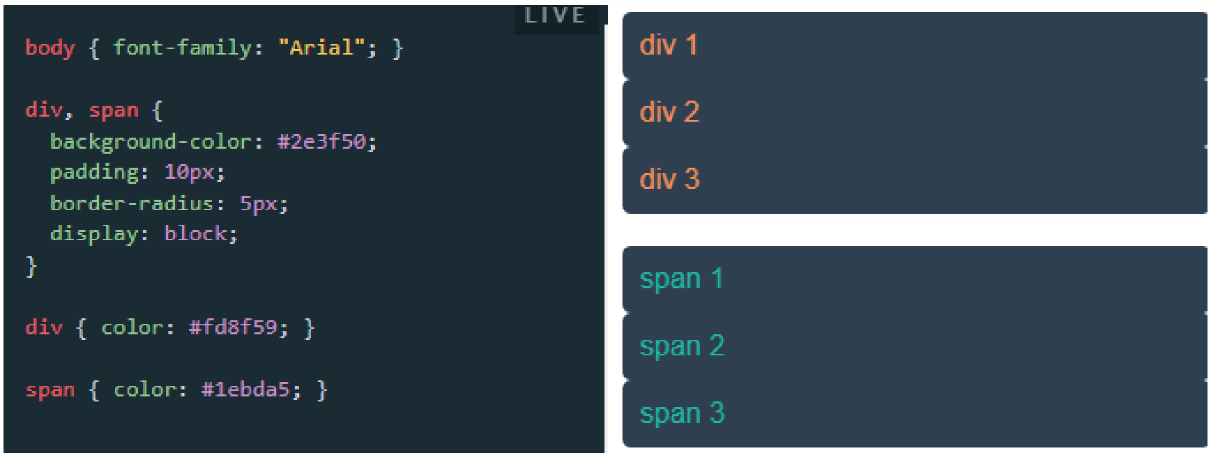
The **inline** display value turns any element into an inline element. These elements will appear on the same line without breaks, like **<span>** elements behave.



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Display Properties</title>
    <style>
      body { font-family: "Arial"; }

div, span {
  background-color: #2e3f50;
  border-radius: 5px;
  display: inline;
}

div { color: #fd8f59; }

span { color: #1ebda5; }
    </style>
</head>
<body>
    <body>
        <div id="div-0">div 1</div>
        <div id="div-1">div 2</div>
        <div id="div-2">div 3</div>
        <br>
        <span id="span-0">span 1</span>
        <span id="span-1">span 2</span>
        <span id="span-2">span 3</span>
      </body>
</body>
</html>
```

CSS Display: block

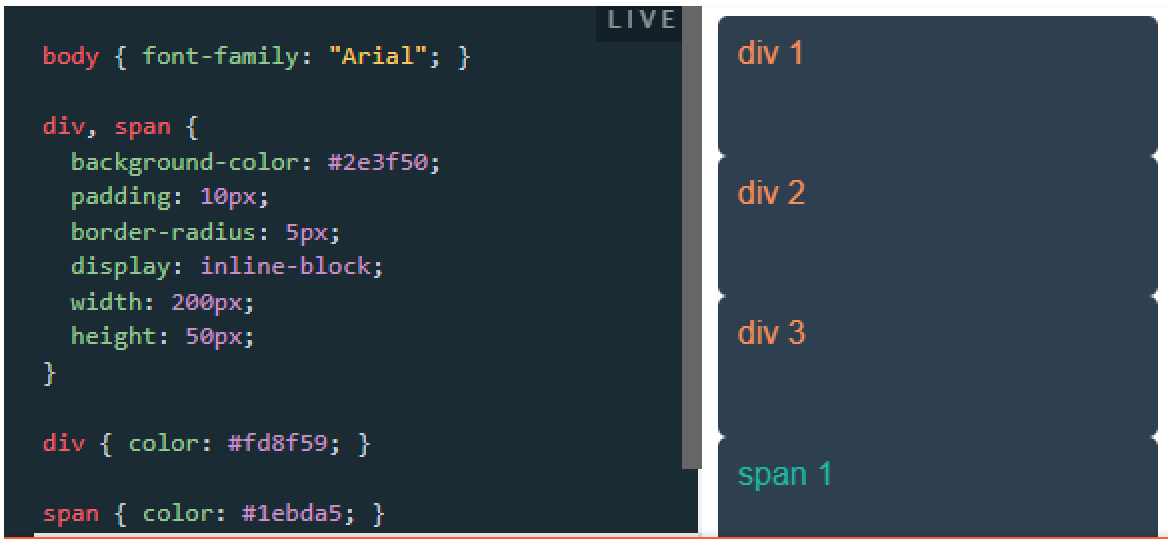
The **block** display value makes an element a block element. Block elements start a new line and span the entire width of the viewport by default, like how **<div>** elements behave. There are also line breaks before and after these elements.



```
body { font-family: "Arial"; }

div, span {
  background-color: #2e3f50;
  padding: 10px;
  border-radius: 5px;
  display: block;
}

div { color: #fd8f59; }

span { color: #1ebda5; }
```

CSS Display: inline-block

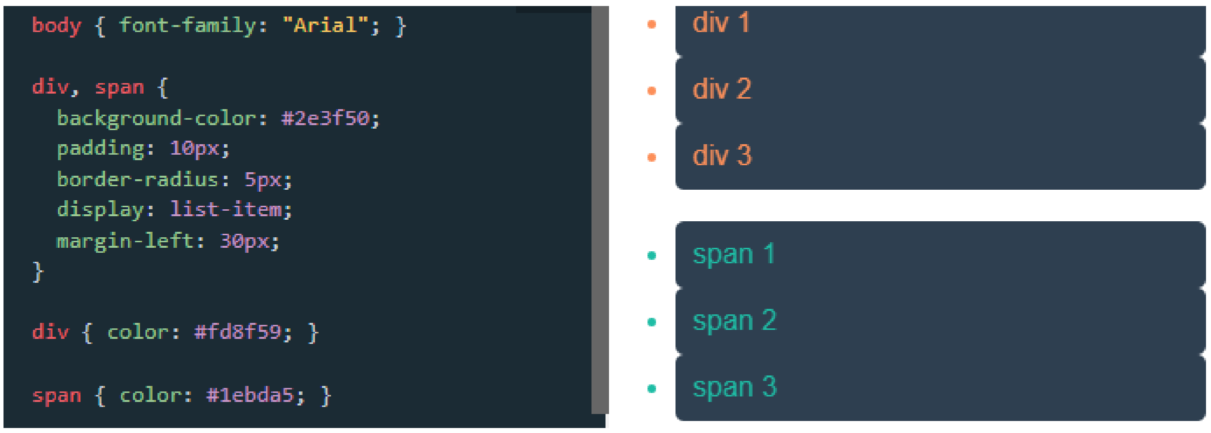
The **inline-block** value is a hybrid of **inline** and **block**. Elements assigned **display: inline-block** appear on the same line with other inline elements, a characteristic of inline elements. However, inline-block elements are also like block elements in that you can change their widths and heights with CSS.



