# CSS Introduction

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

## CSS Demo - One HTML Page - Multiple Styles!

Here we will show one HTML page displayed with four different stylesheets. Click on the "Stylesheet 1", "Stylesheet 2", "Stylesheet 3", "Stylesheet 4" links below to see the different styles:

## Why Use CSS?

CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

### CSS Example

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

h1 {
  color: white;
  text-align: center;
}

p {
  font-family: verdana;
  font-size: 20px;
}
```

---

## CSS Solved a Big Problem

HTML was NEVER intended to contain tags for formatting a web page!

HTML was created to describe the content of a web page, like:

<h1>This is a heading</h1>

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

When tags like <font>, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large websites, where fonts and color information were added to every single page, became a long and expensive process.

To solve this problem, the World Wide Web Consortium (W3C) created CSS.

CSS removed the style formatting from the HTML page!

If you don't know what HTML is, we suggest that you read our HTML Tutorial.

---

## CSS Saves a Lot of Work!

The style definitions are normally saved in external .css files.

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

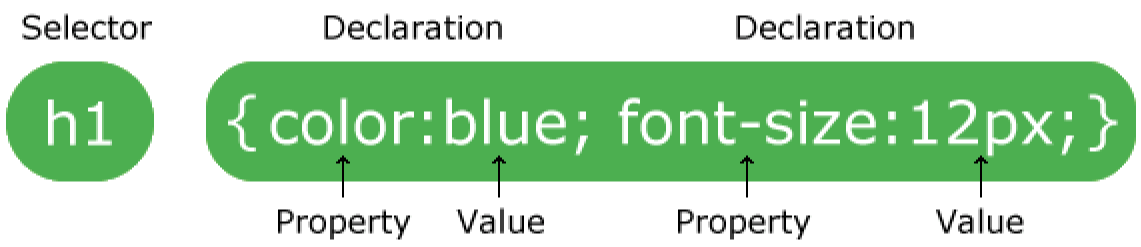
# CSS Syntax

---

A CSS rule consists of a selector and a declaration block.

---

## 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;
}
```

*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

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)

This page will explain the most basic CSS selectors.

---

## 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;
}
```

---

## 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;
}
```

**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;
}
```

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;
}
```

HTML elements can also refer to more than one class.

### 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>
```

**Note:** A class name cannot start with a number!

## 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;
}
```

---

## 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;
}
```

## 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>
```

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 (such as `margin-left: 20 px;`). The correct way is: `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>
```

---

## 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>
```

**Tip:** An inline style loses many of the advantages of a style sheet (by mixing content with presentation). Use this method sparingly.

---

## Multiple Style Sheets

If some properties have been defined for the same selector (element) in different style sheets, the value from the last read style sheet will be used.

Assume that an **external style sheet** has the following style for the <h1> element:

```
h1 {
  color: navy;
}
```

Then, assume that an **internal style sheet** also has the following style for the <h1> element:

```
h1 {
  color: orange;
}
```

### Example

If the internal style is defined **after** the link to the external style sheet, the <h1> elements will be "orange":

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
<style>
h1 {
  color: orange;
}
</style>
</head>
```

### Example

However, if the internal style is defined **before** the link to the external style sheet, the <h1> elements will be "navy":

```
<head>
<style>
```

```
h1 {
  color: orange;
}
</style>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
```

---

## Cascading Order

What style will be used when there is more than one style specified for an HTML element?

All the styles in a page will "cascade" into a new "virtual" style sheet by the following rules, where number one has the highest priority:

1. Inline style (inside an HTML element)
2. External and internal style sheets (in the head section)
3. Browser default

So, an inline style has the highest priority, and will override external and internal styles and browser defaults.

# CSS Comments

---

CSS comments are not displayed in the browser, but they can help document your source code.

---

## CSS Comments

Comments are used to explain the code, and may help when you edit the source code at a later date.

Comments are ignored by browsers.

A CSS comment is placed inside the `<style>` element, and starts with `/*` and ends with `*/`:

### Example

```
/* This is a single-line comment */
p {
  color: red;
}
```

You can add comments wherever you want in the code:

### Example

```
p {
  color: red;  /* Set text color to red */
}
```

Comments can also span multiple lines:

### Example

```
/* This is
a multi-line
comment */

p {
  color: red;
}
```

---

## HTML and CSS Comments

From the HTML tutorial, you learned that you can add comments to your HTML source by using the `<!--...-->` syntax.

In the following example, we use a combination of HTML and CSS comments:

### Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
  color: red; /* Set text color to red */
}
</style>
</head>
```

```
<body>

<h2>My Heading</h2>

<!-- These paragraphs will be red -->
<p>Hello World!</p>
<p>This paragraph is styled with CSS.</p>
<p>CSS comments are not shown in the output.</p>

</body>
</html>
```

# CSS Colors

---

Colors are specified using predefined color names, or RGB, HEX, HSL, RGBA, HSLA values.

---

## CSS Color Names

In CSS, a color can be specified by using a predefined color name:

Tomato

Orange

DodgerBlue

MediumSeaGreen

Gray

SlateBlue

Violet

LightGray

CSS/HTML support 140 standard color names.

---

## CSS Background Color

You can set the background color for HTML elements:

Hello World

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

### Example

```
<h1 style="background-color:DodgerBlue;">Hello World</h1>
<p style="background-color:Tomato;">Lorem ipsum...</p>
```

## CSS Text Color

You can set the color of text:

**Hello World**

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

### Example

```
<h1 style="color:Tomato;">Hello World</h1>
<p style="color:DodgerBlue;">Lorem ipsum...</p>
<p style="color:MediumSeaGreen;">Ut wisi enim...</p>
```

## CSS Border Color

You can set the color of borders:

**Hello World**

**Hello World**

**Hello World**

### Example

```
<h1 style="border:2px solid Tomato;">Hello World</h1>
<h1 style="border:2px solid DodgerBlue;">Hello World</h1>
<h1 style="border:2px solid Violet;">Hello World</h1>
```

## CSS Color Values

In CSS, colors can also be specified using RGB values, HEX values, HSL values, RGBA values, and HSLA values:

Same as color name "Tomato":

**rgb(255, 99, 71)**

**#ff6347**

Same as color name "Tomato", but 50% transparent:

### Example

```
<h1 style="background-color:rgb(255, 99, 71);">...</h1>
<h1 style="background-color:#ff6347;">...</h1>
<h1 style="background-color:hsl(9, 100%, 64%);">...</h1>

<h1 style="background-color:rgba(255, 99, 71, 0.5);">...</h1>
<h1 style="background-color:hsla(9, 100%, 64%, 0.5);">...</h1>
```

# CSS RGB Colors

---

An RGB color value represents RED, GREEN, and BLUE light sources.

---

## RGB Value

In CSS, a color can be specified as an RGB value, using this formula:

**rgb(*red, green, blue*)**

Each parameter (red, green, and blue) defines the intensity of the color between 0 and 255.

