# \*CSS White Space

The **CSS white space property** is used to specify how to display the content within an element. It is used *to handle the white spaces inside an element*.

CSS White Space values

There are many white space values that can be used to display the content inside an element.

|  |  |
| --- | --- |
| **Value** | **Description** |
| normal | This is a default value. in this value, text is wrapped when necessary. sequences of white space will collapse into a single whitespace. |
| nowrap | Sequences of white space will collapse into a single whitespace. in this value, text will never wrap to the next line and only break when <br> tag is used. |
| pre | Whitespace is preserved by the browser. it is act like html <pre> tag. text will only wrap on line breaks. |
| pre-line | Sequences of white space will collapse into a single whitespace. texts are wrapped when necessary, and on line break. |
| pre-wrap | Whitespace is preserved by the browser. texts are wrapped when necessary, and on line break. |
| initial | It sets this property to its default value. |
| inherit | It inherits this property from its parent element. |

CSS White Space Example

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<style>**
5. p {
6. white-space: nowrap;
7. }
8. **</style>**
9. **</head>**
10. **<body>**
11. **<p>**
12. Write some text..Write some text..Write some text..
13. Write some text..Write some text..Write some text..
14. Write some text..Write some text..Write some text..
15. Write some text..Write some text..Write some text..
16. Write some text..Write some text..Write some text..
17. **</p>**
18. **</body>**
19. **</html>**

# CSS Width

The **CSS width property** is used to set the width of the content area of an element.

It does not include padding borders or margins. It sets width of the area inside the padding, border, and margin of the element.

## CSS width values

|  |  |
| --- | --- |
| **Value** | **Description** |
| auto | It is a default value. it is used to calculate the width. |
| length | It is used to define the width in px, cm etc. |
| % | It defines the width of the containing block in %. |
| initial | It is used to set the property to its default value. |
| inherit | It is used to inherit the property from its parent element. |

## CSS Width Example: width in px

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<style>**
5. img.normal {
6. width: auto;
7. }
8. img.big {
9. width: 150px;
10. }
11. p.ex {
12. height: 150px;
13. width: 150px;
14. }
15. **</style>**
16. **</head>**
17. **<body>**
18. **<img** class="normal" src="good-morning.jpg" width="95" height="84"**><br>**
19. **<img** class="big" src="good-morning.jpg" width="95" height="84"**>**
20. **<p** class="ex"**>**The height and width of this paragraph is 150px.**</p>**
21. **<p>**This is a paragraph.**</p>**
22. **</body>**
23. **</html>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=csswidth1)

Output:

  


The height and width of this paragraph is 150px.

This is a paragraph.

## CSS Width Example: width in %

The percent width is a measurement unit for the containing block. It is great for images.

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<style>**
5. img.normal {
6. width: auto;
7. }
8. img.big {
9. width: 50%;
10. }
11. img.small {
12. width: 10%;
13. }
14. **</style>**
15. **</head>**
16. **<body>**
17. **<img** class="normal" src="good-morning.jpg" width="95" height="84"**><br>**
18. **<img** class="big" src="good-morning.jpg" width="95" height="84"**><br>**
19. **<img** class="small" src="good-morning.jpg" width="95" height="84"**>**
20. **</body>**
21. **</html>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=csswidth2)

Output:

  
   


#### Note: You can also use the "min-width" and "max-width" property to control the size of image.

CSS Word Wrap

**CSS word wrap property** is used *to break the long words and wrap onto the next line*. This property is used to prevent overflow when an unbreakable string is too long to fit in the containing box.

CSS Word Wrap Values

|  |  |
| --- | --- |
| **Value** | **Description** |
| normal | This property is used to break words only at allowed break points. |
| break-word | It is used to break unbreakable words. |
| initial | It is used to set this property to its default value. |
| inherit | It inherits this property from its parent element. |

CSS Word Wrap Example

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<style>**
5. p.test {
6. width: 11em;
7. background-color: #00ffff;
8. border: 1px solid #000000;
9. padding:10px;
10. word-wrap: break-word;
11. }
12. **</style>**
13. **</head>**
14. **<body>**
15. **<p** class="test"**>** In this paragraph, there is a very long word:
16. iamsooooooooooooooooooooooooooooooolongggggggggggggggg.The long word will break and wrap to the next line.**</p>**
17. **</body>**
18. **</html>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=csswordwrap1)

Output:

In this paragraph, there is a very long word: iamsooooooooooooooooooooooooooooooolongggggggggggggggg.The long word will break and wrap to the next line.

CSS Outline

CSS outline is just like CSS border property. It facilitates you to draw an extra border around an element to get visual attention.

It is as easy as to apply as a border.

**See this example:**

1. <!DOCTYPE html**>**
2. **<html>**
3. **<style** type="text/css"**>**
4. .box {
5. background-color: #eee;
6. outline: 3px solid red;
7. border: 3px solid lightgreen;
8. padding: 5px 10px;
9. }
10. **</style>**
11. **<div** class="box"**>**Welcome to JavaTpoint**</div>**
12. **</body>**
13. **</html>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=css-outline1)

Difference between Border and Outline

At first glance, border and outline look similar, but there are some very important differences between them:

* It is not possible to apply a different outline width, style and color for the four sides of an element while in border; you can apply the provided value for all four sides of an element.
* The border is a part of element's dimension while the outline is not the part of element's dimension. Means, it doesn't matter how thick an outline you apply to the element, the dimensions of it won't change.

