**CSS**

What is CSS?

* **CSS** stands for **C**ascading **S**tyle **S**heets
* 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 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!

## CSS Syntax

A CSS rule-set consists of a selector and a declaration block:



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.

A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.

## The id Selector

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

The id of an element should be 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.

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

### Example

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

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

In the example below, only <p> elements with class="center" will be center-aligned:

### Example

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

HTML elements can also refer to more than one class.

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

### Example

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

## Grouping Selectors

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

## CSS Comments

A CSS comment starts with /\* and ends with \*/.

Three Ways to Insert CSS

There are three ways of inserting a style sheet:

* External style sheet
* Internal style sheet
* Inline style

## External Style Sheet

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

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

### Example

<head>  
<link rel="stylesheet" type="text/css" href="mystyle.css">  
</head>

An external style sheet can be written in any text editor. The file should not contain any html tags. The style sheet file must be saved with a .css extension.

## Internal Style Sheet

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

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

### Example

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

## Inline Styles

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.

The example below shows how to change the color and the left margin of a <h1> element:

### Example

<h1 style="color:blue;margin-left:30px;">This is a heading</h1>

**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.

### Example

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;      
}

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

an inline style (inside a specific HTML element) has the highest priority, which means that it will override a style defined inside the <head> tag, or in an external style sheet.

# CSS Colors

Colors in CSS are most often specified by:

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

HTML and CSS supports [140 standard color names](https://www.w3schools.com/colors/colors_names.asp).

## RGB (Red, Green, Blue)

RGB color values can be specified using this formula: rgb(red, green, blue).

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

## Hexadecimal Colors

RGB values can also be specified using **hexadecimal** color values in the form: #RRGGBB, where RR (red), GG (green) and BB (blue) are hexadecimal values between 00 and FF (same as decimal 0-255).

# CSS Backgrounds

The CSS background properties are used to define the background effects for elements.

CSS background properties:

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

## 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.

The background image for a page can be set like this:

### Example

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

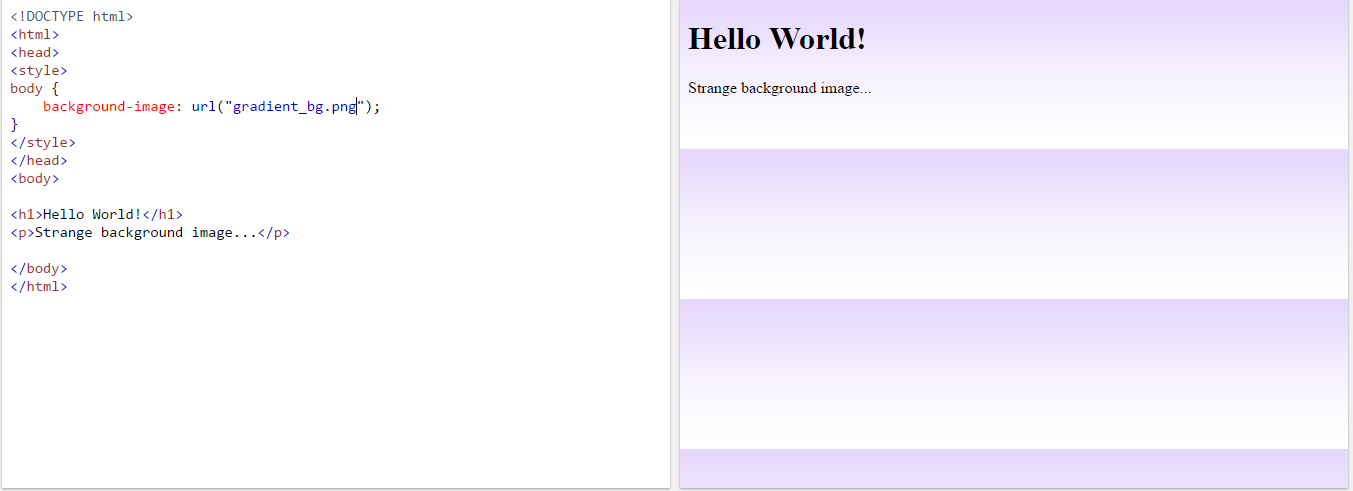
## Background Image - Repeat Horizontally or Vertically

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

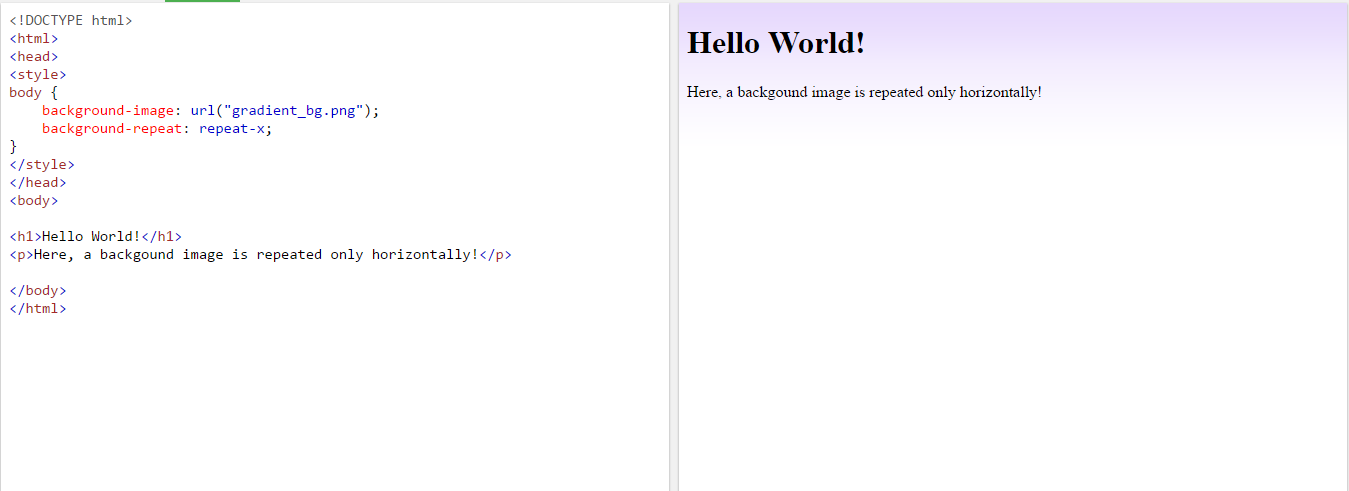
body {  
    background-image: url("gradient\_bg.png");  
}



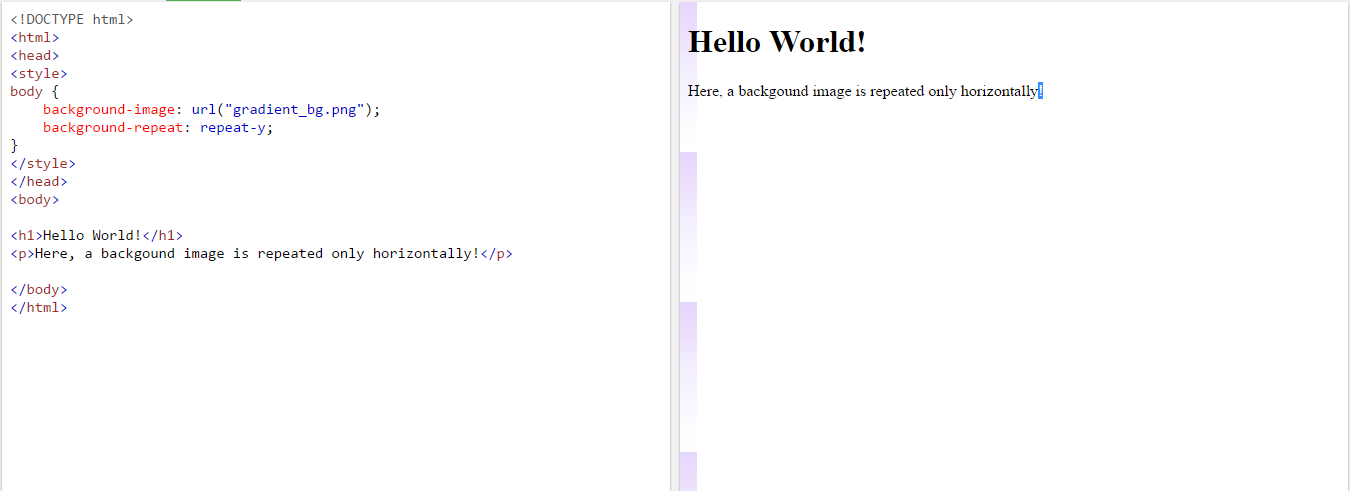
If the image above is repeated only horizontally (background-repeat: repeat-x;), the background will look better:

### Example

body {  
    background-image: url("gradient\_bg.png");  
    background-repeat: repeat-x;  
}



vertically (background-repeat: repeat-y;),

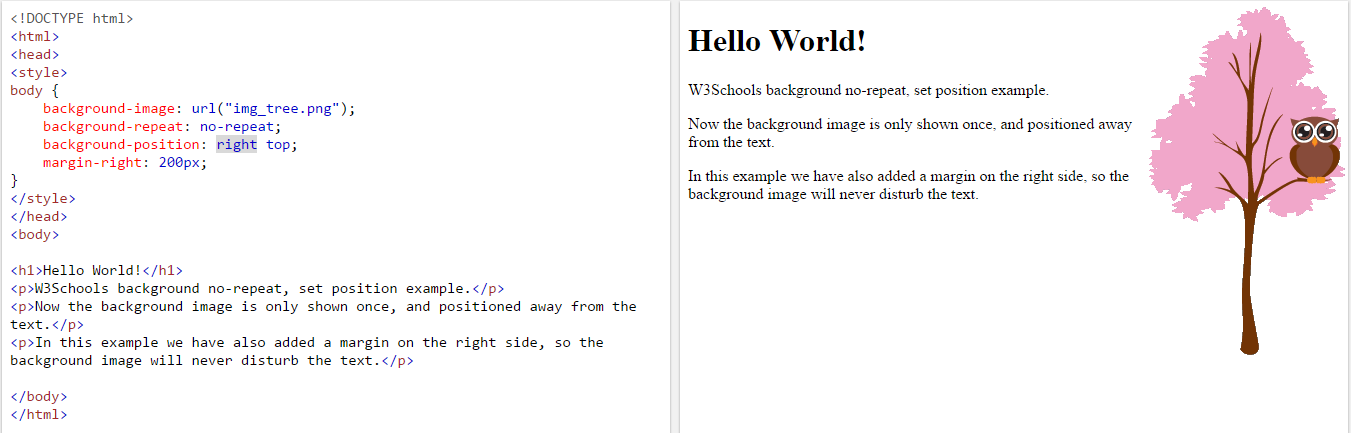


## Background Image - Set position and no-repeat

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

### Example

body {  
    background-image: url("img\_tree.png");  
    background-repeat: no-repeat;  
}

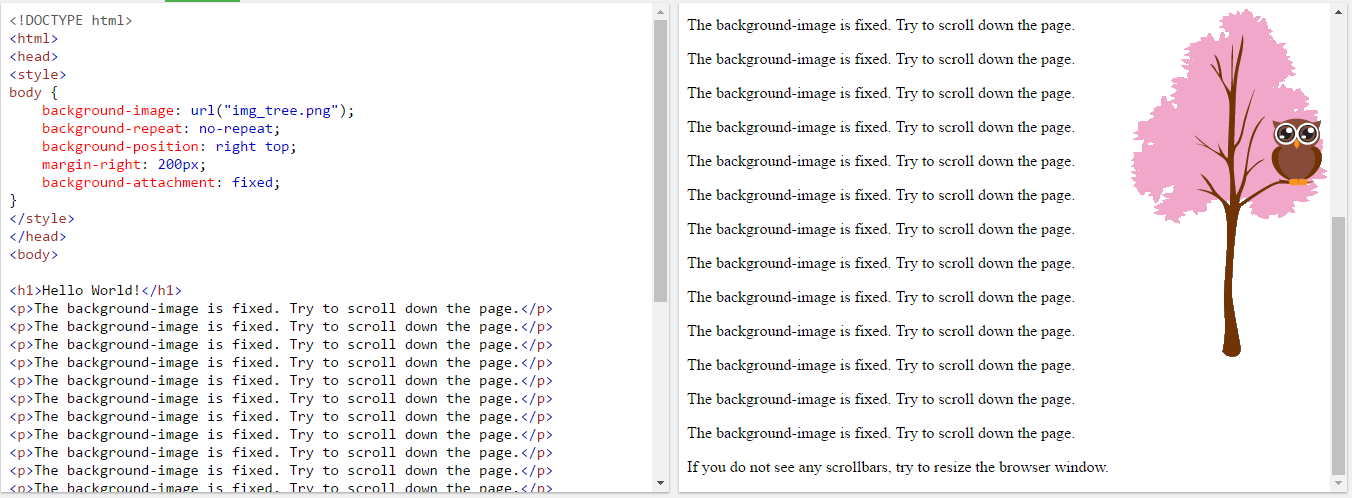
The position of the image is specified by the background-position property: 

## Background Image - Fixed position

To specify that the background image should be fixed (will not scroll with the rest of the page), use the background-attachment property:

### Example

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



## 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.

The shorthand property for background is background:

### Example

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.