For example, rgb(255, 0, 0) is displayed as red, because red is set to its highest value (255) and the others are set to 0.

To display black, set all color parameters to 0, like this: rgb(0, 0, 0).

To display white, set all color parameters to 255, like this: rgb(255, 255, 255).

Experiment by mixing the RGB values below:

**rgb(255, 99, 71)**

RED

255

GREEN

99

BLUE

71

### Example

**rgb(255, 0, 0)**

rgb(0, 0, 255)

rgb(60, 179, 113)

rgb(238, 130, 238)

rgb(255, 165, 0)

rgb(106, 90, 205)

Shades of gray are often defined using equal values for all the 3 light sources:

**Example**

rgb(0, 0, 0)

rgb(60, 60, 60)

rgb(120, 120, 120)

`rgb(180, 180, 180)`

`rgb(240, 240, 240)`

`rgb(255, 255, 255)`

---

## RGBA Value

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 (not transparent at all):

Experiment by mixing the RGBA values below:

RED

255

GREEN

99

BLUE

71

ALPHA

0.5

### Example

```
rgba(255, 99, 71, 0)
```

```
rgba(255, 99, 71, 0.2)
```

# CSS HEX Colors

A hexadecimal color is specified with: #RRGGBB, where the RR (red), GG (green) and BB (blue) hexadecimal integers specify the components of the color.

---

## HEX Value

In CSS, a color can be specified using a hexadecimal value in the form:

### *#rrggbb*

Where rr (red), gg (green) and bb (blue) are hexadecimal values between 00 and ff (same as decimal 0-255).

For example, #ff0000 is displayed as red, because red is set to its highest value (ff) and the others are set to the lowest value (00).

Experiment by mixing the HEX values below:

**#ff6347**

RED

ff

GREEN

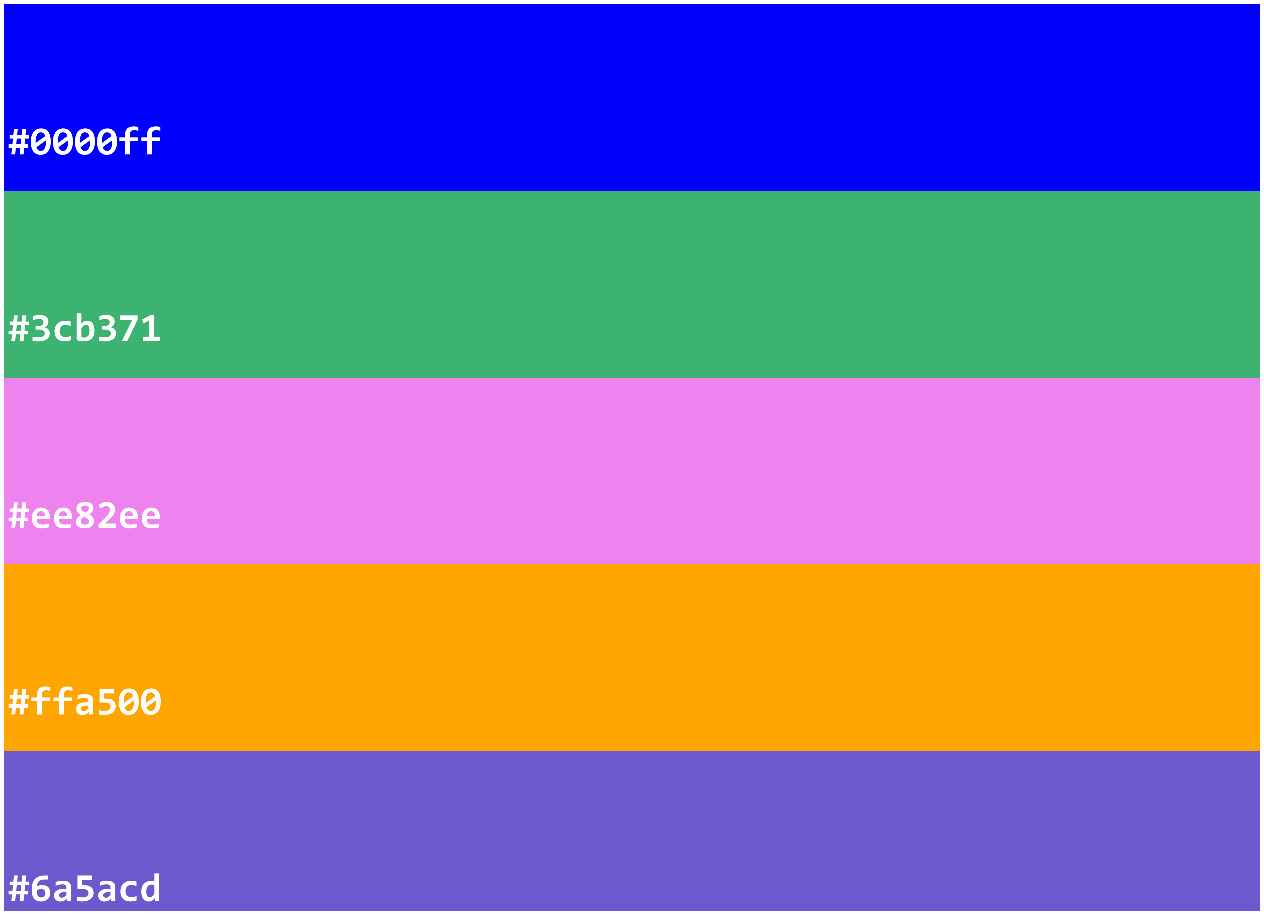
63

BLUE

47

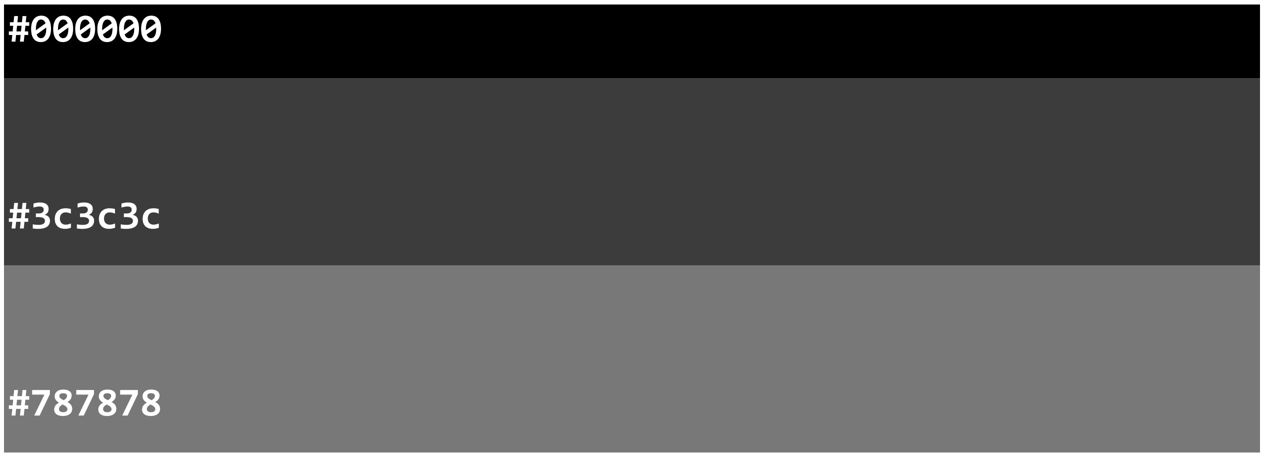
### Example

**#ff0000**



Shades of gray are often defined using equal values for all the 3 light sources:

**Example**



**#b4b4b4**

**#f0f0f0**

**#ffffff**

---

## 3 Digit HEX Value

Sometimes you will see a 3-digit hex code in the CSS source.