```
body { font-family: "Arial"; }

div, span {
  background-color: #2e3f50;
  padding: 10px;
  border-radius: 5px;
  display: inline-block;
  width: 200px;
  height: 50px;
}

div { color: #fd8f59; }

span { color: #1ebda5; }
```

CSS Display: list-item

Elements assigned **display: list-item** behave like **<li>** elements. The entire element becomes a block-level element, the text inside becomes its own inline element, and a bullet point is added.



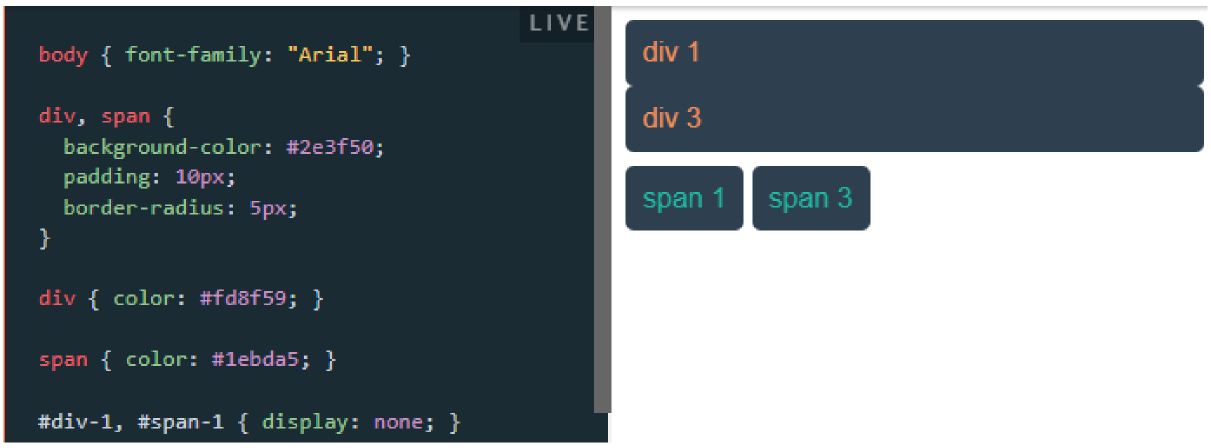
```
body { font-family: "Arial"; }

div, span {
  background-color: #2e3f50;
  padding: 10px;
  border-radius: 5px;
  display: list-item;
  margin-left: 30px;
}

div { color: #fd8f59; }

span { color: #1ebda5; }
```

- div 1
- div 2
- div 3

- span 1
- span 2
- span 3

CSS Display: none

display: none removes the targeted element (and all its child elements) from the page. This causes accompanying elements to behave as if this element does not exist. In this example, **display: none** is applied to the second **<div>** element and the second **<span>** element.



```
body { font-family: "Arial"; }

div, span {
  background-color: #2e3f50;
  padding: 10px;
  border-radius: 5px;
}

div { color: #fd8f59; }

span { color: #1ebda5; }

#div-1, #span-1 { display: none; }
```

div 1

div 3

span 1 span 3

CSS Layout - Horizontal & Vertical Align

Center Align Elements

To horizontally center a block element (like <div>), use margin: auto;

Setting the width of the element will prevent it from stretching out to the edges of its container.

The element will then take up the specified width, and the remaining space will be split equally between the two margins:

This div element is centered.

Example

```
.center {
  margin: auto;
  width: 50%;
  border: 3px solid green;
  padding: 10px;
}
```

Try it Yourself »

Note: Center aligning has no effect if the width property is not set (or set to 100%).

Center Align Text

To just center the text inside an element, use text-align: center;

This text is centered.

Example

```
.center {
  text-align: center;
  border: 3px solid green;
}
```

Try it Yourself »

Tip: For more examples on how to align text, see the CSS Text chapter.

Center an Image

To center an image, set left and right margin to auto and make it into a block element:

Example

```
img {
  display: block;
  margin-left: auto;
  margin-right: auto;
  width: 40%;
}
```

Try it Yourself »

Left and Right Align - Using position

One method for aligning elements is to use position: absolute;:

Example

```
.right {
  position: absolute;
  right: 0px;
  width: 300px;
  border: 3px solid #73AD21;
  padding: 10px;
}
```

Try it Yourself »

Note: Absolute positioned elements are removed from the normal flow, and can overlap elements.

Left and Right Align - Using float

Another method for aligning elements is to use the float property:

Example

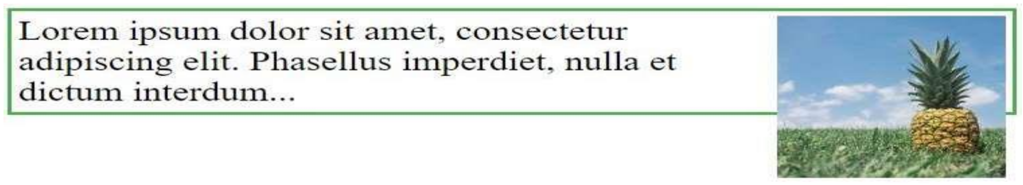
```
.right {
  float: right;
  width: 300px;
  border: 3px solid #73AD21;
  padding: 10px;
}
```

Try it Yourself »

The clearfix Hack

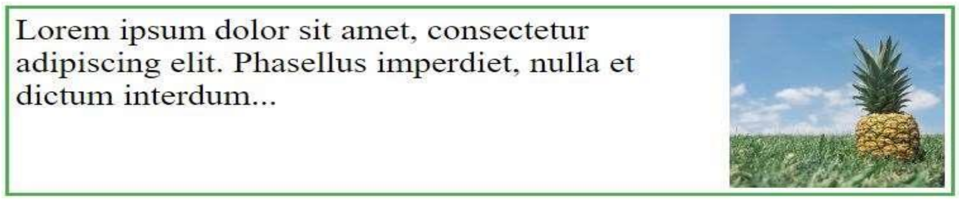
Note: If an element is taller than the element containing it, and it is floated, it will overflow outside of its container. You can use the "clearfix hack" to fix this (see example below).

Without Clearfix



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum...

With Clearfix


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum...

Then we can add the clearfix hack to the containing element to fix this problem:

Example

```
.clearfix::after {
  content: "";
  clear: both;
  display: table;
}
```

Try it Yourself »

Center Vertically - Using padding

There are many ways to center an element vertically in CSS. A simple solution is to use top and bottom padding:

I am vertically centered.

Example

```
.center {
  padding: 70px 0;
  border: 3px solid green;
}
```

Try it Yourself »

To center both vertically and horizontally, use padding and text-align: center:

I am vertically and horizontally centered.

Example

```
.center {
  padding: 70px 0;
  border: 3px solid green;
  text-align: center;
}
```

Try it Yourself »

Center Vertically - Using line-height

Another trick is to use the line-height property with a value that is equal to the height property:

I am vertically and horizontally centered.

Example

```
.center {
  line-height: 200px;
  height: 200px;
  border: 3px solid green;
  text-align: center;
}

/* If the text has multiple lines, add the following: */
.center p {
  line-height: 1.5;
  display: inline-block;
  vertical-align: middle;
}
```

Try it Yourself »

Center Vertically - Using position & transform

If padding and line-height are not options, another solution is to use positioning and the transform property:

I am vertically and horizontally centered.

Example

```
.center {
  height: 200px;
  position: relative;
  border: 3px solid green;
}

.center p {
  margin: 0;
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
}
```

Try it Yourself »

Tip: You will learn more about the transform property in our 2D Transforms Chapter.

Center Vertically - Using Flexbox

You can also use flexbox to center things. Just note that flexbox is not supported in IE10 and earlier versions:

I am vertically and horizontally centered.

Example

```
.center {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 200px;
  border: 3px solid green;
}
```

Try it Yourself »

CSS Pseudo-classes

What are 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;
}
```

Anchor Pseudo-classes

Links can be displayed in different ways:

Example

```
/* unvisited link */
a:link {
  color: #FF0000;
}

/* visited link */
a:visited {
  color: #00FF00;
}

/* mouse over link */
a:hover {
  color: #FF00FF;
}

/* selected link */
a:active {
  color: #0000FF;
}
```

Try it Yourself »

Note: 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.