The outline property is a shorthand property. It can be divided into outline-width, outline-style and outline-color properties. It facilitates you to use any of the property alone, if you need it.

**See this example:**

1. <!DOCTYPE html**>**
2. **<html>**
3. **<style** type="text/css"**>**
4. .box {
5. background-color: #eee;
6. border: 3px solid Lightgreen;
7. padding: 5px 10px;
8. outline-width: 3px;
9. outline-style: solid;
10. outline-color: red;
11. }
12. **</style>**
13. **<div** class="box"**>**Welcome to JavaTpoint**</div>**
14. **</body>**
15. **</html>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=css-outline2)

In the above example, you can see the three outline properties:

**Outline-width:**It is similar to margin and padding. It can be either an absolute value or a relative value or one of the predefined outline width values i.e. thin, medium or thick. It is preferred to use either an absolute value or a relative value because different browsers interpret differently while using predefined outline width values like thin, medium or thick.

**Outline-color:**It specifies the color that you use for the outline. It supports all the colors available in HTML and CSS.

**Outline-style:**In the above example, we have used only solid outline style while there are a lot of outline style i.e. hidden, dotted, dashed, solid, double, groove, ridge, inset and outset.

Let's take an example to demonstrate the usage of different outline-styles.

**See this example:**

1. <!DOCTYPE html**>**
2. **<html>**
3. **<style** type="text/css"**>**
4. .box {
5. outline-color: red;
6. outline-width: 4px;
7. margin: 10px;
8. float: left;
9. border: 2px solid lightgreen;
10. padding: 5px 10px;
11. }
12. **</style>**
13. **<div** class="box" style="outline-style: dashed;"**>**This is dashed outline.**</div>**
14. **<div** class="box" style="outline-style: dotted;"**>**This is dotted outline.**</div>**
15. **<div** class="box" style="outline-style: double;"**>**This is double outline.**</div>**
16. **<div** class="box" style="outline-style: groove;"**>**This is groove outline.**</div>**
17. **<div** class="box" style="outline-style: inset;"**>**This is inset outline.**</div>**
18. **<div** class="box" style="outline-style: outset;"**>**This is outset outline.**</div>**
19. **<div** class="box" style="outline-style: ridge;"**>**This is ridge outline.**</div>**
20. **<div** class="box" style="outline-style: solid;"**>**This is solid outline.**</div>**
21. **</body>**
22. **</html>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=css-outline3)

Outline offset

The outline offset is used to create a distance between outline and border.

It takes a CSS length unit and the empty space between the border and the outline will be transparent and then it takes the background color of the parent element. So you can see a visible difference between outline and border.

Let's take an example to see the difference between outline and border.

**See this example:**

1. <!DOCTYPE html**>**
2. **<html>**
3. **<style** type="text/css"**>**
4. .box {
5. background-color: #eee;
6. outline: 3px solid red;
7. outline-offset: 6px;
8. border: 3px solid Lightgreen;
9. padding: 5px 10px;
10. }
11. **</style>**
12. **<div** class="box"**>**Welcome to JavaTpoint**</div>**
13. **</body>**
14. **</html>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=css-outline4)

CSS Visibility

The CSS visibility property is used to specify whether an element is visible or not.

**Note:** An invisible element also take up the space on the page. By using display property you can create invisible elements that don't take up space.

**Syntax:**

1. visibility: visible|hidden|collapse|initial|inherit;

CSS Visibility Parameters

**visible:**It is the by default value. It specifies that the element is visible.

**hidden:**It specifies that the element is invisible (but still takes up space).

**collapse:**It is used only for table elements. It is used to remove a row or column, but it does not affect the table layout.

The space taken up by the row or column will be available for other content.

If collapse is used on other elements, it renders as "hidden"

**initial:**It is used to set this property to its default value.

**inherit:**It is used to inherit this property from its parent element.

CSS Visibility Example

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<style>**
5. h1.visible {
6. visibility: visible
7. }
8. h1.hidden {
9. visibility: hidden
10. }
11. **</style>**
12. **</head>**
13. **<body>**
14. **<h1** class="visible"**>**I am visible**</h1>**
15. **<h1** class="hidden"**>**I am invisible**</h1>**
16. **<p><strong>**Note:**</strong>** An invisible element also take up the space on the page.
17. By using display property you can create invisible elements that don't
18. take space.**</p>**
19. **</body>**
20. **</html>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=css-visibility1)

JavaScript Syntax:

1. object.style.visibility="hidden"

**See the JavaScript example:**

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<style>**
5. #myDIV {
6. margin: auto;
7. width: 400px;
8. height: 200px;
9. background-color: lightpink;
10. border: 1px solid black;
11. }
12. **</style>**
13. **</head>**
14. **<body>**
15. **<p>**Click the "Try it" button to set the visibility property and hide the div element.**</p>**
16. **<button** onclick="myFunction()"**>**Try it**</button>**
17. **<div** id="myDIV"**>**This is my DIV element.**</div>**
18. **<p><strong>**Note:**</strong>** An invisible element also take up the space on the page. **</p>**
19. **<script>**
20. function myFunction() {
21. document.getElementById("myDIV").style.visibility = "hidden";
22. }
23. **</script>**
24. **</body>**
25. **</html>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=css-visibility2)

# CSS Counters

CSS counters are similar to variables. These are maintained by CSS and those values can be incremented by CSS rules to track how many times they are used.