The 3-digit hex code is a shorthand for some 6-digit hex codes.

The 3-digit hex code has the following form:

***#rgb***

Where r, g, and b represents the red, green, and blue components with values between 0 and f.

The 3-digit hex code can only be used when both the values (RR, GG, and BB) are the same for each components. So, if we have #ff00cc, it can be written like this: #f0c.

Here is an example:

### Example

```
body {
  background-color: #fc9; /* same as #ffcc99 */
}

h1 {
  color: #f0f; /* same as #ff00ff */
}
```

```
p {
  color: #b58; /* same as #bb5588 */
}
```

# CSS HSL Colors

---

HSL stands for hue, saturation, and lightness.

---

## HSL Value

In CSS, a color can be specified using hue, saturation, and lightness (HSL) in the form:

### `hsl(`***`hue, saturation, lightness`***`)`

Hue is a degree on the color wheel from 0 to 360. 0 is red, 120 is green, and 240 is blue.

Saturation is a percentage value, 0% means a shade of gray, and 100% is the full color.

Lightness is also a percentage, 0% is black, 50% is neither light or dark, 100% is white

Experiment by mixing the HSL values below:

**`hsl(0, 100%, 50%)`**

HUE

`0`

SATURATION

`100%`

LIGHTNESS

`50%`

**Example**

---

## Saturation

Saturation can be described as the intensity of a color.

100% is pure color, no shades of gray

50% is 50% gray, but you can still see the color.

0% is completely gray, you can no longer see the color.

### Example

---

## Lightness

The lightness of a color can be described as how much light you want to give the color, where 0% means no light (black), 50% means 50% light (neither dark nor light) 100% means full lightness (white).

### Example

`hsl(0, 100%, 90%)`

`hsl(0, 100%, 100%)`

---

Shades of gray are often defined by setting the hue and saturation to 0, and adjust the lightness from 0% to 100% to get darker/lighter shades:

### Example

**hsl(0, 0%, 94%)**

**hsl(0, 0%, 100%)**

---

## HSLA Value

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`*`)`**

The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all):

Experiment by mixing the HSLA values below:

HUE

`0`

SATURATION

`100%`

LIGHTNESS

`50%`

ALPHA

`0.5`

**Example**

```
hsla(9, 100%, 64%, 0)
```

```
hsla(9, 100%, 64%, 0.2)
```

# CSS Backgrounds

---

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

---

In these chapters, you will learn about the 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;
}
```

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;
}
```

---

## 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;
}
```

**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).

**Tip:** You will learn more about RGBA Colors in our CSS Colors Chapter.

### Example

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

# CSS Background Image

---

## 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");
}
```

### Example

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

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

**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");
}
```

# CSS Background Repeat

---

## CSS background-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");
}
```

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;
}
```

**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;
}
```

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;
}
```

# CSS Background Attachment

---

## 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;
}
```

### 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;
}
```

# CSS Background Shorthand

---

## 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;
}
```

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 Borders

---

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

---

I have borders on all sides.

I have a red bottom border.

I have rounded borders.

I have a blue left 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.

**Note:** None of the OTHER CSS border properties (which you will learn more about in the next chapters) will have ANY effect unless the `border-style` property is set!

# CSS Border Width

## CSS Border Width

The `border-width` property specifies the width of the four borders.

The width can be set as a specific size (in px, pt, cm, em, etc) or by using one of the three pre-defined values: thin, medium, or thick:

### Example

Demonstration of the different border widths:

```
p.one {
  border-style: solid;
  border-width: 5px;
}

p.two {
  border-style: solid;
  border-width: medium;
}

p.three {
  border-style: dotted;
  border-width: 2px;
}

p.four {
  border-style: dotted;
  border-width: thick;
}
```

Result:

5px border-width

medium border-width

2px border-width

thick border-width

---

## Specific Side Widths

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

### Example

```
p.one {
  border-style: solid;
  border-width: 5px 20px; /* 5px top and bottom, 20px on the sides */
}

p.two {
  border-style: solid;
  border-width: 20px 5px; /* 20px top and bottom, 5px on the sides */
}

p.three {
  border-style: solid;
  border-width: 25px 10px 4px 35px; /* 25px top, 10px right, 4px bottom and 35px left */
}
```

# CSS Border Color

---

## CSS Border Color

The `border-color` property is used to set the color of the four borders.

The color can be set by:

- name - specify a color name, like "red"
- HEX - specify a HEX value, like "#ff0000"
- RGB - specify a RGB value, like "rgb(255,0,0)"
- HSL - specify a HSL value, like "hsl(0, 100%, 50%)"
- transparent

**Note:** If `border-color` is not set, it inherits the color of the element.

### Example

Demonstration of the different border colors:

```
p.one {
  border-style: solid;
  border-color: red;
}

p.two {
  border-style: solid;
  border-color: green;
}

p.three {
  border-style: dotted;
  border-color: blue;
}
```

Result:

Red border

Green border

Blue border

---

## Specific Side Colors

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

### Example

```
p.one {
  border-style: solid;
  border-color: red green blue yellow; /* red top, green right, blue bottom and yellow left */
}
```

---

## HEX Values

The color of the border can also be specified using a hexadecimal value (HEX):

### Example

```
p.one {
  border-style: solid;
  border-color: #ff0000; /* red */
}
```

---

## RGB Values

Or by using RGB values:

### Example

```
p.one {
  border-style: solid;
  border-color: rgb(255, 0, 0); /* red */
}
```

---

## HSL Values

You can also use HSL values:

**Example**

```
p.one {
  border-style: solid;
  border-color: hsl(0, 100%, 50%); /* red */
}
```

You can learn more about HEX, RGB and HSL values in our CSS Colors chapters.

# CSS Border Sides

---

## CSS Border - Individual Sides

From the examples on the previous pages, you have seen that it is possible to specify a different border for each side.