Pseudo-classes and HTML Classes

Pseudo-classes can be combined with HTML classes:

When you hover over the link in the example, it will change color:

Example

```
a.highlight:hover {
  color: #ff0000;
}
```

Try it Yourself »

Hover on <div>

An example of using the :hover pseudo-class on a <div> element:

Example

```
div:hover {
  background-color: blue;
}
```

Try it Yourself »

Simple Tooltip Hover

Hover over a <div> element to show a <p> element (like a tooltip):

Hover over me to show the <p> element.

Example

```
p {
  display: none;
  background-color: yellow;
  padding: 20px;
}

div:hover p {
  display: block;
}
```

Try it Yourself »

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:

Example

```
p:first-child {
  color: blue;
}
```

Try it Yourself »

Match the first <i> element in all <p> elements

In the following example, the selector matches the first <i> element in all <p> elements:

Example

```
p i:first-child {
  color: blue;
}
```

Try it Yourself »

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:

Example

```
p:first-child i {
  color: blue;
}
```

Try it Yourself »

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":

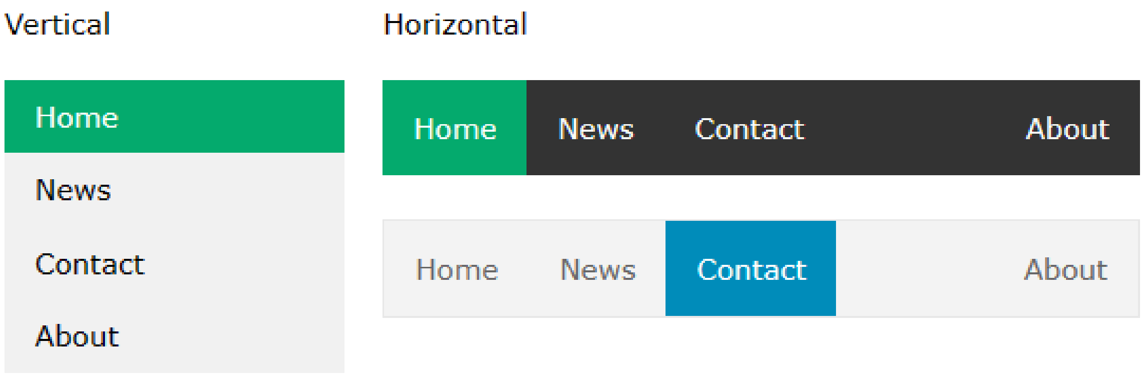
Example

```
<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>

</body>
</html>
```

Try it Yourself »

CSS Navigation Bar



Navigation Bars

Having easy-to-use navigation is important for any web site.

With CSS you can transform boring HTML menus into good-looking navigation bars.

Navigation Bar = List of Links

A navigation bar needs standard HTML as a base.

In our examples we will build the navigation bar from a standard HTML list.

A navigation bar is basically a list of links, so using the <ul> and <li> elements makes perfect sense:

Example

```
<ul>
  <li><a href="default.asp">Home</a></li>
  <li><a href="news.asp">News</a></li>
  <li><a href="contact.asp">Contact</a></li>
  <li><a href="about.asp">About</a></li>
</ul>
```

Try it Yourself »

Now let's remove the bullets and the margins and padding from the list:

Example

```
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
}
```

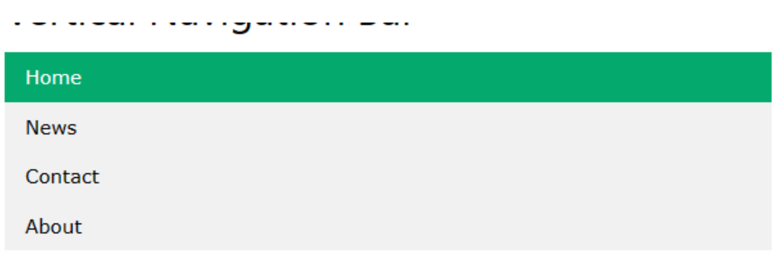
Try it Yourself »

Example explained:

- list-style-type: none; - Removes the bullets. A navigation bar does not need list markers
- Set margin: 0; and padding: 0; to remove browser default settings

The code in the example above is the standard code used in both vertical, and horizontal navigation bars, which you will learn more about in the next chapters.

Vertical Navigation Bar



To build a vertical navigation bar, you can style the <a> elements inside the list, in addition to the code from the previous page:

Example

```
li a {
  display: block;
  width: 60px;
}
```

Try it Yourself »

Example explained:

- display: block; - Displaying the links as block elements makes the whole link area clickable (not just the text), and it allows us to specify the width (and padding, margin, height, etc. if you want)
- width: 60px; - Block elements take up the full width available by default. We want to specify a 60 pixels width

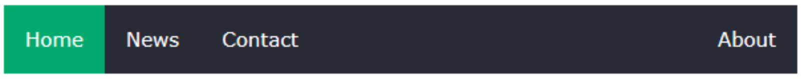
You can also set the width of <ul>, and remove the width of <a>, as they will take up the full width available when displayed as block elements. This will produce the same result as our previous example:

Example

```
ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  width: 60px;
}

li a {
  display: block;
}
```

Try it Yourself »

Horizontal Navigation Bar



There are two ways to create a horizontal navigation bar. Using **inline** or **floating** list items.

Inline List Items

One way to build a horizontal navigation bar is to specify the <li> elements as inline, in addition to the "standard" code from the previous page:

Example

```
li {
  display: inline;
}
```

Try it Yourself »

Example explained:

- display: inline; - By default, <li> elements are block elements. Here, we remove the line breaks before and after each list item, to display them on one line

Floating List Items

Another way of creating a horizontal navigation bar is to float the <li> elements, and specify a layout for the navigation links:

Example

```
li {
  float: left;
}

a {
  display: block;
  padding: 8px;
  background-color: #dddddd;
}
```

Try it Yourself »

Example explained:

- float: left; - Use float to get block elements to float next to each other
- display: block; - Allows us to specify padding (and height, width, margins, etc. if you want)
- padding: 8px; - Specify some padding between each <a> element, to make them look good
- background-color: #dddddd; - Add a gray background-color to each <a> element

Tip: Add the background-color to <ul> instead of each <a> element if you want a full-width background color:

Example

```
ul {
  background-color: #dddddd;
}
```

Try it Yourself »

CSS - Using Images

Images play an important role in any webpage. Though it is not recommended to include a lot of images, but it is still important to use good images wherever required.

CSS plays a good role to control image display. You can set the following image properties using CSS.

- The **border** property is used to set the width of an image border.
- The **height** property is used to set the height of an image.
- The **width** property is used to set the width of an image.
- The **-moz-opacity** property is used to set the opacity of an image.