CSS counters facilitate simple CSS based incrementing and display of a number for generated content.

## CSS Counter Properties

Following is a list of properties that are used with CSS counter:

* **counter-reset:** It is used to create or reset a counter.
* **counter-increment:** It is used to increment the counter value.
* **content:** It is used to insert generated content.
* **counter() or counters() function:** It is used to add the value of a counter to an element.

#### Note:Before using a CSS counter, it must be created with counter-reset.

## CSS Counter Example

Let's take an example to create a counter for a page and increment the counter value for each next element.

**See this example:**

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<style>**
5. body {
6. counter-reset: section;
7. }
8. h2::before {
9. counter-increment: section;
10. content: "Section " counter(section) ": ";
11. }
12. **</style>**
13. **</head>**
14. **<body>**
15. **<h1>**Example of CSS Counters:**</h1>**
16. **<h2>**Java Tutorial**</h2>**
17. **<h2>**HTML Tutorial**</h2>**
18. **<h2>**CSS Tutorial**</h2>**
19. **<h2>**Oracle Tutorial**</h2>**
20. **<p><strong>**Note:**</strong>** IE8 supports these properties only if a !DOCTYPE is specified.**</p>**
21. **</body>**
22. **</html>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=css-counter1)

#### Note: In the above example you can see that a counter is created for the page in the body selector and it increments the counter value for each <h2> element and adds "Section <value of the counter>:" to the beginning of each <h2> element.

## CSS Nesting Counters

You can also create counters within the counter. It is called nesting of a counter. Let's take an example to demonstrate nesting counter.

**See this example:**

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<style>**
5. body {
6. counter-reset: section;
7. }
8. h1 {
9. counter-reset: subsection;
10. }
11. h1::before {
12. counter-increment: section;
13. content: "Section " counter(section) ". ";
14. }
15. h2::before {
16. counter-increment: subsection;
17. content: counter(section) "." counter(subsection) " ";
18. }
19. **</style>**
20. **</head>**
21. **<body>**
22. **<h1>**Java tutorials:**</h1>**
23. **<h2>**Core Java tutorial**</h2>**
24. **<h2>**Servlet tutorial**</h2>**
25. **<h2>**JSP tutorial**</h2>**
26. **<h2>**Spring tutorial**</h2>**
27. **<h2>**Hibernate tutorial**</h2>**
29. **<h1>**Web technology tutorials:**</h1>**
30. **<h2>**HTML tutorial**</h2>**
31. **<h2>**CSS tutorial**</h2>**
32. **<h2>**jQuery tutorial**</h2>**
33. **<h2>**Bootstrap tutorial**</h2>**
35. **<h1>**Database tutorials:**</h1>**
36. **<h2>**SQL tutorial**</h2>**
37. **<h2>**MySQL tutorial**</h2>**
38. **<h2>**PL/SQL tutorial**</h2>**
39. **<h2>**Oracle tutorial**</h2>**
40. **<p><strong>**Note:**</strong>** IE8 supports these properties only if a !DOCTYPE is specified.**</p>**
41. **</body>**
42. **</html>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=css-counter2)

**Note:**In the above example you can see that a counter is created for the section and another nesting counter named subsection is created within section.

## Different level of nesting counters

You can create outlined lists by using nesting counters. It facilitates you to insert a string between different levels of nested counters.

**See this example:**

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<style>**
5. ol {
6. counter-reset: section;
7. list-style-type: none;
8. }
10. li::before {
11. counter-increment: section;
12. content: counters(section,".") " ";
13. }
14. **</style>**
15. **</head>**
16. **<body>**
17. **<h2>**Different level of Nesting counters**</h2>**
18. **<ol>**
19. **<li>**item**</li>**
20. **<li>**item
21. **<ol>**
22. **<li>**item**</li>**
23. **<li>**item**</li>**
24. **<li>**item
25. **<ol>**
26. **<li>**item**</li>**
27. **<li>**item**</li>**
28. **<li>**item**</li>**
29. **</ol>**
30. **</li>**
31. **<li>**item**</li>**
32. **</ol>**
33. **</li>**
34. **<li>**item**</li>**
35. **<li>**item**</li>**
36. **</ol>**
37. **<p><b>**Note:**</b>** IE8 supports these properties only if a !DOCTYPE is specified.**</p>**
38. **</body>**
39. **</html>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=css-counter3)