

CSS Counters

With CSS counters, you can create dynamic numbering of elements (like headings, sections, or list items) without using JavaScript.

CSS counters are "variables" maintained by CSS, and their values can be incremented (or decremented) by CSS rules.

CSS Automatic Numbering With Counters

CSS counters are like "variables". The variable values can be incremented (or decremented) by CSS rules.

To work with CSS counters we will use the following properties:

- counter-reset - Creates or resets a counter
 - counter-increment - Increments or decrements a counter
 - content - Inserts generated content (used inside ::before and ::after, to insert the generated content)
 - counter() and counters() functions - Adds the value of a counter to an element
-

CSS Increase and Decrease Counter

The following example creates a counter for the page (in the body selector), then it increments the counter value by 1 for each `<h2>` element and adds the text "Section + *countervalue*:" to the beginning of each `<h2>` element:

Example

```
body {  
  counter-reset: section;  
}  
  
h2::before {  
  counter-increment: section;  
  content: "Section " counter(section) ": ";  
}
```

The counter-increment property has a second parameter. The default value of this is 1 (which increments the counter by one). To decrease the counter value, you can set it to -1.

The following example is the same as above, but here we decrement the counter value for each <h2> element and adds the text "Section + *countervalue*:" to the beginning of each <h2> element:

Example

```
body {  
  counter-reset: section;  
}  
  
h2::before {  
  counter-increment: section -1;  
  content: "Section " counter(section) ": ";  
}
```

CSS Using Two Counters

The following example creates one counter for the page (named "section") and one counter for each <h1> element (named "subsection"). The "section" counter will be counted for each <h1> element with "Section + *sectioncounter*>.", and the "subsection" counter will be counted for each <h2> element with "*sectioncounter.subsectioncounter*":

Example

```
body {  
  counter-reset: section;  
}  
  
h1 {  
  counter-reset: subsection;  
}  
  
h1::before {  
  counter-increment: section;  
  content: "Section " counter(section) ". ";
```

```
}
```

```
h2::before {  
  counter-increment: subsection;  
  content: counter(section) "." counter(subsection) " ";  
}
```

The CSS counters() Function

The counters() function returns the current values of the named and nested counters, as a string.

Here we use the counters() function to insert a string between different levels of nested counters:

Example

```
ol {  
  counter-reset: section;  
  list-style-type: none;  
}  
  
li::before {  
  counter-increment: section;  
  content: counters(section, ".") " ";  
}
```

CSS Rounded Corners

The CSS border-radius property is used to create rounded corners for elements.

CSS border-radius Property

The border-radius property defines the radius of an element's corners.

This property can be applied to all elements with a background-color, a border, or a background-image.

Example

```
#div1 {  
  border-radius: 25px;  
  background-color: #73AD21;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}
```

```
#div2 {  
  border-radius: 25px;  
  border: 2px solid #73AD21;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}
```

```
#div3 {  
  border-radius: 25px;  
  background-image: url(paper.gif);  
  background-position: left top;  
  background-repeat: repeat;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}
```

CSS Elliptical and Circular Shapes

To create elliptical corners, you must specify two values for each radius, separated by a slash /. The first value defines the horizontal radius, and the second value defines the vertical radius.

Example

Create elliptical, oval and circular shapes:

```
#div1 {  
  border-radius: 70px / 30px;  
  background: #04AA6D;  
  width: 200px;  
  height: 150px;  
}
```

```
#div2 {  
  border-radius: 15px / 50px;  
  background: #04AA6D;  
  width: 200px;  
  height: 150px;  
}
```

```
#div3 {  
  border-radius: 50%;  
  background: #04AA6D;  
  width: 200px;  
  height: 150px;  
}
```

```
#div4 {  
  border-radius: 50%;  
  background: #04AA6D;  
  width: 200px;  
  height: 200px;  
}
```

CSS Border Images

With the CSS border-image property, you can define an image to be used as the border around an element.

CSS border-image Property

The [border-image](#) property allows you to define an image to be used as the border around an element, instead of the normal border.

This property takes an image and slices it into nine sections, like a tic-tac-toe board. It then places the corners at the corners, and the middle sections are repeated or stretched as you specify.

The border-image property is a shorthand property for the following properties:

- [border-image-source](#) - defines the path to the image
- [border-image-slice](#) - defines how to slice the image
- [border-image-width](#) - defines the width of the image
- [border-image-outset](#) defines the amount by which the border image area extends beyond the border box
- [border-image-repeat](#) - defines how to repeat the image

Note: For border-image to work, the element also needs the [border](#) property set!

CSS border-image Examples

We will use the following image (named "border.png"):



Example

```
#borderimg {  
  border: 10px solid transparent; /* Required for border-image */  
  padding: 15px;  
  border-image: url(border.png) 30 round;  
}
```

Here, the stretch value specifies that the middle section of the image is stretched to fill the area:

An image as the border!

Here is the code:

Example

```
#bordering {  
  border: 10px solid transparent; /* Required for border-image */  
  padding: 15px;  
  border-image: url(border.png) 30 stretch;  
}
```