The Image Border Property

The *border* property of an image is used to set the width of an image border. This property can have a value in length or in %.

A width of zero pixels means no border.

```
<html>
  <head>
  </head>

  <body>
    <img style = "border:0px;" src = "/css/images/logo.png" />
    <br />
    <img style = "border:3px dashed red;" src = "/css/images/logo.png" />
  </body>
</html>
```

The Image Height Property

The *height* property of an image is used to set the height of an image. This property can have a value in length or in %. While giving value in %, it applies it in respect of the box in which an image is available.

```
<html>
  <head>
  </head>

  <body>
    <img style = "border:1px solid red; height:100px;" src = "/css/images/logo.png" />
    <br />
    <img style = "border:1px solid red; height:50%;" src = "/css/images/logo.png" />
  </body>
</html>
```

The Image Width Property

The *width* property of an image is used to set the width of an image. This property can have a value in length or in %. While giving value in %, it applies it in respect of the box in which an image is available.

```
<html>
  <head>
  </head>

  <body>
    <img style = "border:1px solid red; width:150px;" src = "/css/images/logo.png" />
    <br />
    <img style = "border:1px solid red; width:100%;" src = "/css/images/logo.png" />
  </body>
</html>
```

The -moz-opacity Property

The *-moz-opacity* property of an image is used to set the opacity of an image. This property is used to create a transparent image in Mozilla. IE uses **filter:alpha(opacity=x)** to create transparent images.

In Mozilla (-moz-opacity:x) x can be a value from 0.0 - 1.0. A lower value makes the element more transparent (The same things goes for the CSS3-valid syntax opacity:x).

In IE (filter:alpha(opacity=x)) x can be a value from 0 - 100. A lower value makes the element more transparent.

```
<html>
  <head>
  </head>

  <body>
    <img style = "border:1px solid red; -moz-opacity:0.4; filter:alpha(opacity=40);" src =
"/css/images/logo.png" />
  </body>
</html>
```

<style>

body {

  background-color:white;

}

img {

  width: 20%;

  height: auto;

  float: left;

  max-width: 135px;

```
}

.blur {filter: blur(4px);}

.brightness {filter: brightness(250%);}

.contrast {filter: contrast(180%);}

.grayscale {filter: grayscale(100%);}

.huerotate {filter: hue-rotate(180deg);}

.invert {filter: invert(100%);}

.opacity {filter: opacity(50%);}

.saturate {filter: saturate(7);}

.sepia {filter: sepia(100%);}

.shadow {filter: drop-shadow(8px 8px 10px green);}

.moz{-moz-opacity:0.1;filter:alpha(opacity=100);}

</style>

</head>

<body>

<h2>Image Filters</h2>

<p><strong>Note:</strong> The filter property is not supported in Internet Explorer or Edge
12.</p>

<img src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="blur" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="brightness" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="contrast" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="grayscale" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="huerotate" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="invert" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="opacity" src="pineapple.jpg" alt="Pineapple" width="300" height="300">
```

```
<img class="saturate" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="sepia" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="shadow" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

<img class="moz" src="pineapple.jpg" alt="Pineapple" width="300" height="300">

</body>

</html>
```

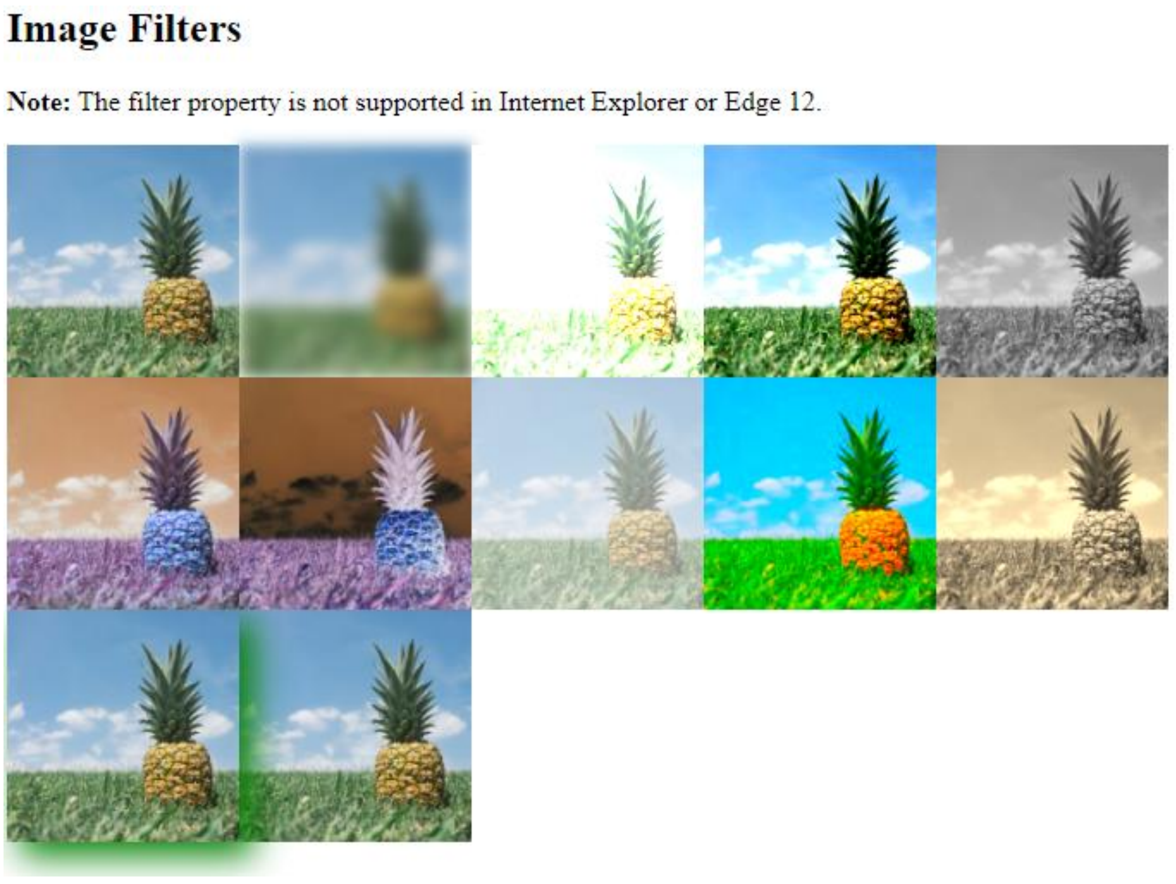


Image Filters

Note: The filter property is not supported in Internet Explorer or Edge 12.

CSS Colors

RGBA Colors

RGBA color values are an extension of RGB color values with an alpha channel - which specifies the opacity for a color.

An RGBA color value is specified with: rgba(red, green, blue, alpha). The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque).

```
rgba(255, 0, 0, 0.2);
```

rgba(255, 0, 0, 0.4);

rgba(255, 0, 0, 0.6);

rgba(255, 0, 0, 0.8);

The following example defines different RGBA colors:

Example

```
#p1 {background-color: rgba(255, 0, 0, 0.3);}  /* red with opacity */
#p2 {background-color: rgba(0, 255, 0, 0.3);}  /* green with opacity */
#p3 {background-color: rgba(0, 0, 255, 0.3);}  /* blue with opacity */
```

Try it Yourself »

HSL Colors

HSL stands for Hue, Saturation and Lightness.