In CSS, there are also properties for specifying each of the borders (top, right, bottom, and left):

**Example**

```
p {
  border-top-style: dotted;
  border-right-style: solid;
  border-bottom-style: dotted;
  border-left-style: solid;
}
```

Result:

Different Border Styles

The example above gives the same result as this:

**Example**

```
p {
  border-style: dotted solid;
}
```

So, here is how it works:

If the `border-style` property has four values:

- **border-style: dotted solid double dashed;**
    - top border is dotted
    - right border is solid
    - bottom border is double
    - left border is dashed

If the `border-style` property has three values:

- **border-style: dotted solid double;**
    - top border is dotted
    - right and left borders are solid
    - bottom border is double

If the `border-style` property has two values:

- **border-style: dotted solid;**
    - top and bottom borders are dotted
    - right and left borders are solid

If the `border-style` property has one value:

- **border-style: dotted;**
    - all four borders are dotted

## Example

```
/* Four values */
p {
  border-style: dotted solid double dashed;
}

/* Three values */
p {
  border-style: dotted solid double;
}

/* Two values */
p {
  border-style: dotted solid;
}

/* One value */
p {
  border-style: dotted;
}
```

The `border-style` property is used in the example above. However, it also works with `border-width` and `border-color`.

# CSS Shorthand Border Property

---

## CSS Border - Shorthand Property

Like you saw in the previous page, there are many properties to consider when dealing with borders.

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

The `border` property is a shorthand property for the following individual border properties:

- `border-width`
- `border-style` (required)
- `border-color`

### Example

```
p {
  border: 5px solid red;
}
```

Result:

Some text

You can also specify all the individual border properties for just one side:

### Left Border

```
p {
  border-left: 6px solid red;
  background-color: lightgrey;
}
```

Result:

Some text

### Bottom Border

```
p {
  border-bottom: 6px solid red;
  background-color: lightgrey;
}
```

Result:

Some text

# CSS Rounded Borders

---

## CSS Rounded Borders

The `border-radius` property is used to add rounded borders to an element:

Normal border

Round border

Rounder border

Roundest border

### Example

```
p {
  border: 2px solid red;
  border-radius: 5px;
}
```

---

## More Examples

All the top border properties in one declaration
This example demonstrates a shorthand property for setting all of the properties for the top border in one declaration.

Set the style of the bottom border
This example demonstrates how to set the style of the bottom border.

Set the width of the left border
This example demonstrates how to set the width of the left border.

Set the color of the four borders
This example demonstrates how to set the color of the four borders. It can have from one to four colors.

Set the color of the right border
This example demonstrates how to set the color of the right border.

## All CSS Border Properties

| Property | Description |
|---|---|
| border | Sets all the border properties in one declaration |
| border-bottom | Sets all the bottom border properties in one declaration |
| border-bottom-color | Sets the color of the bottom border |
| border-bottom-style | Sets the style of the bottom border |
| border-bottom-width | Sets the width of the bottom border |
| border-color | Sets the color of the four borders |
| border-left | Sets all the left border properties in one declaration |
| border-left-color | Sets the color of the left border |
| border-left-style | Sets the style of the left border |
| border-left-width | Sets the width of the left border |
| border-radius | Sets all the four border-*-radius properties for rounded corners |
| border-right | Sets all the right border properties in one declaration |
| border-right-color | Sets the color of the right border |

| | |
|---|---|
| border-right-style | Sets the style of the right border |
| border-right-width | Sets the width of the right border |
| border-style | Sets the style of the four borders |
| border-top | Sets all the top border properties in one declaration |
| border-top-color | Sets the color of the top border |
| border-top-style | Sets the style of the top border |
| border-top-width | Sets the width of the top border |
| border-width | Sets the width of the four borders |

# CSS Margins

---

Margins are used to create space around elements, outside of any defined borders.

---

This element has a margin of 70px.

---

## 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;
}
```

---

---

## 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;
}
```

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;
}
```

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;
}
```

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;
}
```

## 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;
}
```

## 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;
}
```

# CSS Margin Collapse

Sometimes two margins collapse into a single margin.

---

## Margin Collapse

Top and bottom margins of elements are sometimes collapsed into a single margin that is equal to the largest of the two margins.

This does not happen on left and right margins! Only top and bottom margins!

Look at the following example:

### Example

Demonstration of margin collapse:

```
h1 {
  margin: 0 0 50px 0;
}

h2 {
  margin: 20px 0 0 0;
}
```

In the example above, the <h1> element has a bottom margin of 50px and the <h2> element has a top margin set to 20px.

Common sense would seem to suggest that the vertical margin between the <h1> and the <h2> would be a total of 70px (50px + 20px). But due to margin collapse, the actual margin ends up being 50px.

## All CSS Margin Properties

| Property | Description |
|---|---|
| margin | A shorthand property for setting 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 Padding

---

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

---

This element has a padding of 70px.

---

## 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;
}
```

---

---

## 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;
}
```

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;
}
```

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;
}
```

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;
}
```

---

## 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;
}
```

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 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;
}
```

---

## More Examples

Set the left padding
This example demonstrates how to set the left padding of a <p> element.

Set the right padding
This example demonstrates how to set the right padding of a <p> element.

Set the top padding
This example demonstrates how to set the top padding of a <p> element.

Set the bottom padding
This example demonstrates how to set the bottom padding of a <p> element.

## 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 Height and Width

---

The CSS `height` and `width` properties are used to set the height and width of an element.

The CSS `max-width` property is used to set the maximum width of an element.

---

This element has a height of 50 pixels and a width of 100%.

---

## CSS Setting height and width

The `height` and `width` properties are used to set the height and width of an element.

The height and width properties do not include padding, borders, or margins. It sets the height/width of the area inside the padding, border, and margin of the element.

---

## CSS height and width Values

The `height` and `width` properties may have the following values:

- `auto` - This is default. The browser calculates the height and width
- `length` - Defines the height/width in px, cm etc.
- `%` - Defines the height/width in percent of the containing block
- `initial` - Sets the height/width to its default value
- `inherit` - The height/width will be inherited from its parent value

---

## CSS height and width Examples

This element has a height of 200 pixels and a width of 50%

### Example

Set the height and width of a <div> element:

```
div {
  height: 200px;
  width: 50%;
  background-color: powderblue;
}
```

This element has a height of 100 pixels and a width of 500 pixels.

### Example

Set the height and width of another <div> element:

```
div {
  height: 100px;
  width: 500px;
  background-color: powderblue;
}
```

**Note:** Remember that the `height` and `width` properties do not include padding, borders, or margins! They set the height/width of the area inside the padding, border, and margin of the element!

---

---

## Setting max-width

The `max-width` property is used to set the maximum width of an element.

The `max-width` can be specified in *length values*, like px, cm, etc., or in percent (%) of the containing block, or set to none (this is default. Means that there is no maximum width).

The problem with the `<div>` above occurs when the browser window is smaller than the width of the element (500px). The browser then adds a horizontal scrollbar to the page.

Using `max-width` instead, in this situation, will improve the browser's handling of small windows.

**Tip:** Drag the browser window to smaller than 500px wide, to see the difference between the two divs!

This element has a height of 100 pixels and a max-width of 500 pixels.