## CSS Border Properties

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

## 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

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.

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.

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;  
}  
  
p.two {  
    border-style: solid;  
    border-width: medium;  
}  
  
p.three {  
    border-style: solid;  
    border-width: 2px 10px 4px 20px;  
}

## 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)"
* transparent

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

If border-color is not set, it inherits the color of the element.

### Example

p.one {  
    border-style: solid;  
    border-color: red;  
}  
  
p.two {  
    border-style: solid;  
    border-color: green;  
}  
  
p.three {  
    border-style: solid;  
    border-color: red green blue yellow;  
}

## Border - Individual Sides

From the examples above you have seen that it is possible to specify a different border for each side.

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

Different Border Styles

### Example

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

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

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

## Border - Shorthand Property

As you can see from the examples above, 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;  
}

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

### Left Border

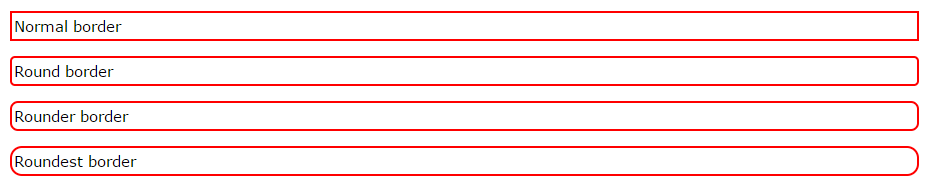
p {  
    border-left: 6px solid red;  
    background-color: lightgrey;  
}

Result:

Some text

## Rounded Borders

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



### Example

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

## CSS Margins

The CSS margin properties are used to generate space around elements.

The margin properties set the size of the white space outside the border.

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.

## 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

### Example

p {  
    margin: 100px 150px 100px 80px;  
}

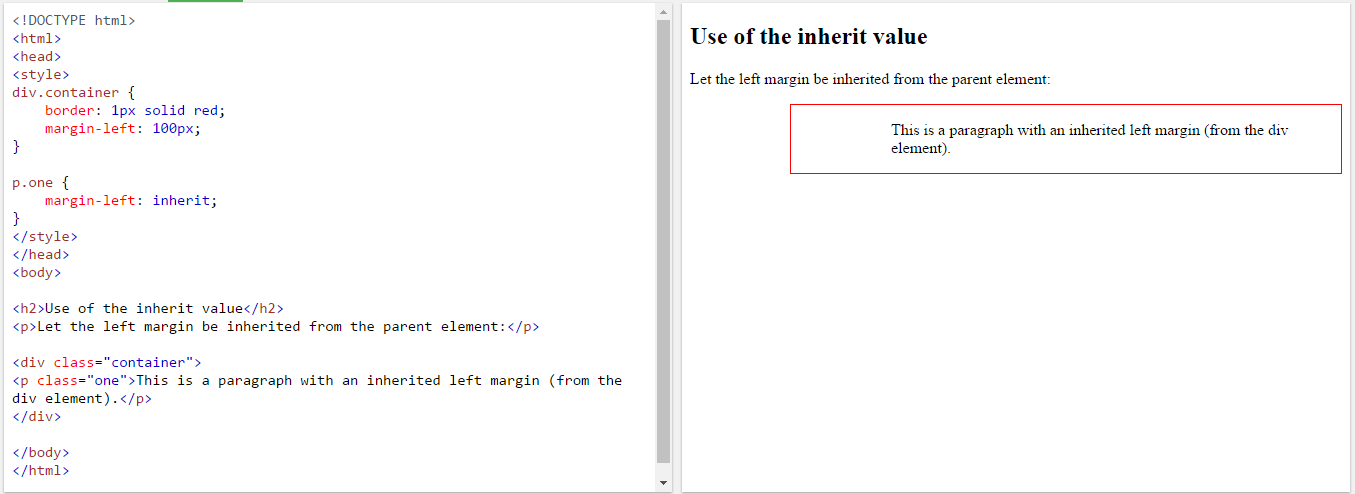
## The auto Value

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.

## The inherit Value

This example lets the left margin be inherited from the parent element

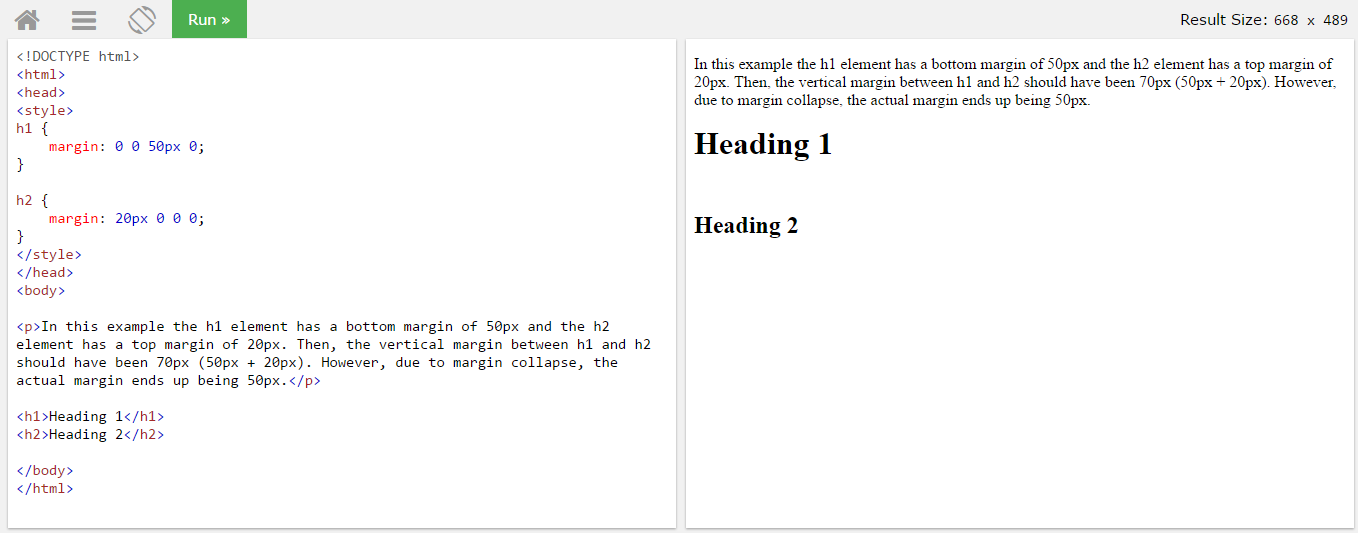


## Margin Collapse

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:

In the example above, the <h1> element has a bottom margin of 50px. 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.

## CSS Padding

The CSS padding properties are used to generate space around content.

The padding clears an area around the content (inside the border) of an element.

With CSS, you have full control over the padding. There are CSS 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

## 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

# CSS Height and Width

## Setting height and width

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

The height and width can be set to auto (this is default. Means that the browser calculates the height and width), or be specified in length values, like px, cm, etc., or in percent (%) of the containing block.

**Note:** 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).

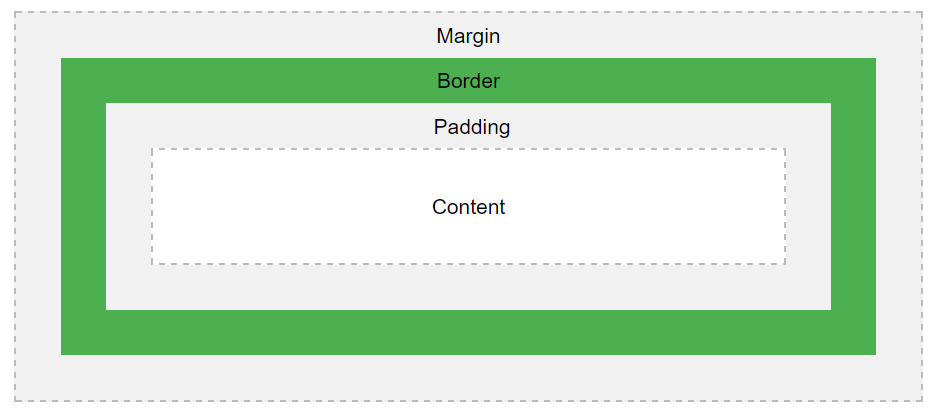
The problem with the <div> above occurs when the browser window is smaller than the width of the element (500px). The browser then adds a horizontal scrollbar to the page.

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

## The CSS Box Model

All HTML elements can be considered as boxes. 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

## 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.

Assume we want to style a <div> element to have a total width of 350px:

### Example

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

Here is the math:

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:

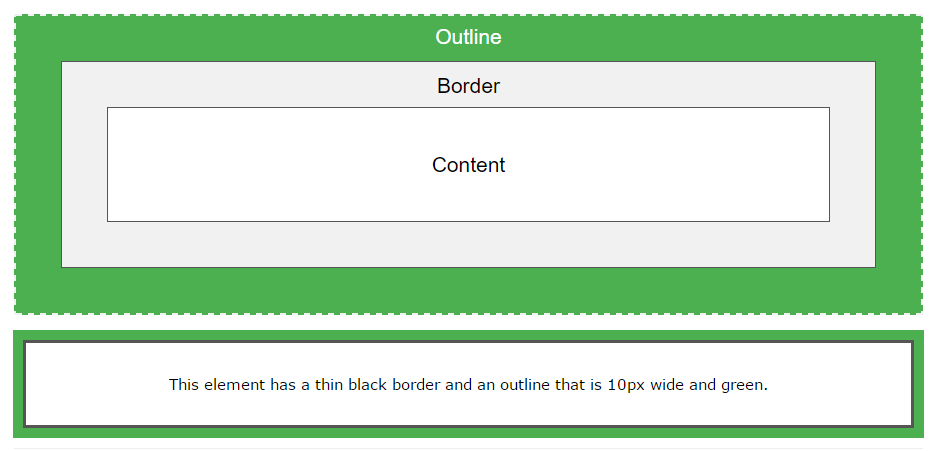
Total element height = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin.

## CSS Outline

The CSS outline properties specify the style, color, and width of an outline.

An outline is a line that is drawn around elements (outside the borders) to make the element "stand out".

However, the outline property is different from the border property - The outline is NOT a part of an element's dimensions; the element's total width and height is not affected by the width of the outline.

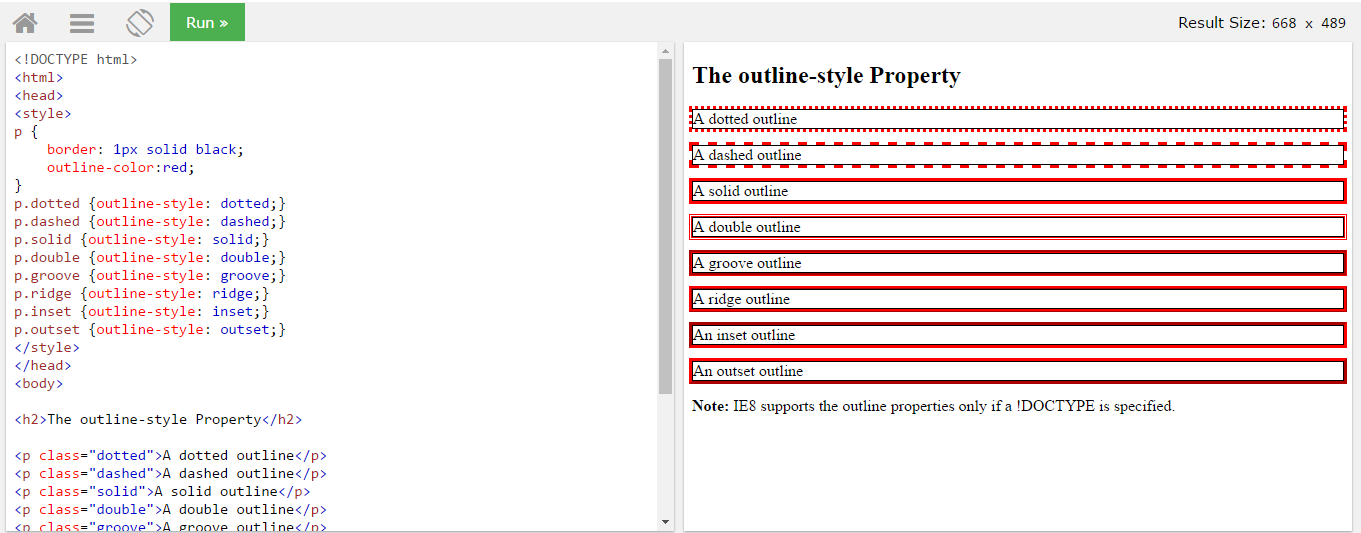


## Outline Style

The outline-style property specifies the style of the outline.