An HSL color value is specified with: hsl(hue, saturation, lightness).

1. Hue is a degree on the color wheel (from 0 to 360):
   - 0 (or 360) is red
   - 120 is green
   - 240 is blue
2. Saturation is a percentage value: 100% is the full color.
3. Lightness is also a percentage; 0% is dark (black) and 100% is white.

hsl(0, 100%, 30%);

hsl(0, 100%, 50%);

hsl(0, 100%, 70%);

hsl(0, 100%, 90%);

The following example defines different HSL colors:

Example

```
#p1 {background-color: hsl(120, 100%, 50%);}  /* green */
#p2 {background-color: hsl(120, 100%, 75%);}  /* light green */
#p3 {background-color: hsl(120, 100%, 25%);}  /* dark green */
#p4 {background-color: hsl(120, 60%, 70%);}   /* pastel green */
```

Try it Yourself »

HSLA Colors

HSLA color values are an extension of HSL color values with an alpha channel - which specifies the opacity for a color.

An HSLA color value is specified with: hsla(hue, saturation, lightness, alpha), where the alpha parameter defines the opacity. The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque).

hsla(0, 100%, 30%, 0.3);

hsla(0, 100%, 50%, 0.3);

hsla(0, 100%, 70%, 0.3);

hsla(0, 100%, 90%, 0.3);

The following example defines different HSLA colors:

Example

```
#p1 {background-color: hsla(120, 100%, 50%, 0.3);}  /* green with opacity */
#p2 {background-color: hsla(120, 100%, 75%, 0.3);}  /* light green with opacity */
#p3 {background-color: hsla(120, 100%, 25%, 0.3);}  /* dark green with opacity */
#p4 {background-color: hsla(120, 60%, 70%, 0.3);}   /* pastel green with opacity */
```

Try it Yourself »

Opacity

The CSS opacity property sets the opacity for the whole element (both background color and text will be opaque/transparent).

The opacity property value must be a number between 0.0 (fully transparent) and 1.0 (fully opaque).

rgb(255, 0, 0);opacity:0.2;

rgb(255, 0, 0);opacity:0.4;

rgb(255, 0, 0);opacity:0.6;

rgb(255, 0, 0);opacity:0.8;

Notice that the text above will also be transparent/opaque!

The following example shows different elements with opacity:

Example

```
#p1 {background-color:rgb(255,0,0);opacity:0.6;}  /* red with opacity */
#p2 {background-color:rgb(0,255,0);opacity:0.6;}  /* green with opacity */
#p3 {background-color:rgb(0,0,255);opacity:0.6;}  /* blue with opacity */
```

Try it Yourself »

CSS Image Sprites

Image Sprites

An image sprite is a collection of images put into a single image.

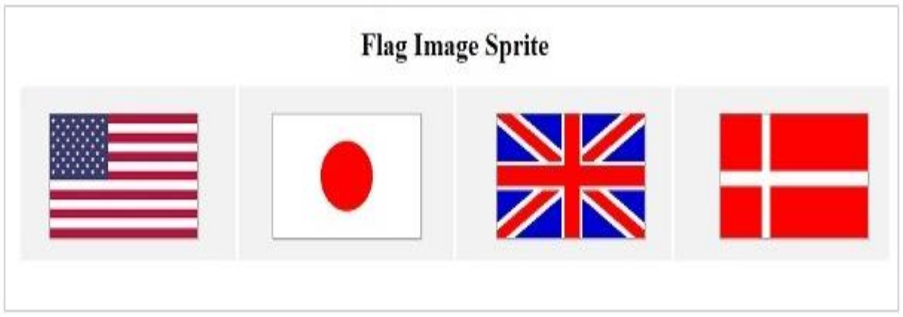
A web page with many images can take a long time to load and generates multiple server requests.

Using image sprites will reduce the number of server requests and save bandwidth.

Example

```
<!DOCTYPE html>
<html>
<head>
<title></title>
<style type="text/css">
.sprite {
  background: url("Capture.png") no-repeat;
  width: 280px;
  height: 200px;
  display: inline-block;
}
.flag1 {
  background-position: 0px 0px;
}
.flag2 {
  background-position: -255px 0px;
}
.flag3 {
  background-position: -510px 0px;
}
.flag4 {
  background-position: -765px 0px;
}
body {
  text-align: center;
}
</style>
</head>
<body>
<div><h1>Flag Image Sprite</h1></div>
<div class="sprite flag1"></div>
<div class="sprite flag2"></div>
<div class="sprite flag3"></div>
<div class="sprite flag4"></div>
</body>
</html>
```

Output



Example

```
<!DOCTYPE html>

<html lang="en">

<head>

<style>

 ul.menu {  list-style-type: none;    }

 ul.menu li {padding: 5px;font-size:16px;font-family:"Trebuchet MS",Arial, sans-serif;}

 ul.menu li a {height: 50px;line-height: 50px;display: inline-block;padding-left: 60px;  color:
red;background: url("/examples/images/mySprite.png") no-repeat; }

    ul.menu li.firefox a {background-position: 0 0;}

    ul.menu li.chrome a {background-position: 0 -100px;}
    ul.menu li.ie a {background-position: 0 -200px;}

    ul.menu li.safari a {background-position: 0 -300px;}

    ul.menu li.opera a {background-position: 0 -400px;}

</style></head><body>

            <ul class="menu">

        <li class="firefox"><a href="#">Firefox</a></li>

        <li class="chrome"><a href="#">Chrome</a></li>

        <li class="ie"><a href="#">Explorer</a></li>

        <li class="opera"><a href="#">Opera</a></li>

        <li class="safari"><a href="#">Safari</a></li>

    </ul>

</body></html>
```

Output

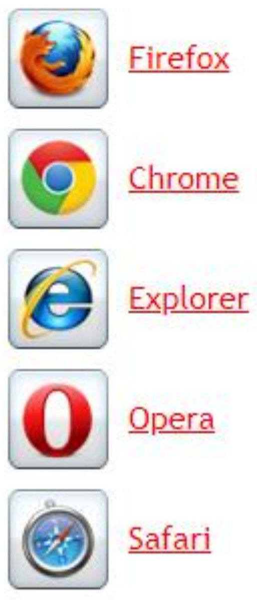
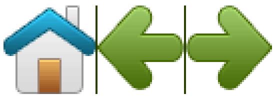


Image Sprites - Simple Example

Instead of using three separate images, we use this single image ("img_navsprites.gif"):



With CSS, we can show just the part of the image we need.

In the following example the CSS specifies which part of the "img_navsprites.gif" image to show:

Example

```
#home {
  width: 46px;
  height: 44px;
  background: url(img_navsprites.gif) 0 0;
}
```

Try it Yourself »

Example explained:

- <img id="home" src="img_trans.gif"> - Only defines a small transparent image because the src attribute cannot be empty. The displayed image will be the background image we specify in CSS
- width: 46px; height: 44px; - Defines the portion of the image we want to use
- background: url(img_navsprites.gif) 0 0; - Defines the background image and its position (left 0px, top 0px)

This is the easiest way to use image sprites, now we want to expand it by using links and hover effects.

Image Sprites - Create a Navigation List

We want to use the sprite image ("img_navsprites.gif") to create a navigation list.