**Note:** The value of the `max-width` property overrides `width`.

### Example

This <div> element has a height of 100 pixels and a max-width of 500 pixels:

```
div {
  max-width: 500px;
  height: 100px;
  background-color: powderblue;
}
```

---

## Try it Yourself - Examples

Set the height and width of elements
This example demonstrates how to set the height and width of different elements.

Set the height and width of an image using percent
This example demonstrates how to set the height and width of an image using a percent value.

Set min-width and max-width of an element
This example demonstrates how to set a minimum width and a maximum width of an element using a pixel value.

Set min-height and max-height of an element
This example demonstrates how to set a minimum height and a maximum height of an element using a pixel value.

## All CSS Dimension Properties

| Property | Description |
|---|---|
| height | Sets the height of an element |
| max-height | Sets the maximum height of an element |
| max-width | Sets the maximum width of an element |

| | |
|---|---|
| min-height | Sets the minimum height of an element |
| min-width | Sets the minimum width of an element |
| width | Sets the width of an element |

# CSS Box Model

All HTML elements can be considered as boxes.

## The 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;
}
```

## 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;
}
```

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 Outline

---

An outline is a line drawn outside the element's border.

---

This element has a black border and a green outline with a width of 10px.

---

## CSS Outline

An outline is a line that is drawn around elements, OUTSIDE the borders, to make the element "stand out".

CSS has the following outline properties:

- `outline-style`
- `outline-color`
- `outline-width`
- `outline-offset`
- `outline`

**Note:** Outline differs from borders! Unlike border, the outline is drawn outside the element's border, and may overlap other content. Also, the outline is NOT a part of the element's dimensions; the element's total width and height is not affected by the width of the outline.

---

## CSS Outline Style

The `outline-style` property specifies the style of the outline, and can have one of the following values:

- `dotted` - Defines a dotted outline
- `dashed` - Defines a dashed outline
- `solid` - Defines a solid outline
- `double` - Defines a double outline
- `groove` - Defines a 3D grooved outline
- `ridge` - Defines a 3D ridged outline
- `inset` - Defines a 3D inset outline
- `outset` - Defines a 3D outset outline
- `none` - Defines no outline
- `hidden` - Defines a hidden outline

The following example shows the different `outline-style` values:

### Example

Demonstration of the different outline styles:

```
p.dotted {outline-style: dotted;}
p.dashed {outline-style: dashed;}
p.solid {outline-style: solid;}
p.double {outline-style: double;}
p.groove {outline-style: groove;}
p.ridge {outline-style: ridge;}
p.inset {outline-style: inset;}
p.outset {outline-style: outset;}
```

Result:

A dotted outline.

A dashed outline.

A solid outline.

A double outline.

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

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

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

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

**Note:** None of the other outline properties (which you will learn more about in the next chapters) will have ANY effect unless the `outline-style` property is set!

# CSS Outline Width

---

## CSS Outline Width

The `outline-width` property specifies the width of the outline, and can have one of the following values:

- thin (typically 1px)
- medium (typically 3px)
- thick (typically 5px)
- A specific size (in px, pt, cm, em, etc)

The following example shows some outlines with different widths:

A thin outline.

A medium outline.

A thick outline.

A 4px thick outline.

**Example**

```
p.ex1 {
  border: 1px solid black;
  outline-style: solid;
  outline-color: red;
  outline-width: thin;
}

p.ex2 {
  border: 1px solid black;
  outline-style: solid;
  outline-color: red;
  outline-width: medium;
}

p.ex3 {
  border: 1px solid black;
  outline-style: solid;
  outline-color: red;
  outline-width: thick;
}

p.ex4 {
  border: 1px solid black;
  outline-style: solid;
  outline-color: red;
  outline-width: 4px;
}
```

## CSS Outline Color

---

### CSS Outline Color

The `outline-color` property is used to set the color of the outline.

The color can be set by:

- name - specify a color name, like "red"
- HEX - specify a hex value, like "#ff0000"
- RGB - specify a RGB value, like "rgb(255,0,0)"
- HSL - specify a HSL value, like "hsl(0, 100%, 50%)"
- invert - performs a color inversion (which ensures that the outline is visible, regardless of color background)

The following example shows some different outlines with different colors. Also notice that these elements also have a thin black border inside the outline:

A solid red outline.

A dotted blue outline.

An outset grey outline.

### Example

```
p.ex1 {
  border: 2px solid black;
  outline-style: solid;
  outline-color: red;
}

p.ex2 {
  border: 2px solid black;
  outline-style: dotted;
  outline-color: blue;
}

p.ex3 {
  border: 2px solid black;
  outline-style: outset;
  outline-color: grey;
}
```

## HEX Values

The outline color can also be specified using a hexadecimal value (HEX):

**Example**

```
p.ex1 {
  outline-style: solid;
  outline-color: #ff0000; /* red */
}
```

---

## RGB Values

Or by using RGB values:

**Example**

```
p.ex1 {
  outline-style: solid;
  outline-color: rgb(255, 0, 0); /* red */
}
```

---

## HSL Values

You can also use HSL values:

**Example**

```
p.ex1 {
  outline-style: solid;
  outline-color: hsl(0, 100%, 50%); /* red */
}
```

You can learn more about HEX, RGB and HSL values in our CSS Colors chapters.

---

## Invert Color

The following example uses `outline-color: invert`, which performs a color inversion. This ensures that the outline is visible, regardless of color background:

A solid invert outline.

**Example**

```
p.ex1 {
  border: 1px solid yellow;
  outline-style: solid;
  outline-color: invert;
}
```

# CSS Outline Shorthand

## CSS Outline - Shorthand property

The `outline` property is a shorthand property for setting the following individual outline properties:

- `outline-width`
- `outline-style` (required)
- `outline-color`

The `outline` property is specified as one, two, or three values from the list above. The order of the values does not matter.

The following example shows some outlines specified with the shorthand `outline` property:

A dashed outline.

A dotted red outline.

A 5px solid yellow outline.

A thick ridge pink outline.

### Example

```
p.ex1 {outline: dashed;}
p.ex2 {outline: dotted red;}
p.ex3 {outline: 5px solid yellow;}
p.ex4 {outline: thick ridge pink;}
```

# CSS Outline Offset

## CSS Outline Offset

The `outline-offset` property adds space between an outline and the edge/border of an element. The space between an element and its outline is transparent.

The following example specifies an outline 15px outside the border edge:

This paragraph has an outline 15px outside the border edge.

### Example

```
p {
  margin: 30px;
  border: 1px solid black;
  outline: 1px solid red;
  outline-offset: 15px;
}
```

The following example shows that the space between an element and its outline is transparent:

This paragraph has an outline of 15px outside the border edge.

### Example

```
p {
  margin: 30px;
  background: yellow;
  border: 1px solid black;
  outline: 1px solid red;
  outline-offset: 15px;
}
```

# CSS Text

---

CSS has a lot of properties for formatting text.

---

## TEXT FORMATTING

This text is styled with some of the text formatting properties. The heading uses the text-align, text-transform, and color properties. The paragraph is indented, aligned, and the space between characters is specified. The underline is removed from this colored "Try it Yourself" link.