The outline-style property 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. The effect depends on the outline-color value
* ridge - Defines a 3D ridged outline. The effect depends on the outline-color value
* inset - Defines a 3D inset outline. The effect depends on the outline-color value
* outset - Defines a 3D outset outline. The effect depends on the outline-color value
* none - Defines no outline
* hidden - Defines a hidden outline



## Outline Color

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

## Outline Width

The outline-width property specifies the width of the outline.

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.

## Outline - Shorthand property

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

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

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

# CSS Text

## Text Color

The color property is used to set the color of the text.

With CSS, a color is most often specified by:

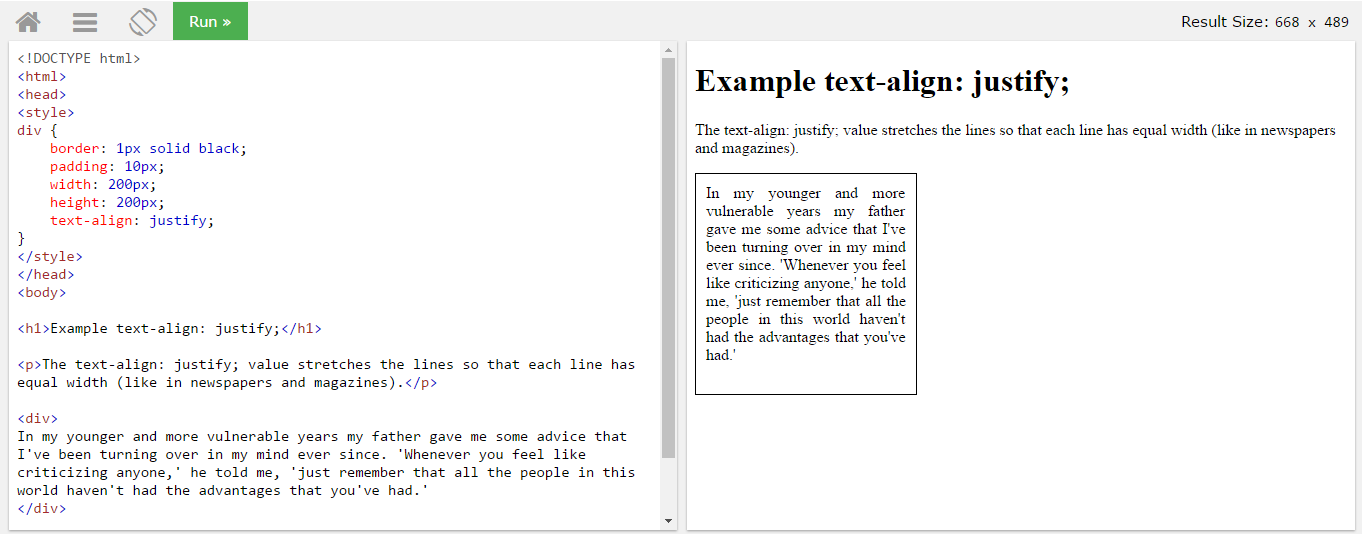
* a color name - like "red"
* a HEX value - like "#ff0000"
* an RGB value - like "rgb(255,0,0)"

## 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.

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):



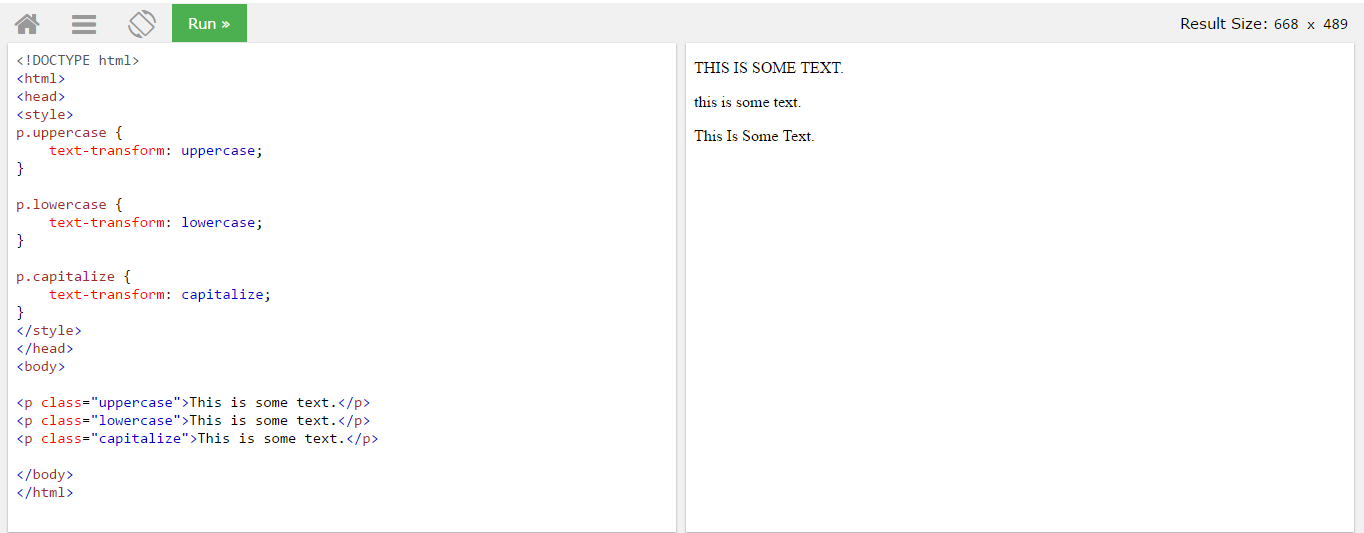
## Text Decoration

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.

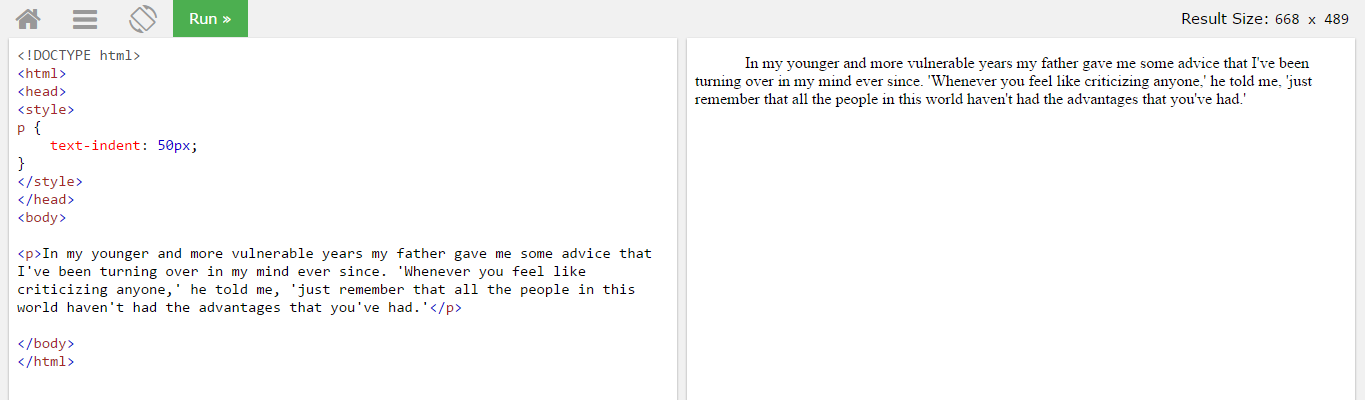
## Text Transformation

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: 

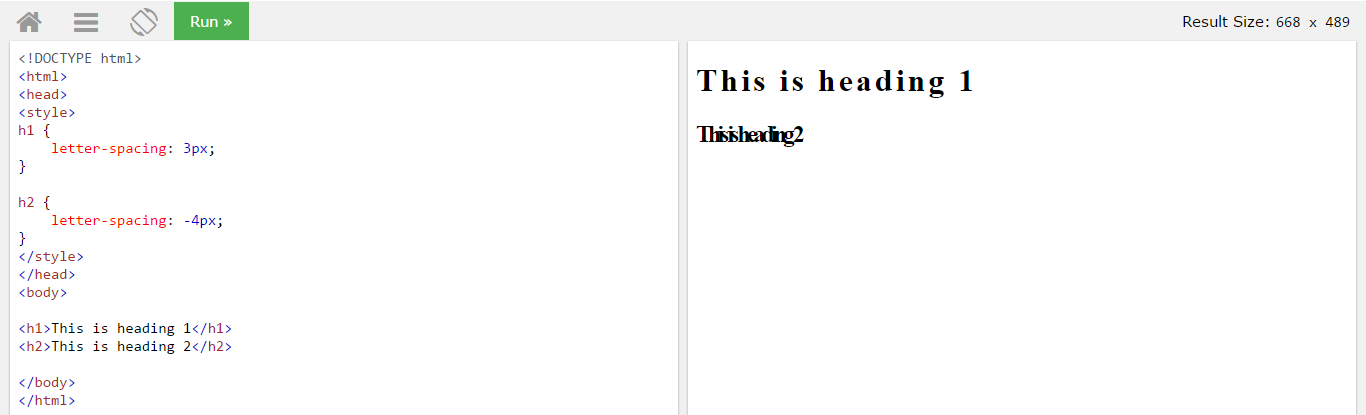
## Text Indentation

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



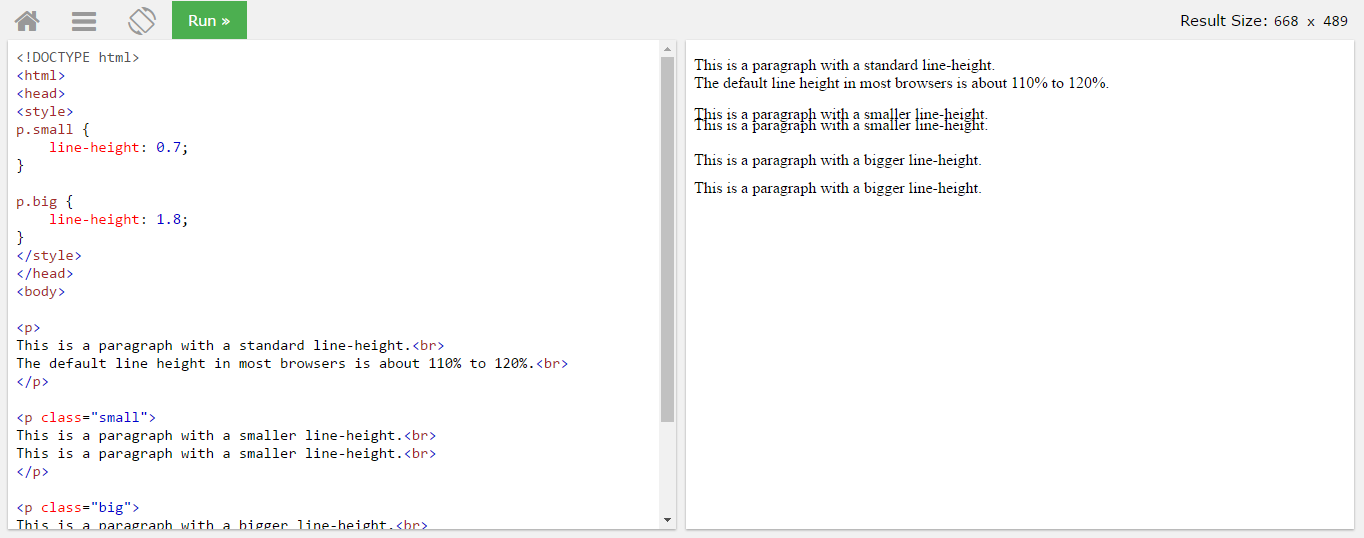
## Letter Spacing

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



## Line Height

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

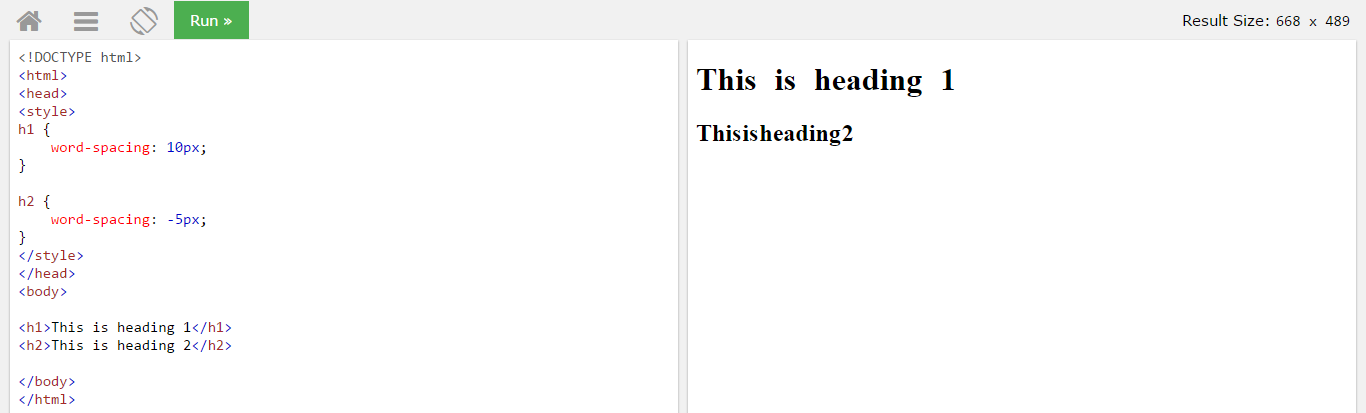


## Text Direction

The direction property is used to change the text direction of an element:

## Word Spacing

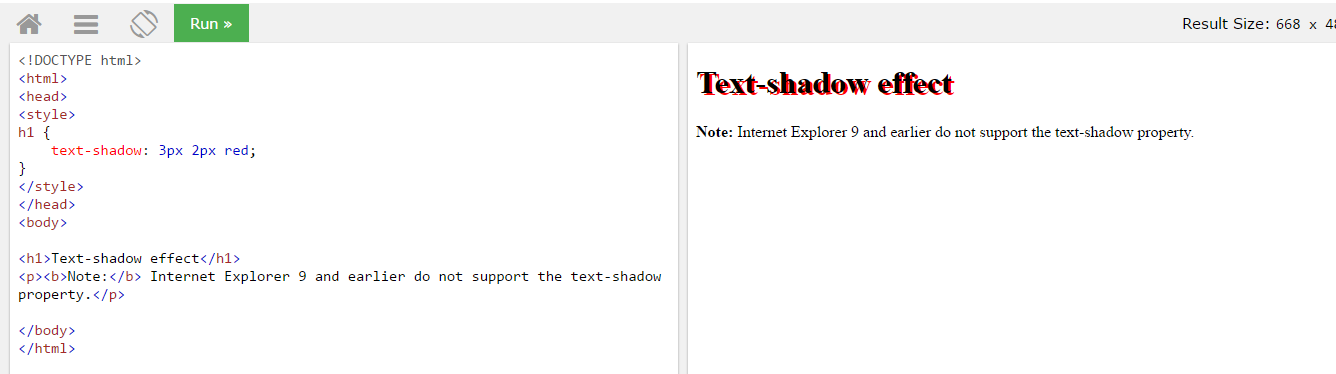
The word-spacing property is used to specify the space between the words in a text.



## Text Shadow

The text-shadow property adds shadow to text.

The following example specifies the position of the horizontal shadow (3px), the position of the vertical shadow (2px) and the color of the shadow (red):

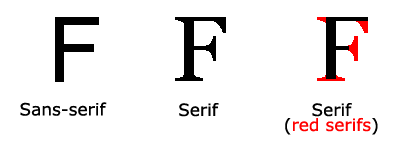


## All CSS Text Properties

|  |  |
| --- | --- |
| **Property** | **Description** |
| [color](https://www.w3schools.com/cssref/pr_text_color.asp) | Sets the color of text |
| [direction](https://www.w3schools.com/cssref/pr_text_direction.asp) | Specifies the text direction/writing direction |
| [letter-spacing](https://www.w3schools.com/cssref/pr_text_letter-spacing.asp) | Increases or decreases the space between characters in a text |
| [line-height](https://www.w3schools.com/cssref/pr_dim_line-height.asp) | Sets the line height |
| [text-align](https://www.w3schools.com/cssref/pr_text_text-align.asp) | Specifies the horizontal alignment of text |
| [text-decoration](https://www.w3schools.com/cssref/pr_text_text-decoration.asp) | Specifies the decoration added to text |
| [text-indent](https://www.w3schools.com/cssref/pr_text_text-indent.asp) | Specifies the indentation of the first line in a text-block |
| [text-shadow](https://www.w3schools.com/cssref/css3_pr_text-shadow.asp) | Specifies the shadow effect added to text |
| [text-transform](https://www.w3schools.com/cssref/pr_text_text-transform.asp) | Controls the capitalization of text |
| [text-overflow](https://www.w3schools.com/cssref/css3_pr_text-overflow.asp) | Specifies how overflowed content that is not displayed should be signaled to  the user |
| [unicode-bidi](https://www.w3schools.com/cssref/pr_text_unicode-bidi.asp) | Used together with the [direction](https://www.w3schools.com/cssref/pr_text_direction.asp) property to set or return whether the text should be  overridden to support multiple languages in the same document |
| [vertical-align](https://www.w3schools.com/cssref/pr_pos_vertical-align.asp) | Sets the vertical alignment of an element |
| [white-space](https://www.w3schools.com/cssref/pr_text_white-space.asp) | Specifies how white-space inside an element is handled |
| [word-spacing](https://www.w3schools.com/cssref/pr_text_word-spacing.asp) | Increases or decreases the space between words in a text |

# CSS Fonts

Difference Between Serif and Sans-serif Fonts



## CSS Font Families

In CSS, there are two types of font family names:

* **generic family** - a group of font families with a similar look (like "Serif" or "Monospace")
* **font family** - a specific font family (like "Times New Roman" or "Arial")

|  |  |  |
| --- | --- | --- |
| **Generic family** | **Font family** | **Description** |
| Serif | Times New Roman Georgia | Serif fonts have small lines at the ends  on some characters |
| Sans-serif | Arial Verdana | "Sans" means without - these fonts do not have the lines at the ends of characters |
| Monospace | Courier New Lucida Console | All monospace characters have the  same width |

## Font Family

The font family of a text is set with the font-family property.

The font-family property should hold several font names as a "fallback" system. If the browser does not support the first font, it tries the next font, and so on.

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.

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

More than one font family is specified in a comma-separated list:

### Example

p {  
    font-family: "Times New Roman", Times, serif;  
}

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)



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 Em

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

The em size unit is recommended by the W3C.

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 \*/}

## 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:

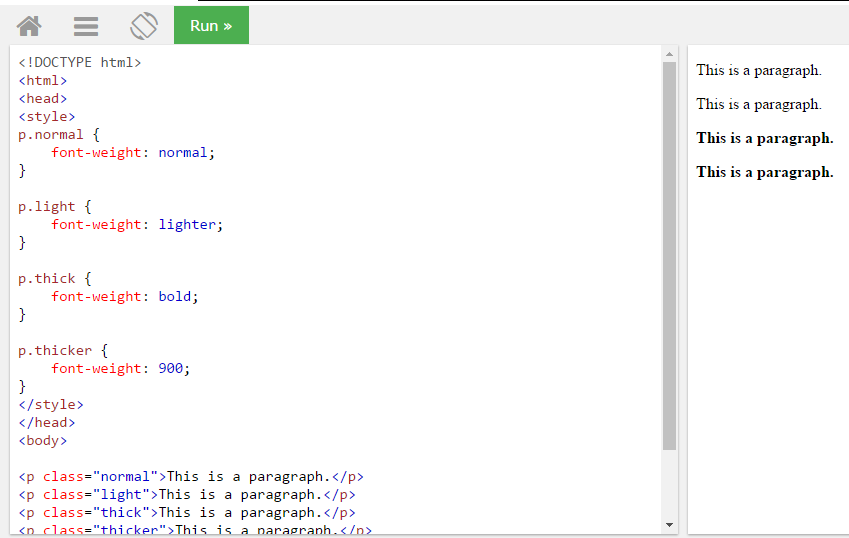
### Example

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!

Font Weight

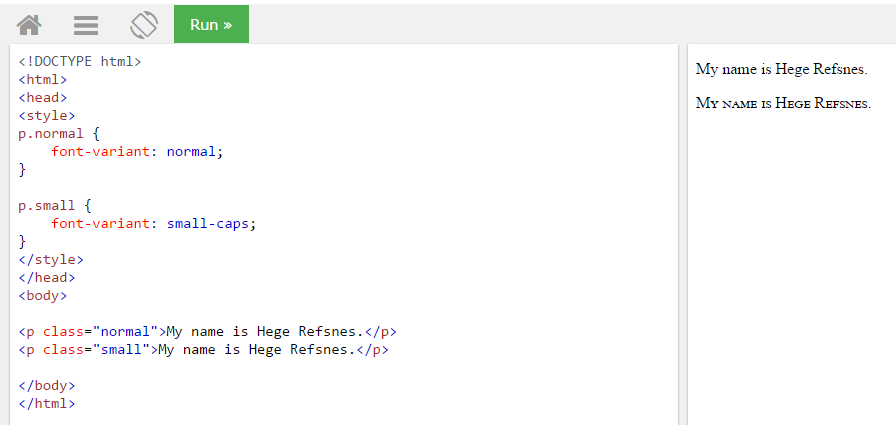
The font-weight property specifies the weight of a font:



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.



## All CSS Font Properties

|  |  |
| --- | --- |
| **Property** | **Description** |
| [font](https://www.w3schools.com/cssref/pr_font_font.asp) | Sets all the font properties in one declaration |
| [font-family](https://www.w3schools.com/cssref/pr_font_font-family.asp) | Specifies the font family for text |
| [font-size](https://www.w3schools.com/cssref/pr_font_font-size.asp) | Specifies the font size of text |
| [font-style](https://www.w3schools.com/cssref/pr_font_font-style.asp) | Specifies the font style for text |
| [font-variant](https://www.w3schools.com/cssref/pr_font_font-variant.asp) | Specifies whether or not a text should be displayed in a small-caps font |
| [font-weight](https://www.w3schools.com/cssref/pr_font_weight.asp) | Specifies the weight of a font |

# CSS Icons

How To Add Icons

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

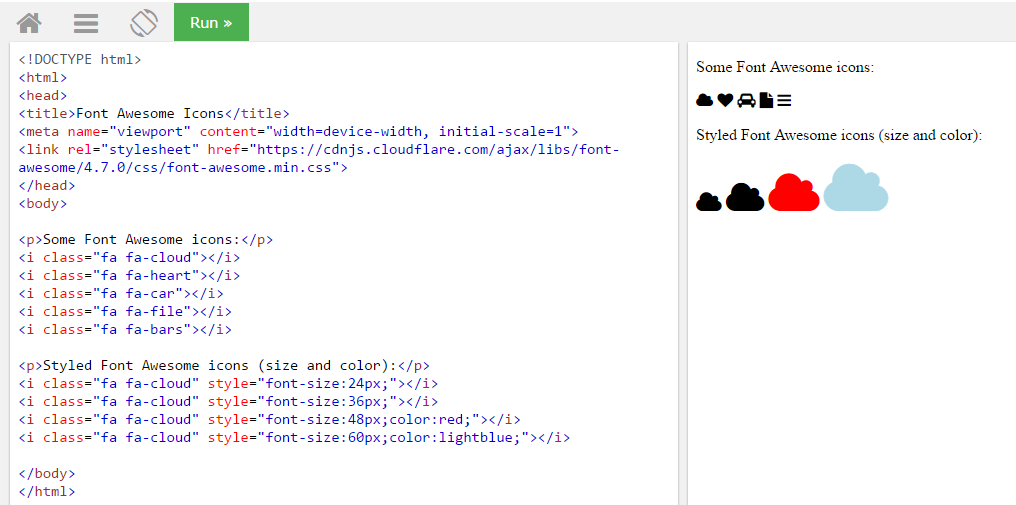
Add the name of the specified icon class to any inline HTML element (like <i> or <span>).