We will use an HTML list, because it can be a link and also supports a background image:

Example

```
#navlist {
  position: relative;
}
```

```
#navlist li {
  margin: 0;
  padding: 0;
  list-style: none;
  position: absolute;
  top: 0;
}

#navlist li, #navlist a {
  height: 44px;
  display: block;
}

#home {
  left: 0px;
  width: 46px;
  background: url('img_navsprites.gif') 0 0;
}

#prev {
  left: 63px;
  width: 43px;
  background: url('img_navsprites.gif') -47px 0;
}

#next {
  left: 129px;
  width: 43px;
  background: url('img_navsprites.gif') -91px 0;
}
```

Try it Yourself »

Example explained:

- #navlist {position:relative;} - position is set to relative to allow absolute positioning inside it
- #navlist li {margin:0;padding:0;list-style:none;position:absolute;top:0;} - margin and padding are set to 0, list-style is removed, and all list items are absolute positioned
- #navlist li, #navlist a {height:44px;display:block;} - the height of all the images is 44px

Now start to position and style for each specific part:

- #home {left:0px;width:46px;} - Positioned all the way to the left, and the width of the image is 46px
- #home {background:url(img_navsprites.gif) 0 0;} - Defines the background image and its position (left 0px, top 0px)

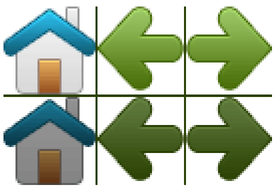
- #prev {left:63px;width:43px;} - Positioned 63px to the right (#home width 46px + some extra space between items), and the width is 43px
- #prev {background:url('img_navsprites.gif') -47px 0;} - Defines the background image 47px to the right (#home width 46px + 1px line divider)
- #next {left:129px;width:43px;} - Positioned 129px to the right (start of #prev is 63px + #prev width 43px + extra space), and the width is 43px
- #next {background:url('img_navsprites.gif') -91px 0;} - Defines the background image 91px to the right (#home width 46px + 1px line divider + #prev width 43px + 1px line divider)

Image Sprites - Hover Effect

Now we want to add a hover effect to our navigation list.

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

Our new image ("img_navsprites_hover.gif") contains three navigation images and three images to use for hover effects:



Because this is one single image, and not six separate files, there will be **no loading delay** when a user hovers over the image.

We only add three lines of code to add the hover effect:

Example

```
#home a:hover {
  background: url('img_navsprites_hover.gif') 0 -45px;
}

#prev a:hover {
  background: url('img_navsprites_hover.gif') -47px -45px;
}

#next a:hover {
  background: url('img_navsprites_hover.gif') -91px -45px;
}
```

Try it Yourself »

Example explained:

- #home a:hover {background: url('img_navsprites_hover.gif') 0 -45px;} - For all three hover images we specify the same background position, only 45px further down

CSS Attribute Selectors

Style HTML Elements With Specific Attributes

It is possible to style HTML elements that have specific attributes or attribute values.

CSS [attribute] Selector

The `[attribute]` selector is used to select elements with a specified attribute.

The following example selects all <a> elements with a target attribute:

Example

```
a[target] {
  background-color: yellow;
}
```

Try it Yourself »

CSS [attribute="value"] Selector

The `[attribute="value"]` selector is used to select elements with a specified attribute and value.

The following example selects all <a> elements with a target="_blank" attribute:

Example

```
a[target="_blank"] {
  background-color: yellow;
}
```

Try it Yourself »

CSS [attribute~="value"] Selector

The `[attribute~="value"]` selector is used to select elements with an attribute value containing a specified word.

The following example selects all elements with a title attribute that contains a space-separated list of words, one of which is "flower":

Example

```
[title~="flower"] {
  border: 5px solid yellow;
}
```

Try it Yourself »

The example above will match elements with title="flower", title="summer flower", and title="flower new", but not title="my-flower" or title="flowers".

CSS [attribute|="value"] Selector

The `[attribute|="value"]` selector is used to select elements with the specified attribute, whose value can be exactly the specified value, or the specified value followed by a hyphen (-).

Note: The value has to be a whole word, either alone, like class="top", or followed by a hyphen(-), like class="top-text".

Example

```
[class|="top"] {
  background: yellow;
}
```

Try it Yourself »

CSS [attribute^="value"] Selector

The `[attribute^="value"]` selector is used to select elements with the specified attribute, whose value starts with the specified value.

The following example selects all elements with a class attribute value that starts with "top":

Note: The value does not have to be a whole word!

Example

```
[class^="top"] {
  background: yellow;
}
```

Try it Yourself »

CSS [attribute$="value"] Selector

The `[attribute$="value"]` selector is used to select elements whose attribute value ends with a specified value.

The following example selects all elements with a class attribute value that ends with "test":

Note: The value does not have to be a whole word!

Example

```
[class$="test"] {
  background: yellow;
}
```

Try it Yourself »

CSS [attribute*="value"] Selector

The `[attribute*="value"]` selector is used to select elements whose attribute value contains a specified value.

The following example selects all elements with a class attribute value that contains "te":

Note: The value does not have to be a whole word!

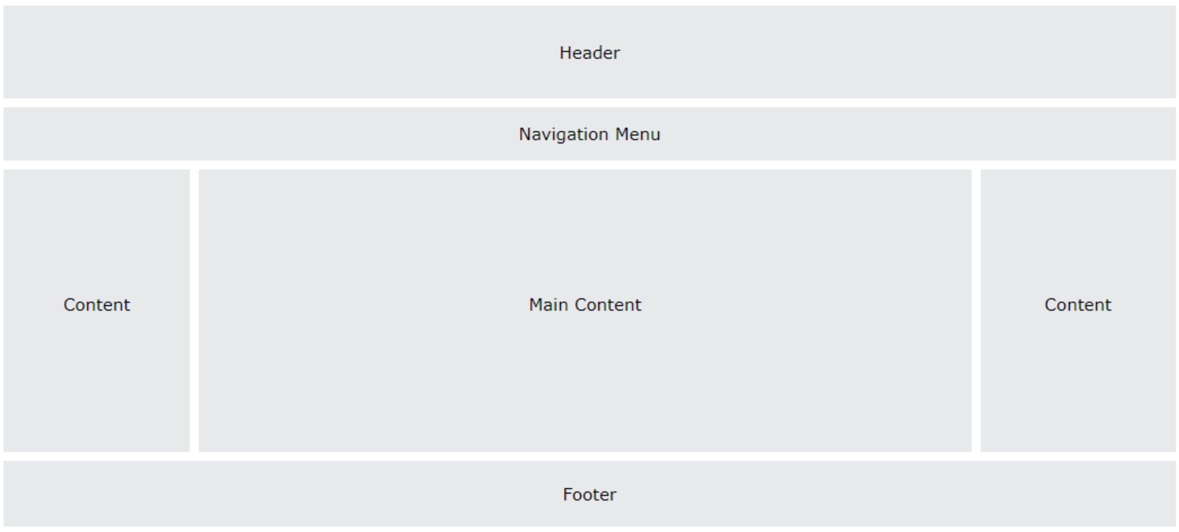
Example

```
[class*="te"] {
  background: yellow;
}
```

Try it Yourself »

Creating page Layout and Site Design

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



Header section contains a website logo, a search bar and profile of user. The navigation menu contains link to various categories of articles available and content section is divided into 3 parts(columns) with left and right sidebar containing links to other articles and advertisements whereas the main content section is the one containing this article, then at the bottom there is a footer section which contains address, links, contacts etc

CSS media queries

The **Media query** in CSS is used to create a responsive web design. It means that the view of a web page differs from system to system based on screen or media types. The breakpoint specifies for what device-width size, the content is just starting to break or deform.

Media queries can be used to check many things:

- width and height of the viewport
- width and height of the device
- Orientation
- Resolution

A media query consist of a media type that can contain one or more expression which can be either true or false. The result of the query is true if the specified media matches the type of device the document is displayed on. If the media query is true then a style sheet is applied.

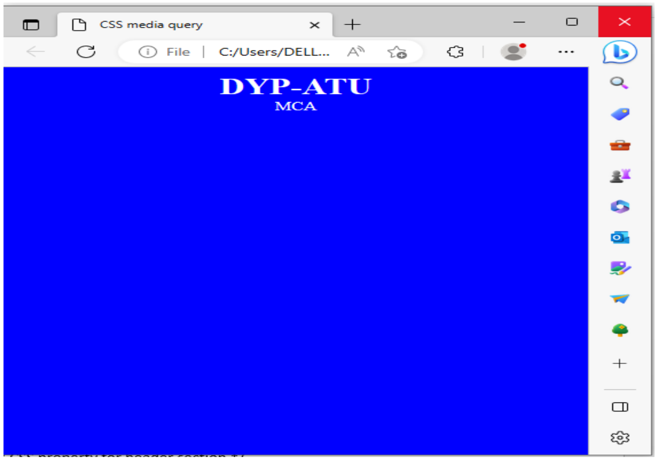
Media Types in CSS: There are many types of media types which are listed below:

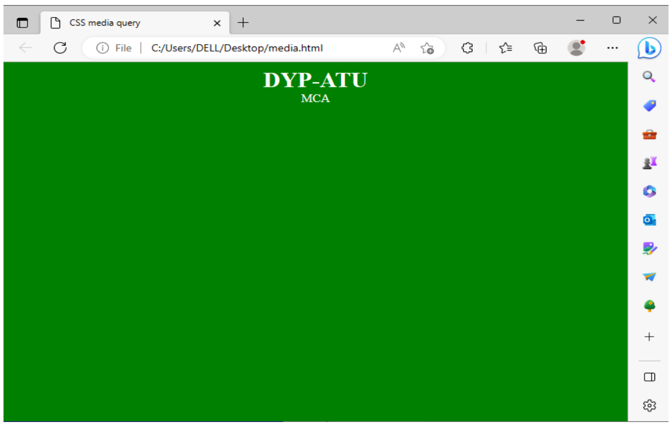
- **all:** It is used for all media devices
- **print:** It is used for printer.
- **screen:** It is used for computer screens, smartphones, etc.
- **speech:** It is used for screen readers that read the screen aloud.

Example-

```
<!DOCTYPE html>
<html>
 <head>
    <title>CSS media query</title>
    <style>
    body {
        text-align: center;
    }
         .dyp {
        font-size: 40px;
        font-weight: bold;
        color: green;
    }
         @media screen and (max-width:1200px) {
        body {
            text-align: center;
            background-color: green;
        }
        .dyp {
            font-size: 30px;
            font-weight: bold;
            color: white;
        }
        .a {
            color: white;
        }
    }

    @media screen and (max-width:700px) {
        body {
            text-align: center;
            background-color: blue;
        }
    }
    </style>
</head>

<body>
    <div class="dyp">DYP-ATU</div>
    <div class="a">MCA</div>
</body>

</html>
```

Output: From the output, we can see that if the max-width of the screen is reduced to 1200px then the background color changes to green & if the max-width of the screen is reduced to 700px then the background color will turn to blue. For the desktop size width, the background color will be white.

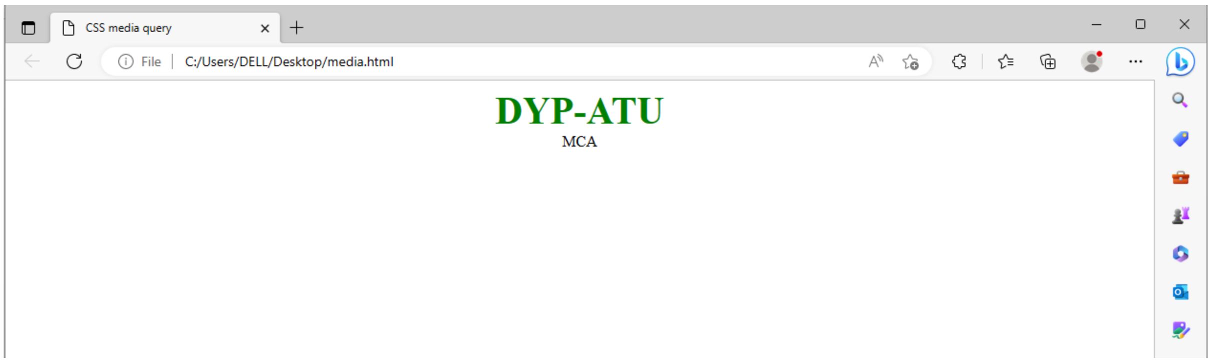



CSS media query

C:/Users/DELL/Desktop/media.html

DYP-ATU

MCA


CSS media query

C:/Users/DELL/Desktop/media.html

DYP-ATU

MCA

Example-Website design and layout

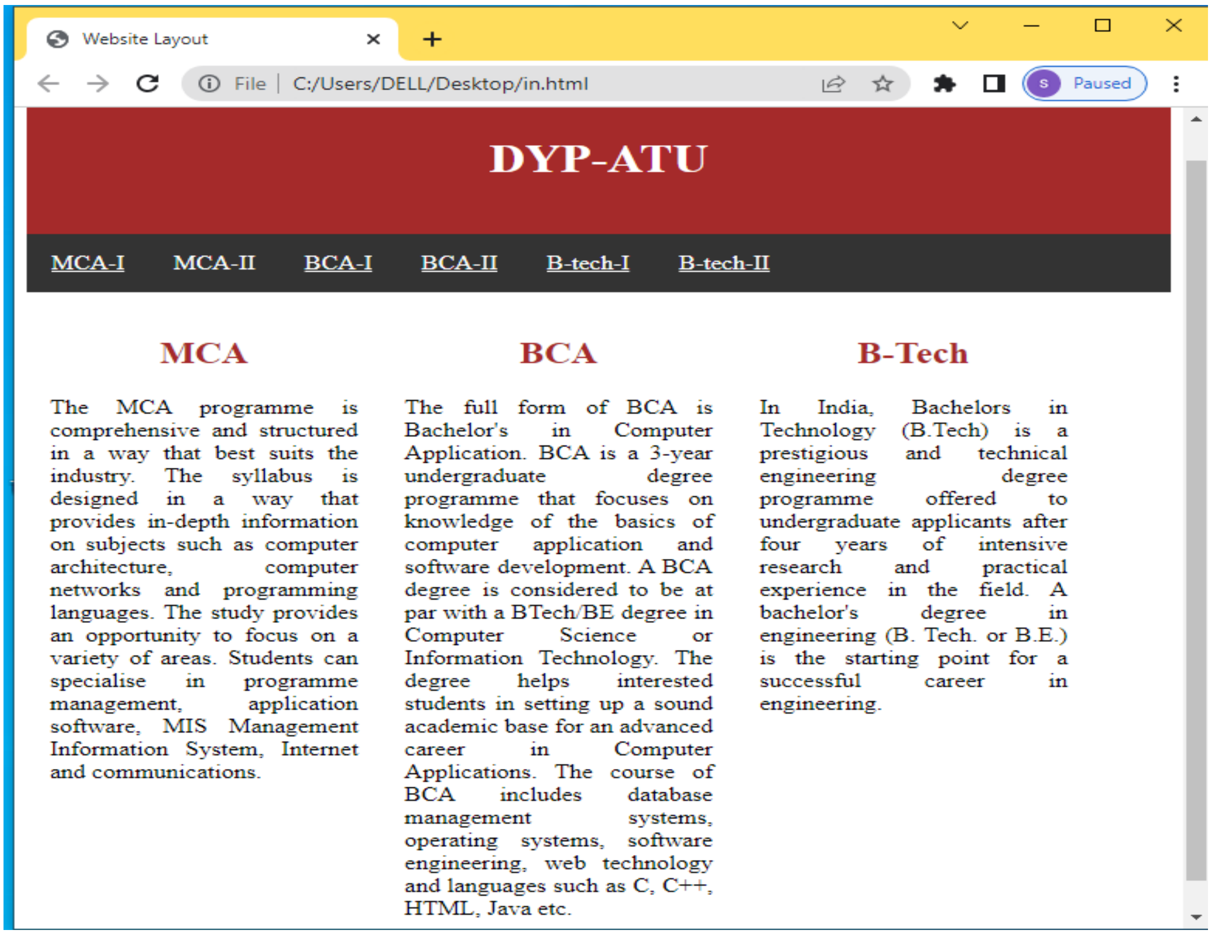
```
<!DOCTYPE html>

<html>

        <head>

                <title>

                        Website Layout

                </title>

                <style>

                        * {

                                box-sizing: border-box;

                        }


                        /* CSS property for header section */

                        .header {

                                background-color: brown;

                                padding: 15px;

                                text-align: center;
```

```
                                    }

                                    /* CSS property for navigation menu */
                                    .nav_menu {
                                          overflow: hidden;
                                          background-color: #333;
                                    }
                                    .nav_menu a {
                                          float: left;
                                          display: block;
                                          color: white;
                                          text-align: center;
                                          padding: 14px 16px;
                                          text-decoration: underline;
                                    }
                                    .nav_menu a:hover {
                                          background-color: white;
                                          color: green;
                                    }
/* The subnavigation menu */
.subnav {
  float: left;
  overflow: hidden;
}
/* Subnav button */
.subnav .subnavbtn {
  font-size: 16px;
  border: none;
  outline: none;
  color: white;
  padding: 14px 16px;
  background-color: inherit;
  font-family: inherit;
  margin: 0;
}
/* Add a red background color to navigation links on hover */
.navbar a:hover, .subnav:hover .subnavbtn {
  background-color: red;
}
/* Style the subnav content - positioned absolute */
.subnav-content {
  display: none;
  position: absolute;
  left: 0;
  background-color: skyblue;
  width: 100%;
  z-index: 1;
}
/* Style the subnav links */
.subnav-content a {
  float: left;
  color: white;
  text-decoration: none;
}
/* Add a grey background color on hover */
.subnav-content a:hover {
  background-color: #eee;
  color: black;
}
/* When you move the mouse over the subnav container, open the subnav content */
.subnav:hover .subnav-content {
  display: block;
```

```
}

                    /* CSS property for content section */
                    .columnA, .columnB, .columnC {
                            float: left;
                            width: 31%;
                            padding: 15px;
                            text-align:justify;
                    }
                    h2 {
                            color:brown;
                            text-align:center;
                    }

                    /* Media query to set website layout
                    according to screen size */
                    @media screen and (max-width:600px) {
                            .columnA, .columnB, .columnC {
                                    width: 50%;
                            }
                    }
                    @media screen and (max-width:400px) {
                            .columnA, .columnB, .columnC {
                                    width: 100%;
                            }
                    }
            </style>
    </head>
            <body>
                            <!-- header of website layout -->
            <div class = "header">
```

```
                                <h2 style = "color:white;font-size:200%">
                                        DYP-ATU
                                </h2>
                        </div>
                                        <!-- navigation menu of website layout -->
                        <div class = "nav_menu">
                                <a href = "#">MCA-I</a>
        <div class="subnav">
          <button class="subnavbtn">MCA-II <i class="sss"></i></button>
          <div class="subnav-content">
            <a href="C:\Users\DELL\Desktop\sports.jpg">WT</a>
            <a href="#DA">DA</a>
            <a href="#DSA">DSA</a>
          </div>
         </div>
                                <a href = "#">BCA-I</a>
                                <a href = "#">BCA-II</a>
                                <a href = "#">B-tech-I</a>
                                <a href = "#">B-tech-II</a>
                        </div>
                        <!-- Content section of website layout -->
                        <div class = "row">
                                        <div class = "columnA">
                                        <h2>MCA</h2>
                                <p>The MCA programme is comprehensive and structured in a way
that best suits the industry. The syllabus is designed in a way that provides in-depth information on
subjects such as computer architecture, computer networks and programming languages. The study
provides an opportunity to focus on a variety of areas. Students can specialise in programme
management, application software, MIS Management Information System, Internet and
communications.</p>
                                </div>
                                                <div class = "columnB">
                                        <h2>BCA</h2>
```

```
	<p>The full form of BCA is Bachelor's in Computer Application. BCA is a 3-year undergraduate degree programme that focuses on knowledge of the basics of computer application and software development. A BCA degree is considered to be at par with a BTech/BE degree in Computer Science or Information Technology. The degree helps interested students in setting up a sound academic base for an advanced career in Computer Applications. The course of BCA includes database management systems, operating systems, software engineering, web technology and languages such as C, C++, HTML, Java etc. </p>
					</div>
					<div class = "columnC">
						<h2>B-Tech</h2>
	<p>In India, Bachelors in Technology (B.Tech) is a prestigious and technical engineering degree programme offered to undergraduate applicants after four years of intensive research and practical experience in the field. A bachelor's degree in engineering (B. Tech. or B.E.) is the starting point for a successful career in engineering.</p>
					</div>
				</div>
			</body>
	</html>
```



DYP-ATU

MCA-I MCA-II BCA-I BCA-II B-tech-I B-tech-II

MCA

The MCA programme is comprehensive and structured in a way that best suits the industry. The syllabus is designed in a way that provides in-depth information on subjects such as computer architecture, computer networks and programming languages. The study provides an opportunity to focus on a variety of areas. Students can specialise in programme management, application software, MIS Management Information System, Internet and communications.

BCA

The full form of BCA is Bachelor's in Computer Application. BCA is a 3-year undergraduate degree programme that focuses on knowledge of the basics of computer application and software development. A BCA degree is considered to be at par with a BTech/BE degree in Computer Science or Information Technology. The degree helps interested students in setting up a sound academic base for an advanced career in Computer Applications. The course of BCA includes database management systems, operating systems, software engineering, web technology and languages such as C, C++, HTML, Java etc.

B-Tech

In India, Bachelors in Technology (B.Tech) is a prestigious and technical engineering degree programme offered to undergraduate applicants after four years of intensive research and practical experience in the field. A bachelor's degree in engineering (B. Tech. or B.E.) is the starting point for a successful career in engineering.