---

## 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;
}
```

**Note:** For W3C compliant CSS: If you define the `color` property, you must also define the `background-color`.

---

## 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;
}
```

# CSS Text Alignment

---

## 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;
}
```

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;
}
```

---

## 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;
}
```

---

## Vertical Alignment

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

This example demonstrates how to set the vertical alignment of an image in a text:

### Example

```
img.top {
  vertical-align: top;
}

img.middle {
  vertical-align: middle;
}

img.bottom {
  vertical-align: bottom;
}
```

# CSS Text Decoration

## Text Decoration

The `text-decoration` property is used to set or remove decorations from text.

The value `text-decoration: none;` is often used to remove underlines from links:

### Example

```
a {
  text-decoration: none;
}
```

The other `text-decoration` values are used to decorate text:

### Example

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

h2 {
  text-decoration: line-through;
}

h3 {
  text-decoration: underline;
}
```

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

# CSS Text Transformation

## 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;
}
```

# CSS Text Spacing

---

## Text Indentation

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

### Example

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

---

## 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: 3px;
}

h2 {
  letter-spacing: -3px;
}
```

---

## 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;
}
```

---

## 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

```
h1 {
  word-spacing: 10px;
}

h2 {
  word-spacing: -5px;
}
```

---

## 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;
}
```

# CSS Text Shadow

---

## 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;
}
```

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

## Text shadow effect!

### Example

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

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

## Text shadow effect!

### Example

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

**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

## All CSS Text Properties

| Property | Description |
|---|---|
| color | Sets the color of text |
| direction | Specifies the text direction/writing direction |
| letter-spacing | Increases or decreases the space between characters in a text |
| line-height | Sets the line height |
| text-align | Specifies the horizontal alignment of text |
| text-decoration | Specifies the decoration added to text |
| text-indent | Specifies the indentation of the first line in a text-block |
| text-shadow | Specifies the shadow effect added to text |
| text-transform | Controls the capitalization of text |
| text-overflow | Specifies how overflowed content that is not displayed should be signaled to the user |
| 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 |

| | |
|---|---|
| white-space | Specifies how white-space inside an element is handled |
| word-spacing | Increases or decreases the space between words in a text |

# CSS Fonts

---

Choosing the right font for your website is important!

---

## 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

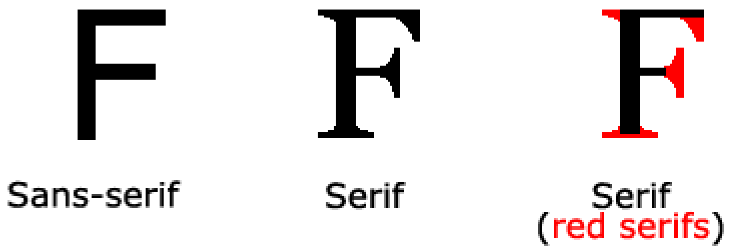
In CSS there are five generic font families:

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<br>Georgia<br>Garamond |
| Sans-serif | Arial<br>Verdana<br>Helvetica |
| Monospace | Courier New<br>Lucida Console<br>Monaco |
| Cursive | Brush Script MT<br>Lucida Handwriting |
| Fantasy | COPPERPLATE<br>Papyrus |

---

## The CSS font-family Property

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

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.

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

### 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;
}
```

# CSS Web Safe Fonts

---

## 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;
}
```

---

## 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)
- Helvetica (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**

# Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet.

0 1 2 3 4 5 6 7 8 9

## Verdana (sans-serif)

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

**Example**

# Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet.

0 1 2 3 4 5 6 7 8 9

## Helvetica (sans-serif)

The Helvetica font is loved by designers. It is suitable for many types of business.

**Example**

# Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet.

0 1 2 3 4 5 6 7 8 9

## Tahoma (sans-serif)

The Tahoma font has less space between the characters.

**Example**

# Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet.

0 1 2 3 4 5 6 7 8 9

## Trebuchet MS (sans-serif)

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

**Example**

# Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet.

0 1 2 3 4 5 6 7 8 9

## 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.

**Example**

# Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet.

0 1 2 3 4 5 6 7 8 9

## 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.

**Example**

# Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet.

0 1 2 3 4 5 6 7 8 9

---

## Garamond (serif)

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

### Example

## Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet.

0 1 2 3 4 5 6 7 8 9

---

## 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.

### Example

## Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet.

0 1 2 3 4 5 6 7 8 9

---

## 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**

## Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet.

0 1 2 3 4 5 6 7 8 9

**Tip:** Also check out all available Google Fonts and how to use them.

# CSS Font Fallbacks

---

## Commonly Used Font Fallbacks

Below are some commonly used font fallbacks, organized by the 5 generic font families:

- **Serif**
- **Sans-serif**
- **Monospace**
- **Cursive**
- **Fantasy**

---

## Serif Fonts

| font-family | Example text | Code |
|---|---|---|
| "Times New Roman", Times, serif | **This is a Heading** This is a paragraph. | |
| Georgia, serif | **This is a Heading** This is a paragraph. | |
| Garamond, serif | **This is a** | |

## Heading

This is a paragraph.

## Sans-Serif Fonts

| font-family | Example text | Code |
|---|---|---|
| Arial, Helvetica, sans-serif | **This is a Heading** This is a paragraph. | |
| Tahoma, Verdana, sans-serif | **This is a Heading** This is a paragraph. | |
| "Trebuchet MS", Helvetica, sans-serif | **This is a Heading** This is a paragraph. | |
| Georgia, Verdana, sans-serif | **This is a Heading** This is a paragraph. | |

## Monospace Fonts

| font-family | Example text | Code |
|---|---|---|
| "Courier New", Courier, monospace | **This is a Heading** This is a paragraph. | |

## Cursive Fonts

| font-family | Example text | Code |
|---|---|---|

| | | |
|---|---|---|
| "Brush Script MT", cursive | This is a Heading<br><br>This is a paragraph. | |

## Fantasy Fonts

| font-family | Example text | Code |
|---|---|---|
| Copperplate, Papyrus, fantasy | This is a Heading<br><br>This is a paragraph. | |

**Tip:** Also check out all available Google Fonts and how to use them.

# CSS Font Style

## 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;
}
```

## Font Weight

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

### Example

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

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

## 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;
}
```

# CSS Font Size

## 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;
}
```

**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 */
}
```

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;
}
```

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

---

## Responsive Font Size

The text size can be set with a `vw` unit, which means the "viewport width".

That way the text size will follow the size of the browser window:

# Hello World

Resize the browser window to see how the font size scales.

### Example

```
<h1 style="font-size:10vw">Hello World</h1>
```

Viewport is the browser window size. 1vw = 1% of viewport width. If the viewport is 50cm wide, 1vw is 0.5cm.

# CSS Google Fonts

---

## Google Fonts

If you do not want to use any of the standard fonts in HTML, you can use Google Fonts.

Google Fonts are free to use, and have more than 1000 fonts to choose from.

---

## How To Use Google Fonts

Just add a special style sheet link in the <head> section and then refer to the font in the CSS.

### Example

Here, we want to use a font named "Sofia" from Google Fonts:

```
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Sofia">
<style>
body {
  font-family: "Sofia", sans-serif;
}
</style>
</head>
```

Result:

# Sofia Font

Lorem ipsum dolor sit amet.

123456790

### Example

Here, we want to use a font named "Trirong" from Google Fonts:

```
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Trirong">
<style>
body {
  font-family: "Trirong", serif;
}
</style>
</head>
```

Result:

# Trirong Font

Lorem ipsum dolor sit amet.

123456790

### Example

Here, we want to use a font named "Audiowide" from Google Fonts:

```
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Audiowide">
<style>
body {
  font-family: "Audiowide", sans-serif;
}
</style>
</head>
```

Result:

## Audiowide Font

Lorem ipsum dolor sit amet.

123456790

**Note:** When specifying a font in CSS, always list at minimum one fallback font (to avoid unexpected behaviors). So, also here you should add a generic font family (like serif or sans-serif) to the end of the list.

For a list of all available Google Fonts, visit our How To - Google Fonts Tutorial.

---

## Use Multiple Google Fonts

To use multiple Google fonts, just separate the font names with a pipe character (|), like this:

### Example

Request multiple fonts:

```
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Audiowide|Sofia|Trirong">
```

```
<style>
h1.a {font-family: "Audiowide", sans-serif;}
h1.b {font-family: "Sofia", sans-serif;}
h1.c {font-family: "Trirong", serif;}
</style>
</head>
```

Result:

# Audiowide Font

# Sofia Font

# Trirong Font

**Note:** Requesting multiple fonts may slow down your web pages! So be careful about that.

---

---

## Styling Google Fonts

Of course you can style Google Fonts as you like, with CSS!

### Example

Style the "Sofia" font:

```
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Sofia">
<style>
body {
  font-family: "Sofia", sans-serif;
  font-size: 30px;
  text-shadow: 3px 3px 3px #ababab;
}
</style>
</head>
```

Result:

# Sofia Font

Lorem ipsum dolor sit amet.

123456790

---

## Enabling Font Effects

Google have also enabled different font effects that you can use.

First add `effect=`*`effectname`* to the Google API, then add a special class name to the element that is going to use the special effect. The class name always starts with `font-effect-` and ends with the *`effectname`*.

### Example

Add the fire effect to the "Sofia" font:

```
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Sofia&effect=fire">
<style>
body {
  font-family: "Sofia", sans-serif;
  font-size: 30px;
}
</style>
</head>
<body>

<h1 class="font-effect-fire">Sofia on Fire</h1>

</body>
```

Result:

# Sofia on Fire

To request multiple font effects, just separate the effect names with a pipe character ( | ), like this:

## Example

Add multiple effects to the "Sofia" font:

```
<head>
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Sofia&effect=neon|outline|emboss|shadow-
multiple">
<style>
body {
  font-family: "Sofia", sans-serif;
  font-size: 30px;
}
</style>
</head>
<body>

<h1 class="font-effect-neon">Neon Effect</h1>
<h1 class="font-effect-outline">Outline Effect</h1>
<h1 class="font-effect-emboss">Emboss Effect</h1>
<h1 class="font-effect-shadow-multiple">Multiple Shadow Effect</h1>

</body>
```

Result:

**Neon Effect**

**Outline Effect**

**Emboss Effect**

**Multiple Shadow Effect**

# CSS Great Font Pairings

---

Great font pairings are essential to great design.

---

## Font Pairing Rules

Here are some basic rules to create great font pairings:

### 1. Compliment

It is always safe to find font pairings that complement one another.

A great font combination should harmonize, without being too similar or too different.

### 2. Use Font Superfamilies

A font superfamily is a set of fonts designed to work well together. So, using different fonts within the same superfamily is safe.

For example, the Lucida superfamily contains the following fonts: Lucida Sans, Lucida Serif, Lucida Typewriter Sans, Lucida Typewriter Serif and Lucida Math.

### 3. Contrast is King

Two fonts that are too similar will often conflict. However, contrasts, done the right way, brings out the best in each font.

Example: Combining serif with sans serif is a well known combination.

A strong superfamily includes both serif and sans serif variations of the same font (e.g. Lucida and Lucida Sans).

### 4. Choose Only One Boss

One font should be the boss. This establishes a hierarchy for the fonts on your page. This can be achieved by varying the size, weight and color.

### Example

No doubt "Georgia" is the boss here:

```
body {
  background-color: black;
  font-family: Verdana, sans-serif;
  font-size: 16px;
  color: gray;
}

h1 {
  font-family: Georgia, serif;
  font-size: 60px;
  color: white;
}
```

Below, we have shown some popular font pairings that will suit many brands and contexts.

---

---

## Georgia and Verdana

Georgia and Verdana is a classic combination. It also sticks to the web safe font standards:

### Example

Use the "Georgia" font for headings, and "Verdana" for text:

# Beautiful Norway

Norway has a total area of 385,252 square kilometres and a population of 5,438,657 (December 2020). Norway is bordered by Sweeden, Finland and Russia to the north-east, and the Skagerrak to the south, with Denmark on the other side.

Norway has beautiful mountains, glaciers and stunning fjords. Oslo, the capital, is a city of green spaces and museums. Bergen, with colorful wooden houses, is the starting point for cruises to the dramatic Sognefjord. Norway is also known for fishing, hiking and skiing.

---

## Helvetica and Garamond

Helvetica and Garamond is another classic combination that uses web safe fonts:

### Example

Use the "Helvetica" font for headings, and "Garamond" for text:

# Beautiful Norway

Norway has a total area of 385,252 square kilometres and a population of 5,438,657 (December 2020). Norway is bordered by Sweeden, Finland and Russia to the north-east, and the Skagerrak to the south, with Denmark on the other side.

Norway has beautiful mountains, glaciers and stunning fjords. Oslo, the capital, is a city of green spaces and museums. Bergen, with colorful wooden houses, is the starting point for cruises to the dramatic Sognefjord. Norway is also known for fishing, hiking and skiing.

---

## Popular Google Font Pairings

If you do not want to use standard fonts in HTML, you can use Google Fonts.

Google Fonts are free to use, and have more than 1000 fonts to choose from.

Below are some popular Google Web Font Pairings.

---

## Merriweather and Open Sans

### Example

Use the "Merriweather" font for headings, and "Open Sans" for text:

# Beautiful Norway

Norway has a total area of 385,252 square kilometres and a population of 5,438,657 (December 2020). Norway is bordered by Sweeden, Finland and Russia to the north-east, and the Skagerrak to the south, with Denmark on the other side.

Norway has beautiful mountains, glaciers and stunning fjords. Oslo, the capital, is a city of green spaces and museums. Bergen, with colorful wooden houses, is the starting point for cruises to the dramatic Sognefjord. Norway is also known for fishing, hiking and skiing.

---

## Ubuntu and Lora

### Example

Use the "Ubuntu" font for headings, and "Lora" for text:

# Beautiful Norway

Norway has a total area of 385,252 square kilometres and a population of 5,438,657 (December 2020). Norway is bordered by Sweeden, Finland and Russia to the north-east, and the Skagerrak to the south, with Denmark on the other side.

Norway has beautiful mountains, glaciers and stunning fjords. Oslo, the capital, is a city of green spaces and museums. Bergen, with colorful wooden houses, is the starting point for cruises to the dramatic Sognefjord. Norway is also known for fishing, hiking and skiing.

---

## Abril Fatface and Poppins

### Example

Use the "Abril Fatface" font for headings, and "Poppins" for text:

# Beautiful Norway

Norway has a total area of 385,252 square kilometres and a population of 5,438,657 (December 2020). Norway is bordered by Sweeden, Finland and Russia to the north-east, and the Skagerrak to the south, with Denmark on the other side.

Norway has beautiful mountains, glaciers and stunning fjords. Oslo, the capital, is a city of green spaces and museums. Bergen, with colorful wooden houses, is the starting point for cruises to the dramatic Sognefjord. Norway is also known for fishing, hiking and skiing.

---

## Cinzel and Fauna One

### Example

Use the "Cinzel" font for headings, and "Fauna One" for text:

# Beautiful Norway

Norway has a total area of 385,252 square kilometres and a population of 5,438,657 (December 2020). Norway is bordered by Sweeden, Finland and Russia to the north-east, and the Skagerrak to the south, with Denmark on the other side.

Norway has beautiful mountains, glaciers and stunning fjords. Oslo, the capital, is a city of green spaces and museums. Bergen, with colorful wooden houses, is the starting point for cruises to the dramatic Sognefjord. Norway is also known for fishing, hiking and skiing.

---

## Fjalla One and Libre Baskerville

### Example

Use the "Fjalla One" font for headings, and "Libre Baskerville" for text:

# Beautiful Norway

Norway has a total area of 385,252 square kilometres and a population of 5,438,657 (December 2020). Norway is bordered by Sweeden, Finland and Russia to the north-east, and the Skagerrak to the south, with Denmark on the other side.

Norway has beautiful mountains, glaciers and stunning fjords. Oslo, the capital, is a city of green spaces and museums. Bergen, with colorful wooden houses, is the starting point for cruises to the dramatic Sognefjord. Norway is also known for fishing, hiking and skiing.

---

## Space Mono and Muli

### Example

Use the "Space Mono" font for headings, and "Muli" for text:

# Beautiful Norway

Norway has a total area of 385,252 square kilometres and a population of 5,438,657 (December 2020). Norway is bordered by Sweeden, Finland and Russia to the north-east, and the Skagerrak to the south, with Denmark on the other side.

Norway has beautiful mountains, glaciers and stunning fjords. Oslo, the capital, is a city of green spaces and museums. Bergen, with colorful wooden houses, is the starting point for cruises to the dramatic Sognefjord. Norway is also known for fishing, hiking and skiing.

---

## Spectral and Rubik

### Example

Use the "Spectral" font for headings, and "Rubik" for text:

# Beautiful Norway

Norway has a total area of 385,252 square kilometres and a population of 5,438,657 (December 2020). Norway is bordered by Sweeden, Finland and Russia to the north-east, and the Skagerrak to the south, with Denmark on the other side.

Norway has beautiful mountains, glaciers and stunning fjords. Oslo, the capital, is a city of green spaces and museums. Bergen, with colorful wooden houses, is the starting point for cruises to the dramatic Sognefjord. Norway is also known for fishing, hiking and skiing.

---

## Oswald and Noto Sans

### Example

Use the "Oswald" font for headings, and "Noto Sans" for text:

# Beautiful Norway

Norway has a total area of 385,252 square kilometres and a population of 5,438,657 (December 2020). Norway is bordered by Sweeden, Finland and Russia to the north-east, and the Skagerrak to the south, with Denmark on the other side.

Norway has beautiful mountains, glaciers and stunning fjords. Oslo, the capital, is a city of green spaces and museums. Bergen, with colorful wooden houses, is the starting point for cruises to the dramatic Sognefjord. Norway is also known for fishing, hiking and skiing.

# CSS Font Property

---

## The CSS Font Property

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

The `font` property is a shorthand property for:

- `font-style`
- `font-variant`
- `font-weight`
- `font-size/line-height`
- `font-family`

**Note:** The `font-size` and `font-family` values are required. If one of the other values is missing, their default value are used.

### Example

Use `font` to set several font properties in one declaration:

```
p.a {
  font: 20px Arial, sans-serif;
}

p.b {
  font: italic small-caps bold 12px/30px Georgia, serif;
}
```

## 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 Icons

---

Icons can easily be added to your HTML page, by using an icon library.

---

---

## How To Add Icons

The simplest way to add an icon to your HTML page, is with an icon library, such as Font Awesome.

Add the name of the specified icon class to any inline HTML element (like `<i>` or `<span>`).

All the icons in the icon libraries below, are scalable vectors that can be customized with CSS (size, color, shadow, etc.)

---

## Font Awesome Icons

To use the Font Awesome icons, go to fontawesome.com, sign in, and get a code to add in the `<head>` section of your HTML page:

```
<script src="https://kit.fontawesome.com/yourcode.js"
crossorigin="anonymous"></script>
```

Read more about how to get started with Font Awesome in our Font Awesome 5 tutorial.

**Note:** No downloading or installation is required!

### Example

```
<!DOCTYPE html>
<html>
<head>
<script src="https://kit.fontawesome.com/a076d05399.js" crossorigin="anonymous"></script>
</head>
<body>

<i class="fas fa-cloud"></i>
<i class="fas fa-heart"></i>
<i class="fas fa-car"></i>
<i class="fas fa-file"></i>
<i class="fas fa-bars"></i>

</body>
</html>
```

Result:

For a complete reference of all Font Awesome icons, visit our Icon Reference.

---

---

## Bootstrap Icons

To use the Bootstrap glyphicons, add the following line inside the `<head>` section of your HTML page:

```
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
```

**Note:** No downloading or installation is required!

### Example

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
</head>
<body>

<i class="glyphicon glyphicon-cloud"></i>
<i class="glyphicon glyphicon-remove"></i>
<i class="glyphicon glyphicon-user"></i>
<i class="glyphicon glyphicon-envelope"></i>
<i class="glyphicon glyphicon-thumbs-up"></i>

</body>
</html>
```

Result:

---

## Google Icons

To use the Google icons, add the following line inside the `<head>` section of your HTML page:

```
<link rel="stylesheet"
href="https://fonts.googleapis.com/icon?family=Material+Icons">
```

**Note:** No downloading or installation is required!

### Example

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">
</head>
<body>

<i class="material-icons">cloud</i>
<i class="material-icons">favorite</i>
<i class="material-icons">attachment</i>
<i class="material-icons">computer</i>
<i class="material-icons">traffic</i>
```

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