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, add the following line inside the <head> section of your HTML page:

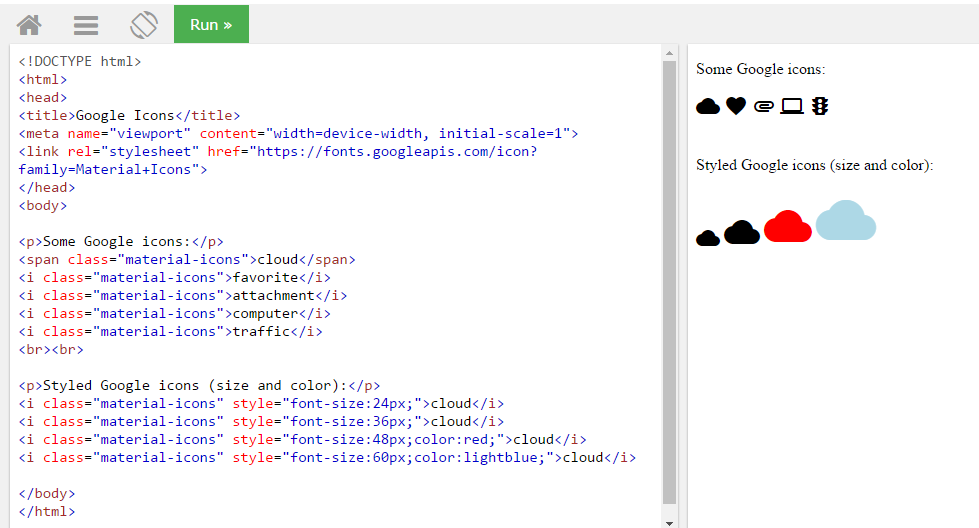
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">



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">



# CSS Links

With CSS, links can be styled in different ways.

[Text Link](javascript:void(0))  [Text Link](javascript:void(0))  [Link Button](javascript:void(0))  [Link Button](javascript:void(0))

Styling Links

Links can be styled with any CSS property (e.g. color, font-family, background, etc.).

In addition, links can be styled differently depending on what **state** they are in.

The four links states are:

* a:link - a normal, unvisited link
* a:visited - a link the user has visited
* a:hover - a link when the user mouses over it
* a:active - a link the moment it is clicked

### Example

/\* unvisited link \*/  
a:link {  
    color: red;  
}  
  
/\* visited link \*/  
a:visited {  
    color: green;  
}  
  
/\* mouse over link \*/  
a:hover {  
    color: hotpink;  
}  
  
/\* selected link \*/  
a:active {  
    color: blue;  
}

When setting the style for several link states, there are some order rules:

* a:hover MUST come after a:link and a:visited
* a:active MUST come after a:hover

## Text Decoration

The text-decoration property is mostly used to remove underlines from links:

### Example

a:link {  
    text-decoration: none;  
}  
  
a:visited {  
    text-decoration: none;  
}  
  
a:hover {  
    text-decoration: underline;  
}  
  
a:active {  
    text-decoration: underline;  
}

## Background Color

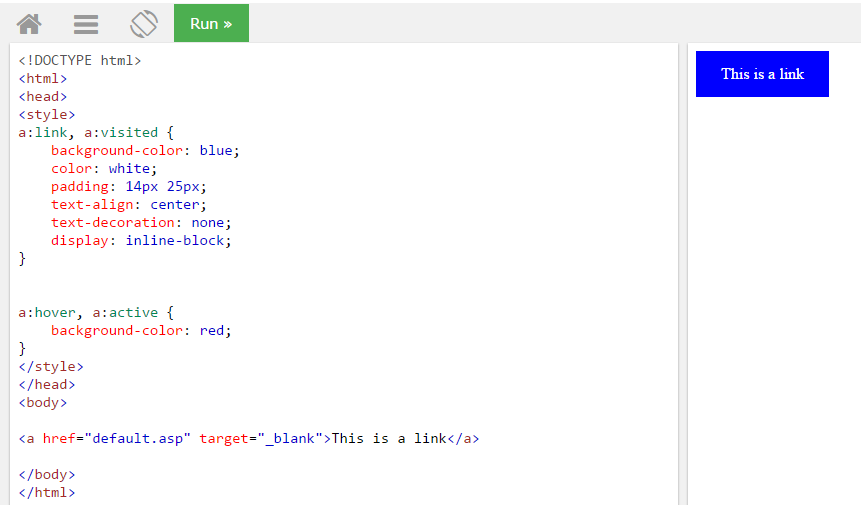
The background-color property can be used to specify a background color for links:

### Example

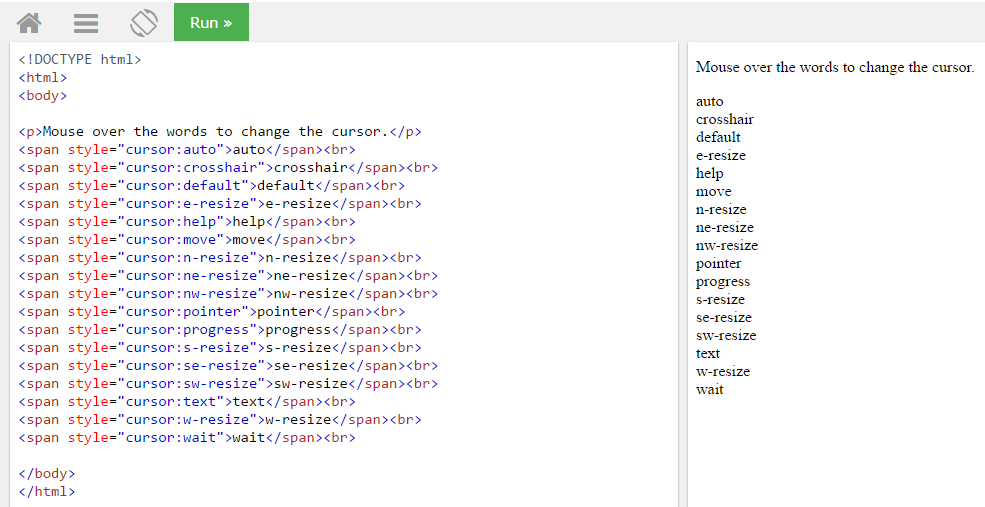
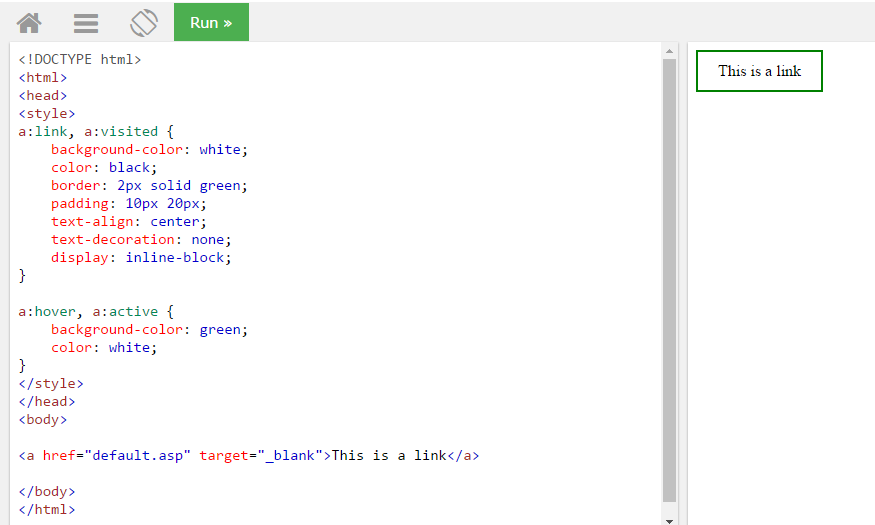
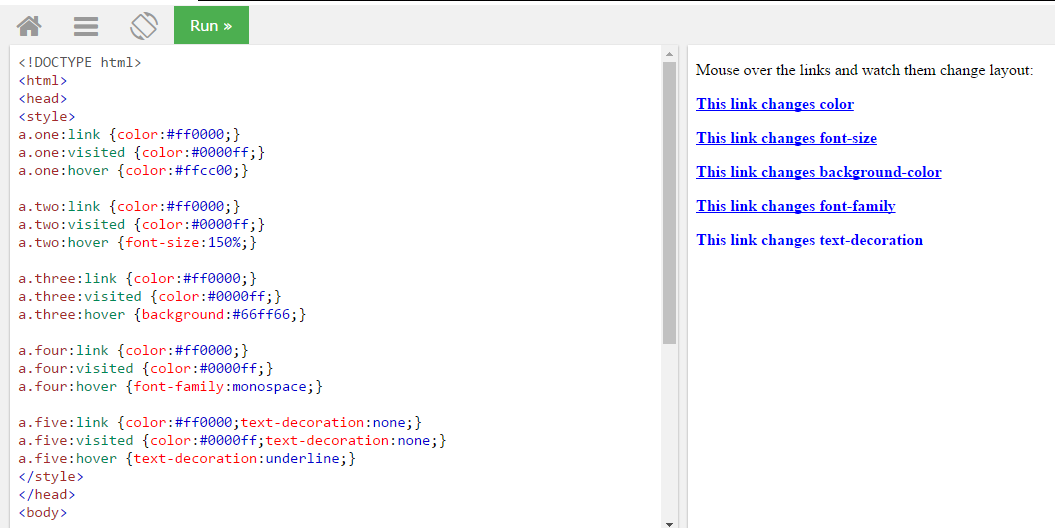
a:link {  
    background-color: yellow;  
}

Advanced - Link Buttons

This example demonstrates a more advanced example where we combine several CSS properties to display links as boxes/buttons:



More Examples



# CSS Lists

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

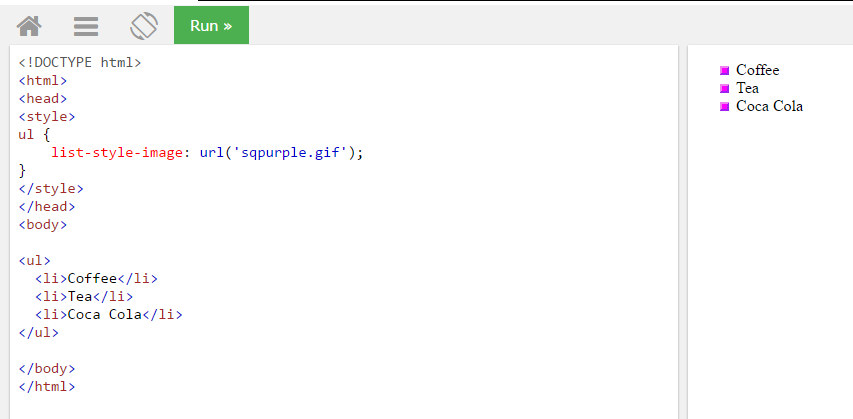
## Different List Item Markers

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

### Example

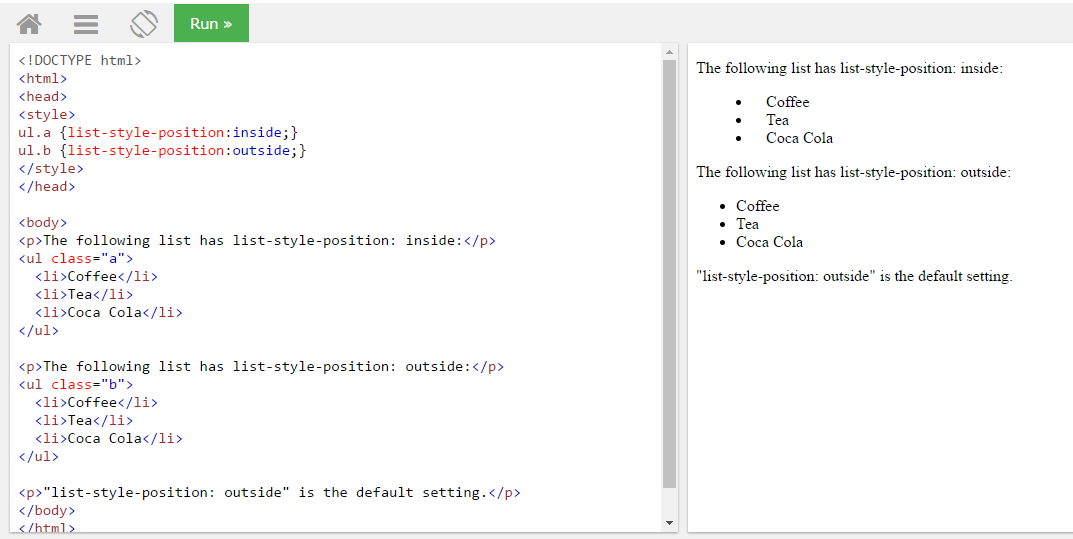
ul.a {  
    list-style-type: circle;  
}  
  
ul.b {  
    list-style-type: square;  
}  
  
ol.c {  
    list-style-type: upper-roman;  
}

An Image as The List Item Marker



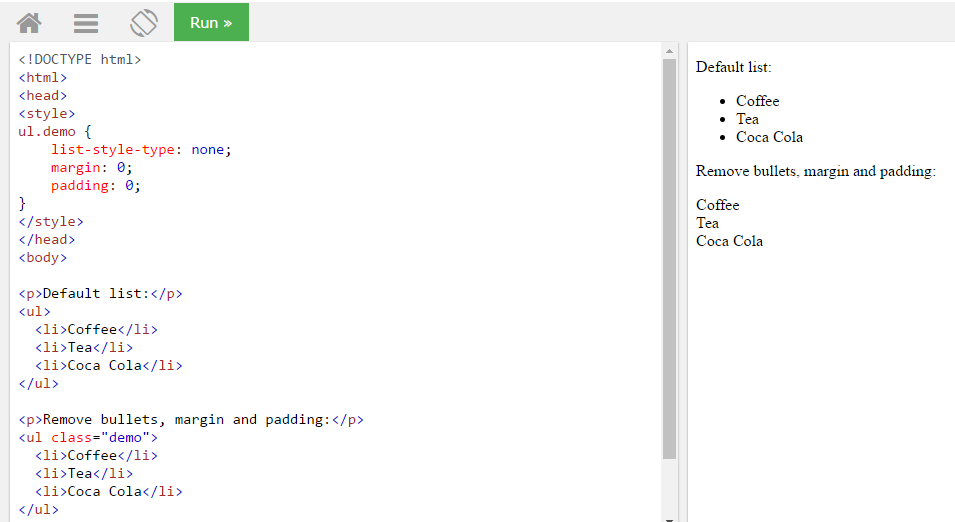
Position The List Item Markers

The list-style-position property specifies whether the list-item markers should appear inside or outside the content flow:



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>:



## List - Shorthand property

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

### Example

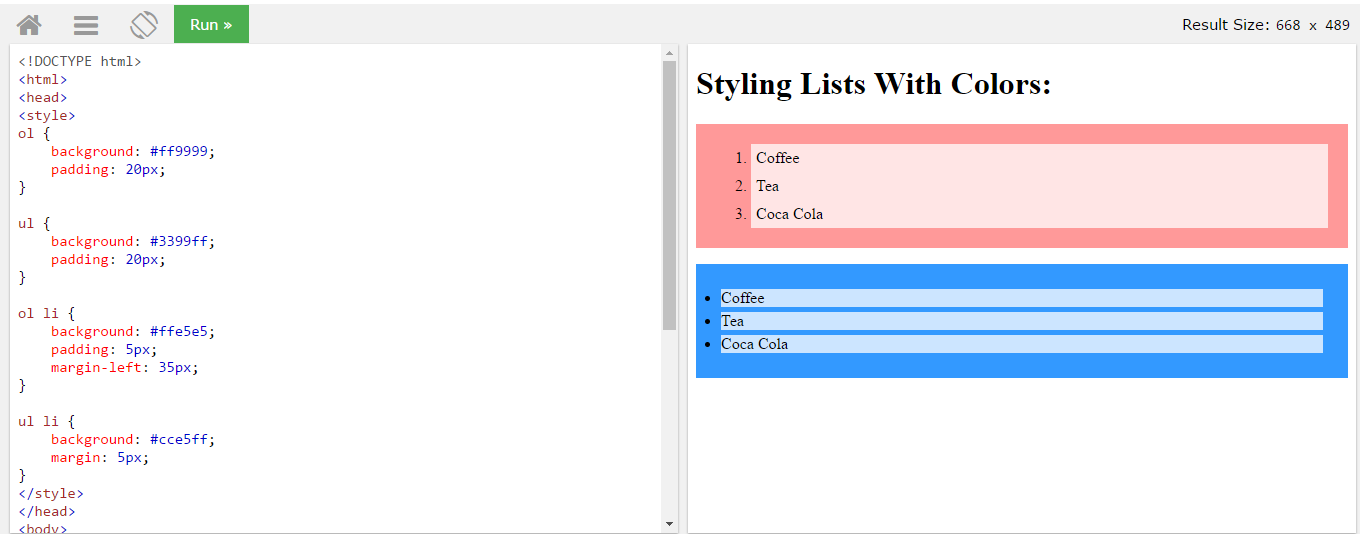
ul {  
    list-style: square inside url("sqpurple.gif");  
}

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

* 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)

Styling List With Colors

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



## All CSS List Properties

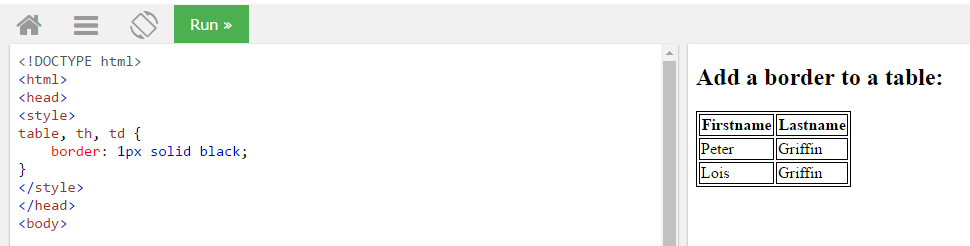
|  |  |
| --- | --- |
| **Property** | **Description** |
| [list-style](https://www.w3schools.com/cssref/pr_list-style.asp) | Sets all the properties for a list in one declaration |
| [list-style-image](https://www.w3schools.com/cssref/pr_list-style-image.asp) | Specifies an image as the list-item marker |
| [list-style-position](https://www.w3schools.com/cssref/pr_list-style-position.asp) | Specifies if the list-item markers should appear inside or outside the content flow |
| [list-style-type](https://www.w3schools.com/cssref/pr_list-style-type.asp) | Specifies the type of list-item marker |

# CSS Tables

Table Borders

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

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



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

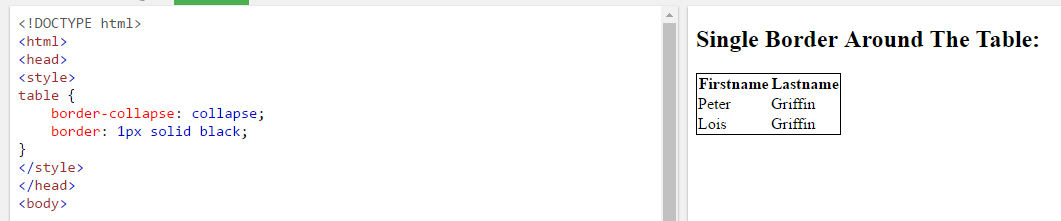
Collapse Table Borders

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

### Example

table {  
    border-collapse: collapse;  
}  
  
table, th, td {  
    border: 1px solid black;  
}

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



## Table Width and Height

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

## 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.

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:

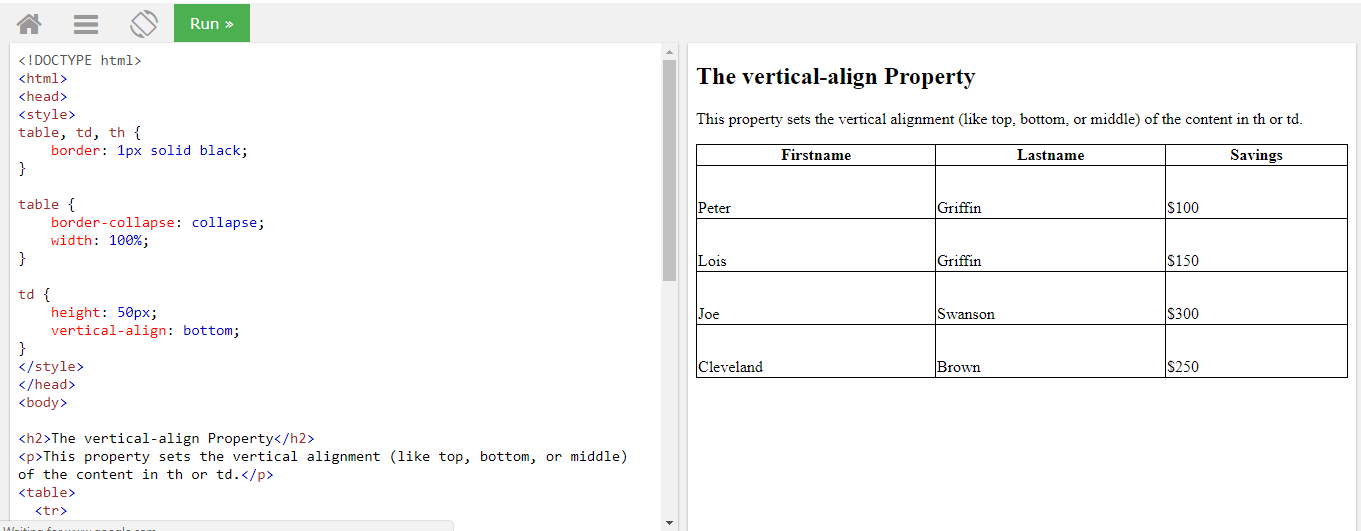


Table Padding

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



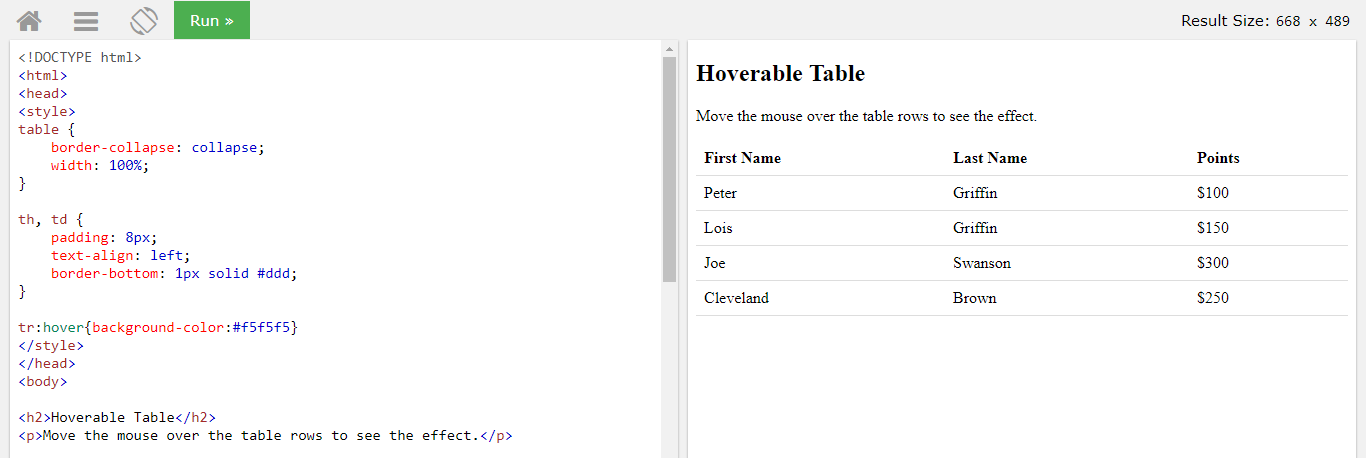
Horizontal Dividers

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



Hoverable Table

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



Striped Tables

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

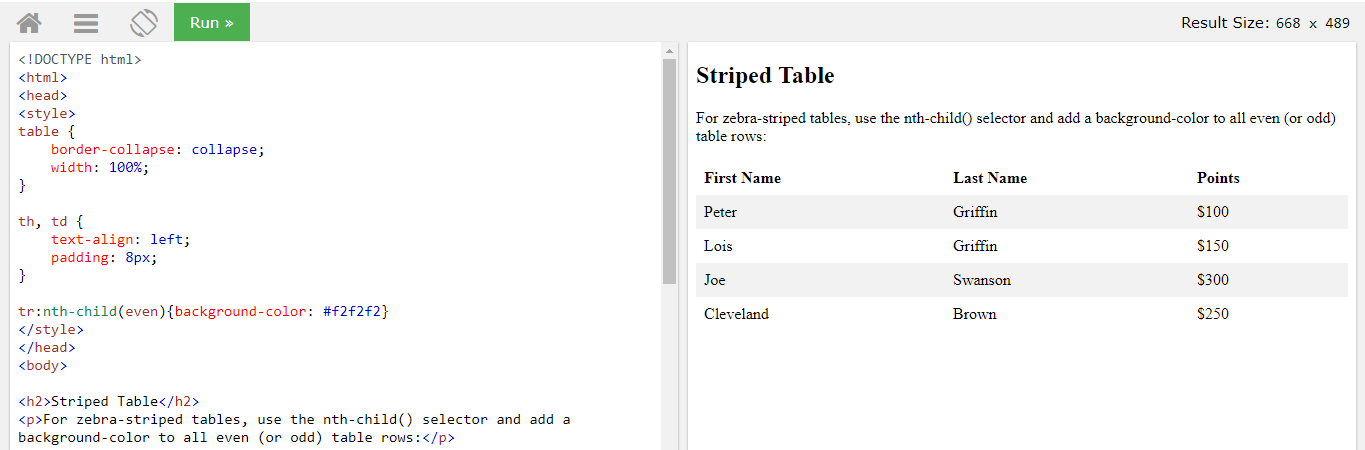
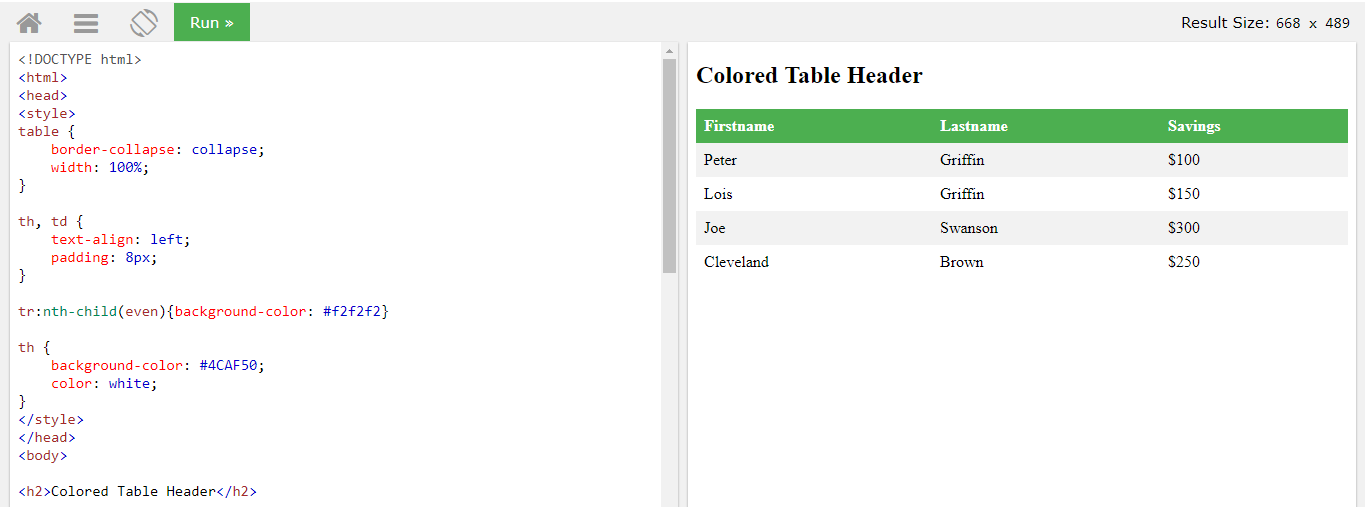


Table Color

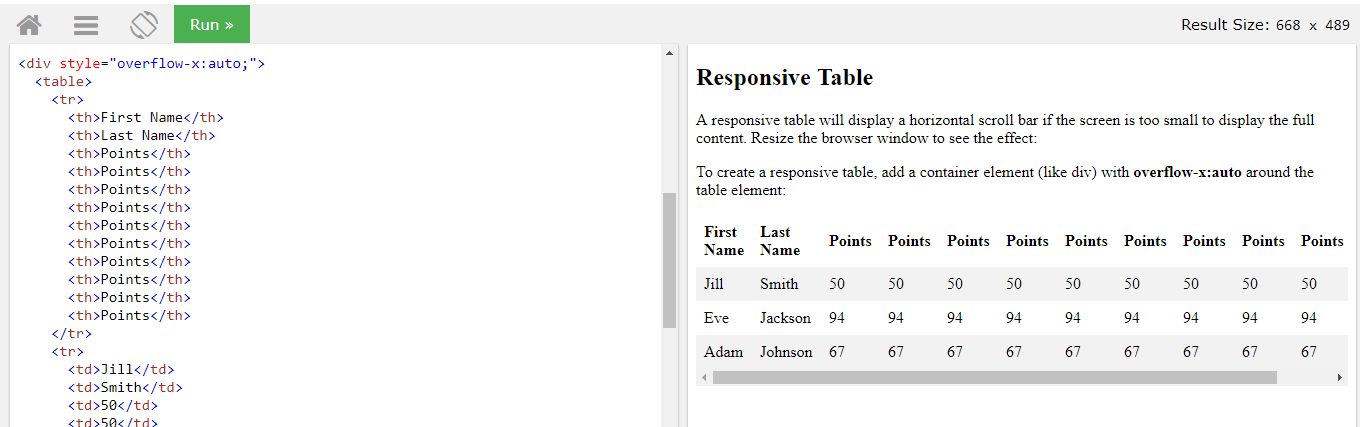
The example below specifies the background color and text color of <th> elements:



Responsive Table

A responsive table will display a horizontal scroll bar if the screen is too small to display the full content:

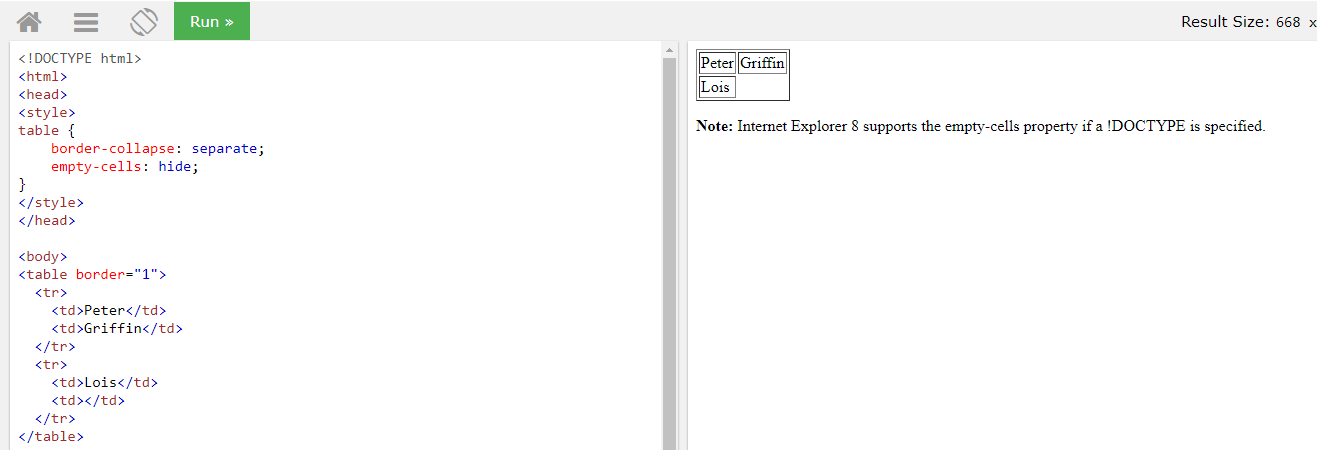
Add a container element (like <div>) with overflow-x:auto around the <table> element to make it responsive:



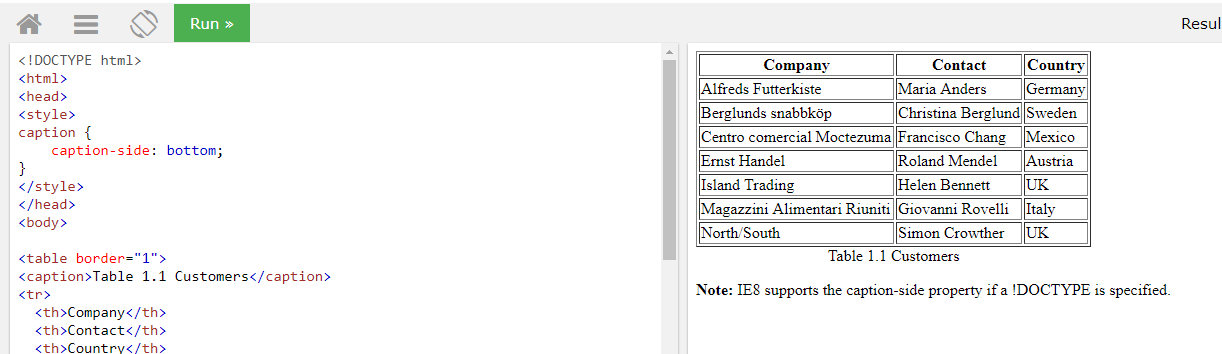
## CSS Table Properties

|  |  |
| --- | --- |
| **Property** | **Description** |
| [border](https://www.w3schools.com/cssref/pr_border.asp) | Sets all the border properties in one declaration |
| [border-collapse](https://www.w3schools.com/cssref/pr_border-collapse.asp) | Specifies whether or not table borders should be collapsed |
| [border-spacing](https://www.w3schools.com/cssref/pr_border-spacing.asp) | Specifies the distance between the borders of adjacent cells |
| [caption-side](https://www.w3schools.com/cssref/pr_tab_caption-side.asp) | Specifies the placement of a table caption |
| [empty-cells](https://www.w3schools.com/cssref/pr_tab_empty-cells.asp) | Specifies whether or not to display borders and background on empty cells in a table |
| [table-layout](https://www.w3schools.com/cssref/pr_tab_table-layout.asp) | Sets the layout algorithm to be used for a table |

Empty-cells:



# caption-side Property



# CSS Layout - The display Property

The display property is the most important CSS property for controlling layout.

The display Property

The display property specifies if/how an element is displayed.

Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is block or inline.

Block-level Elements

A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can).

The <div> element is a block-level element.

Examples of block-level elements:

* <div>
* <h1> - <h6>
* <p>
* <form>
* <header>
* <footer>
* <section>

Inline Elements

An inline element does not start on a new line and only takes up as much width as necessary.

This is an inline <span> element inside a paragraph.

Examples of inline elements:

* <span>
* <a>
* <img>

Display: none;

display: none; is commonly used with JavaScript to hide and show elements without deleting and recreating them.

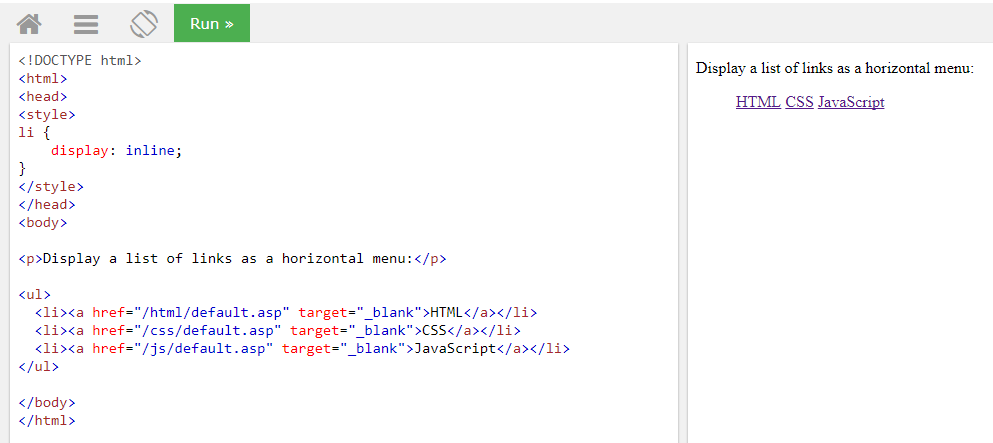
The <script> element uses display: none; as default.

## Override The Default Display Value

As mentioned, every element has a default display value. However, you can override this.

Changing an inline element to a block element, or vice versa, can be useful for making the page look a specific way, and still follow the web standards.

A common example is making inline <li> elements for horizontal menus:



**Note:** Setting the display property of an element only changes **how the element is displayed**, NOT what kind of element it is. So, an inline element with display: block; is not allowed to have other block elements inside it.

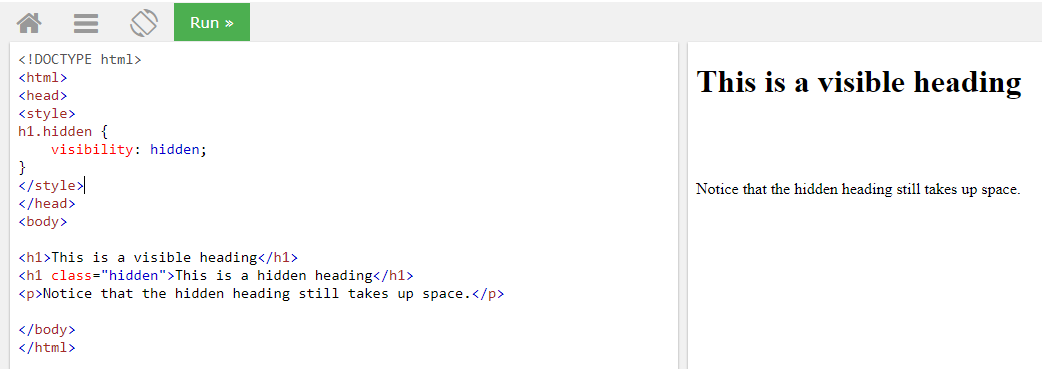
## Hide an Element - display:none or visibility:hidden?

Hiding an element can be done by setting the display property to none. The element will be hidden, and the page will be displayed as if the element is not there:

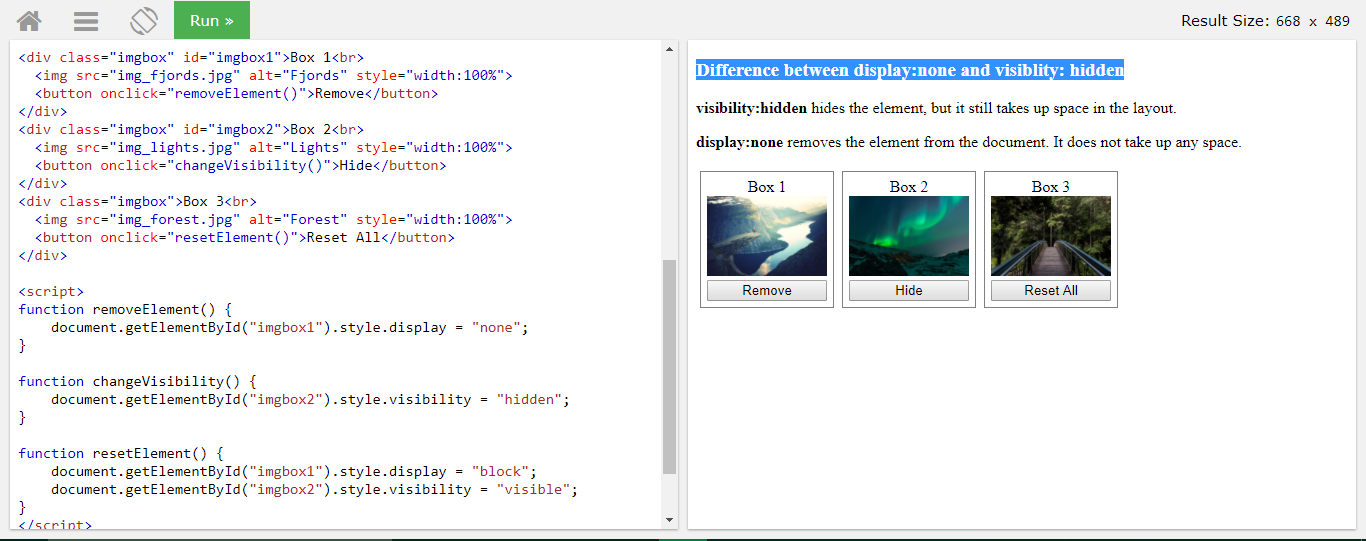


visibility:hidden; also hides an element.

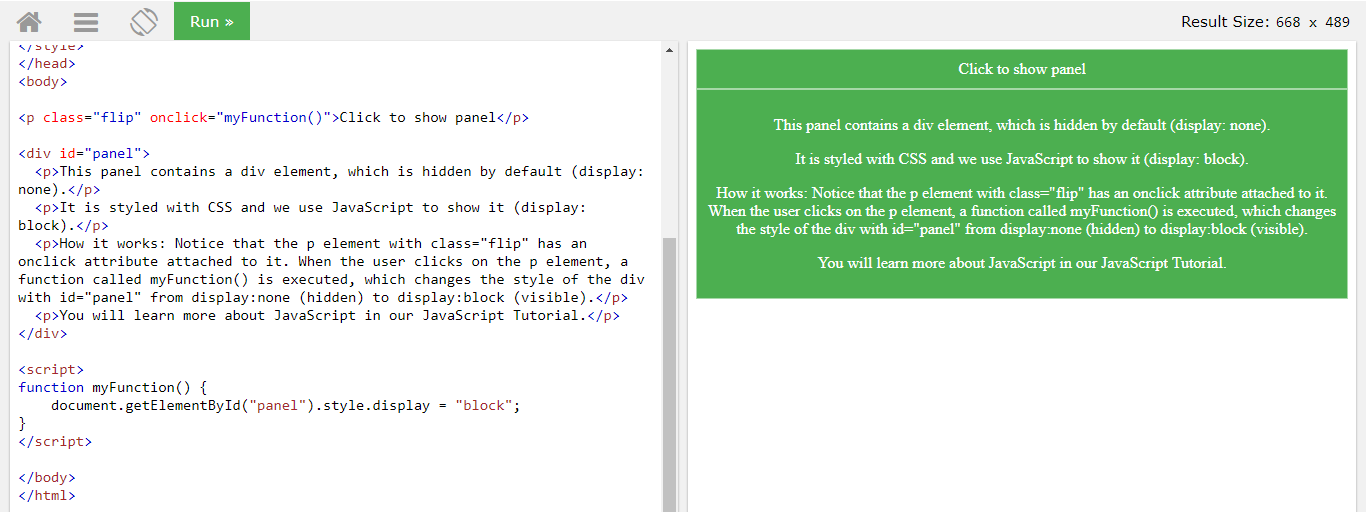
However, the element will still take up the same space as before. The element will be hidden, but still affect the layout:



### Difference between display:none and visiblity: hidden



This example demonstrates how to use CSS and JavaScript to show an element on click.



# CSS Layout - width and max-width

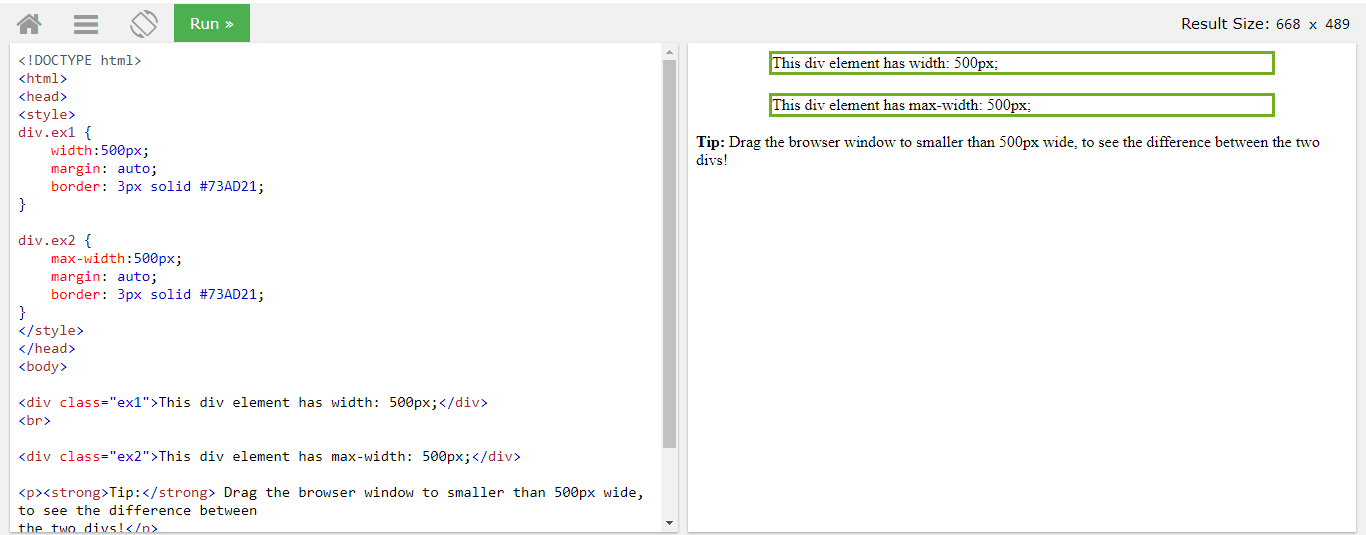
## Using width, max-width and margin: auto;

As mentioned in the previous chapter; a block-level element always takes up the full width available (stretches out to the left and right as far as it can).

Setting the width of a block-level element will prevent it from stretching out to the edges of its container. Then, you can set the margins to auto, to horizontally center the element within its container. The element will take up the specified width, and the remaining space will be split equally between the two margins.

**Note:** The problem with the <div> above occurs when the browser window is smaller than the width of the element. 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. This is important when making a site usable on small devices:



# CSS Layout - The position Property

## The position Property

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

There are four different position values:

* static
* relative
* fixed
* absolute

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:

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.

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.

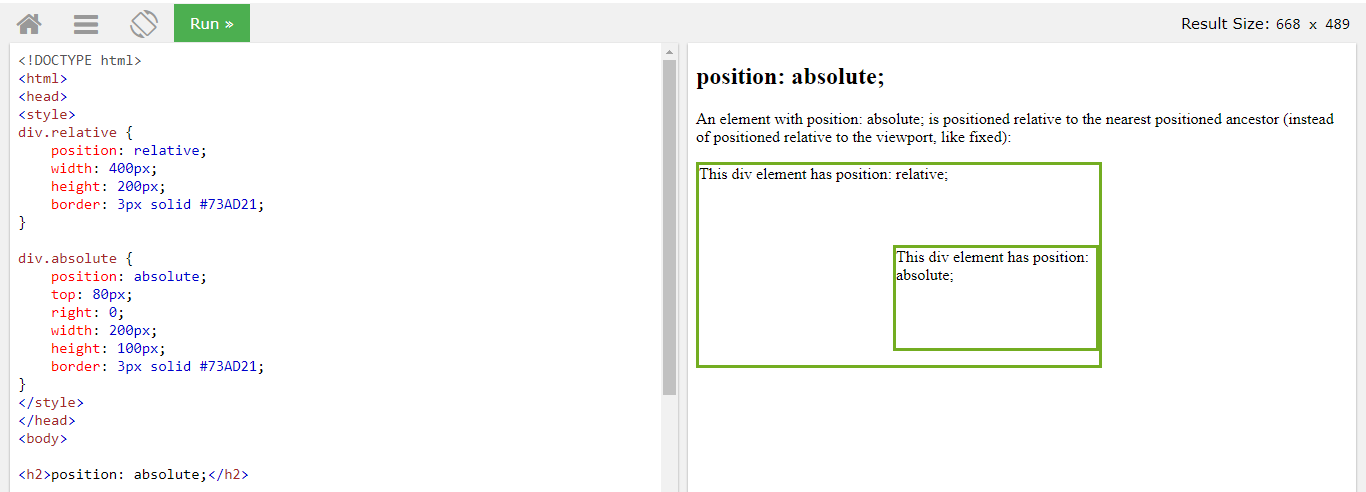
A fixed element does not leave a gap in the page where it would normally have been located.

## 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:** A "positioned" element is one whose position is anything except static.

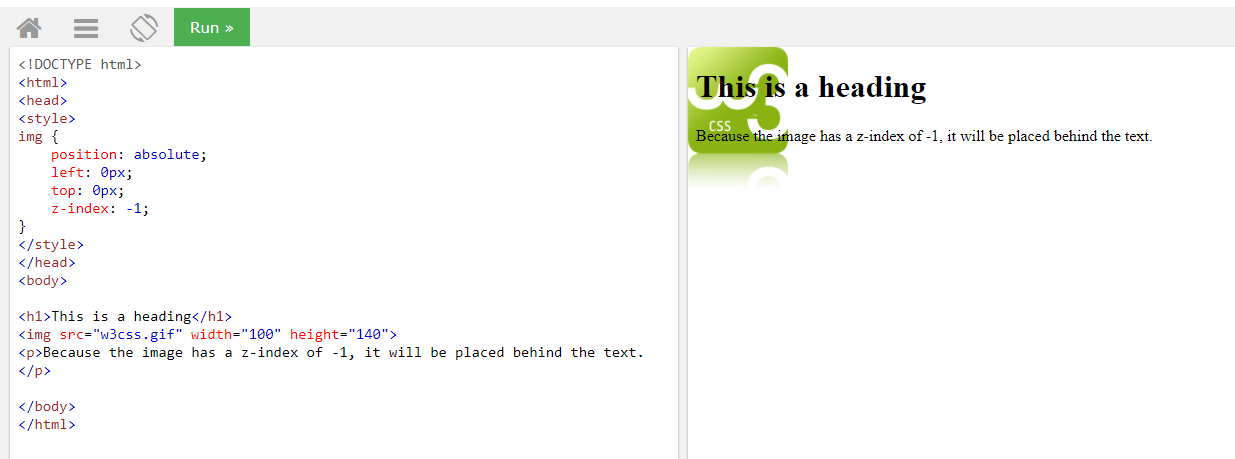


## Overlapping Elements

When elements are positioned, they can overlap other elements.

The z-index property specifies the stack order of an element (which element should be placed in front of, or behind, the others).

An element can have a positive or negative stack order:



An element with greater stack order is always in front of an element with a lower stack order.

**Note:** If two positioned elements overlap without a z-index specified, the element positioned last in the HTML code will be shown on top.

## Positioning Text In an Image

How to position text over an image:

