

CSS Tutorial

Save a lot of work with CSS!

In our CSS tutorial you will learn how to use CSS to control the style and layout of multiple Web pages all at once.

Examples in Each Chapter

This CSS tutorial contains hundreds of CSS examples.

With our online editor, you can edit the CSS, and click on a button to view the result.

CSS Example

```
body {  
    background-color: #d0e4fe;  
}  
  
h1 {  
    color: orange;  
    text-align: center;  
}  
  
p {  
    font-family: "Times New Roman";  
    font-size: 20px;  
}
```

What is CSS?

- **CSS** stands for **C**ascading **S**tyle **S**heets
- CSS defines **how HTML elements are to be displayed**
- Styles were added to HTML 4.0 **to solve a problem**
- CSS saves a lot of work

- External Style Sheets are stored in **CSS files**

• CSS Solved a Big Problem

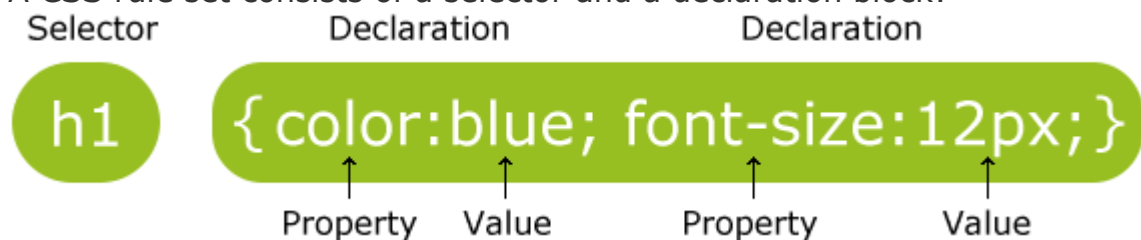
- HTML was NEVER intended to contain tags for formatting a document.
- HTML was intended to **define the content** of a document, like:
 - `<h1>This is a heading</h1>`
 - `<p>This is a paragraph.</p>`
- When tags like ``, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large web sites, 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.
- In HTML 4.0, all formatting could (and should!) be removed from the HTML document, and stored in a separate CSS file.

CSS Saves a Lot of Work!

- The style definitions are normally saved in external .css files.
- With an external style sheet file, you can change the look of an entire Web site by changing just one file!

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 property name and a value, separated by a colon.

• CSS Example

- A CSS declaration always ends with a semicolon, and declaration groups are surrounded by curly braces:
- `p {color:red;text-align:center;}`

- To make the CSS code more readable, you can put one declaration on each line.
- In the following example all <p> elements will be center-aligned, with a red text color:

• Example

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

• CSS Comments

- Comments are used to explain your code, and may help you when you edit the source code at a later date. Comments are ignored by browsers.
- A CSS comment starts with /* and ends with */. Comments can also span multiple lines:

• Example

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

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

• CSS Selectors

- CSS selectors allow you to select and manipulate HTML elements.
- CSS selectors are used to "find" (or select) HTML elements based on their id, class, type, attribute, and more.

• The element Selector

- The element selector selects elements based on the element name.
- You can select all <p> elements on a page like this: (all <p> elements will be center-aligned, with a red text color)

• Example

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

• The id Selector

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

- An id should be unique within a page, so the id selector is used if you want to select a single, 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;
}
```

• The class Selector

- The class selector selects elements with a specific class attribute.
- To select elements with a specific class, write a period character, followed by the name of the class:
- In the example below, all HTML elements with class="center" will be center-aligned:

- **Example**

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

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

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

Example

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

Grouping Selectors

If you have elements with the same style definitions, like this:

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

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

you can group the selectors, to minimize the code.

To group selectors, separate each selector with a comma.

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

Example

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

CSS How To...

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

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

An external style sheet is ideal when the style is applied to many pages. With an external style sheet, you can change the look of an entire Web site by changing just one file.

Each page must include a link to the style sheet with the <link> tag. The <link> tag goes inside the head section:

```
<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. An example of a style sheet file called "myStyle.css", is shown below:

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



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 Style Sheet

An internal style sheet should be used when a single document has a unique style. You define internal styles in the head section of an HTML page, inside the <style> tag, like this:

Example

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

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

Inline Styles

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

To use inline styles, add the style attribute to the relevant tag. The style attribute can contain any CSS property. The example 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>
```

Multiple Style Sheets

If some properties have been set for the same selector in different style sheets, the values will be inherited from the more specific style sheet.

For example, assume that an external style sheet has the following properties for the <h1> element:

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

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

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

If the page with the internal style sheet also links to the external style sheet the properties for the <h1> element will be:

```
color: orange;  
margin-left: 20px;
```

Multiple Styles Will Cascade into One

Styles can be specified:

- inside an HTML element
- inside the <head> section of an HTML page
- in an external CSS file

Tip: Even multiple external style sheets can be referenced inside a single HTML document.

Cascading order

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

Generally speaking we can say that all the styles will "cascade" into a new "virtual" style sheet by the following rules, where number four has the highest priority:

1. Browser default
2. External style sheet
3. Internal style sheet (in the head section)
4. Inline style (inside an HTML element)
5. So, an inline style (inside an 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, or in a browser (a default value).



Note: If the link to the external style sheet is placed after the internal style sheet in HTML <head>, the external style sheet will override the internal style sheet!

CSS Background

Background Color

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

The background color of a page is set like this:

Example

```
body {  
    background-color: #b0c4de;  
}
```

With CSS, a color is most often specified by:

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

Look at [CSS Color Values](#) for a complete list of possible color values.

In the example below, the <h1>, <p>, and <div> elements have different background colors:

Example

```
h1 {  
    background-color: #6495ed;  
}  
  
p {  
    background-color: #e0ffff;  
}  
  
div {  
    background-color: #b0c4de;  
}  
  
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:

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

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

Example

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



Note: To repeat an image vertically set background-repeat: repeat-y;

Background Image - Set position and no-repeat



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

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

Example

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

In the example above, the background image is shown 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.

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

Example

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

Background - Shorthand property

As you can see from the examples above, there are many properties to consider when dealing with backgrounds.

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

The shorthand property for background is simply "background":

Example


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

CSS Font

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

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.

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;  
}
```

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 */  
}  
  
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 IE. 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;  
}
```

CSS Links

Links can be styled in different ways.

Styling Links

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

Example

```
a {  
    color: #FF0000;  
}
```

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: #FF0000;
}

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

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

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

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

## Common Link Styles

In the example above the link changes color depending on what state it is in.

Lets go through some of the other common ways to style links:

## 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 specifies the background color for links:

### Example

```
a:link {
 background-color: #B2FF99;
}

a:visited {
 background-color: #FFFF85;
}

a:hover {
 background-color: #FF704D;
}

a:active {
 background-color: #FF704D;
}
```

## CSS Box Model

### 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 HTML elements, and it consists of: margins, borders, padding, and the actual content.

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

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

### Example

```
div {
 width: 300px;
 padding: 25px;
 border: 25px solid navy;
 margin: 25px;
}
```

## 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.

Let's style a <div> element to have a total width of 350px:

### Example

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

Let's do 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

## Browsers Compatibility Issue

Internet Explorer 8 and earlier versions, include padding and border in the width property.

To fix this problem, add a <!DOCTYPE html> to the HTML page.

# CSS Border

## CSS Border Properties

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

## Border Style

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



**Note:** None of the border properties will have ANY effect unless the **border-style** p

## border-style values:

none: Defines no border

dotted: Defines a dotted border

dashed: Defines a dashed border

solid: Defines a solid border

double: Defines two borders. The width of the two borders are the same as the border-width value

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

# Border Width

The border-width property is used to set the width of the border.

The width is set in pixels, or by using one of the three pre-defined values: thin, medium, or thick.

**Note:** The "border-width" property does not work if it is used alone. Use the "border-style" property to set the borders first.

## Example

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

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

# Border Color

The border-color property is used to set the color of the border. The color can be set by:

- name - specify a color name, like "red"
- RGB - specify a RGB value, like "rgb(255,0,0)"
- Hex - specify a hex value, like "#ff0000"

You can also set the border color to "transparent".

If the border color is not set it is inherited from the color property of the element.

**Note:** The "border-color" property does not work if it is used alone. Use the "border-style" property to set the borders first.

## Example

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

p.two {
 border-style: solid;
 border-color: #98bf21;
}
```

# Border - Individual sides

In CSS it is possible to specify different borders for different sides:

## Example

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

The example above can also be set with a single property:

## Example

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

The border-style property can have from one to four values.

- **border-style: dotted solid double dashed;**

- top border is dotted
- right border is solid
- bottom border is double
- left border is dashed

- **border-style: dotted solid double;**

- top border is dotted
- right and left borders are solid
- bottom border is double

- **border-style: dotted solid;**

- top and bottom borders are dotted
- right and left borders are solid

- **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. This is called a shorthand property.

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

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

### Example

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

## CSS Display and Visibility

The display property specifies if/how an element is displayed, and the visibility property specifies if an element should be visible or hidden.

### Hiding an Element - display:none or visibility:hidden

Hiding an element can be done by setting the display property to "none" or the visibility property to "hidden". However, notice that these two methods produce different results:

visibility:hidden hides an element, but it will still take up the same space as before. The element will be hidden, but still affect the layout:

### Example

```
h1.hidden {
 visibility: hidden;
}
```

display:none hides an element, and it will not take up any space. The element will be hidden, and the page will be displayed as if the element is not there:

### Example

```
h1.hidden {
 display: none;
}
```

## CSS Display - Block and Inline Elements

A block element is an element that takes up the full width available, and has a line break before and after it.

Examples of block elements:

- <h1>
- <p>
- <li>
- <div>

An inline element only takes up as much width as necessary, and does not force line breaks.

Examples of inline elements:

- <span>
- <a>

## Changing How an Element is Displayed

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 web standards.

The following example displays <li> elements as inline elements:

### Example

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

The following example displays `<span>` elements as block elements:

### Example

```
span {
 display: block;
}
```



**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 of it.

# CSS Navigation Bar

## Demo: Navigation Bar

[HOME](#) [NEWS](#) [ARTICLES](#) [FORUM](#) [CONTACT](#) [ABOUT](#)

## Navigation Bars

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

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

## Navigation Bar = List of Links

A navigation bar needs standard HTML as a base.

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

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

### Example

```

 Home
 News
 Contact
 About

```

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

### Example

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

Example explained:

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

The code in the example above is the standard code used in both vertical, and horizontal navigation bars.

## Vertical Navigation Bar

To build a vertical navigation bar we only need to style the <a> elements, in addition to the code above:

### Example

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

Example explained:

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

**Tip:** Also take a look at our [fully styled vertical navigation bar example](#).



**Note:** Always specify the width for <a> elements in a vertical navigation bar. If you omit the width, IE6 can produce unexpected results.



# Horizontal Navigation Bar

There are two ways to create a horizontal navigation bar.

Using **inline** or **floating** list items.

Both methods work fine, but if you want the links to be the same size, you have to use the floating method.

## Inline List Items

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

### Example

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

Example explained:

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

## Floating List Items

In the example above the links have different widths.

For all the links to have an equal width, float the `<li>` elements and specify a width for the `<a>` elements:

### Example

```
li {
 float: left;
}

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

Example explained:

- `float: left` - use float to get block elements to slide next to each other

- `display: block` - Displaying the links as block elements makes the whole link area clickable (not just the text), and it allows us to specify the width
- `width: 60px` - Since block elements take up the full width available, they cannot float next to each other. We specify the width of the links to 60px

# CSS3 Introduction

CSS3 is the latest standard for CSS.

CSS3 is completely backwards-compatible with earlier versions of CSS.

This section teaches you about the new features in CSS3!

## CSS3 Modules

CSS3 has been split into "modules". It contains the "old CSS specification" (which has been split into smaller pieces). In addition, new modules are added.

Some of the most important CSS3 modules are:

- Selectors
- Box Model
- Backgrounds and Borders
- Image Values and Replaced Content
- Text Effects
- 2D/3D Transformations
- Animations
- Multiple Column Layout
- User Interface

## CSS3 Recommendation

Most of the CSS3 Modules are W3C Recommendations, and CSS3 properties are implemented in all modern browsers.

## CSS3 Borders

### CSS3 Borders

With CSS3, you can create rounded borders, add shadow to boxes, and use an image as a border - without using a design program, like Photoshop.

In this chapter you will learn about the following border properties:

- border-radius
- box-shadow
- border-image

## • Browser Support

- The numbers in the table specify the first browser version that fully supports the property.
- Numbers followed by -webkit-, -moz-, or -o- specify the first version that worked with a prefix.

Property	IE	Chrome	Firefox	Safari	Opera
<b>border-radius</b>	9.0	5.0 4.0 - webkit-	4.0 3.0 -moz-	5.0 3.1 - webkit-	10.5
<b>box-shadow</b>	9.0	10.0 4.0 - webkit-	4.0 3.5 -moz-	5.1 3.1 - webkit-	10.5
<b>border-image</b>	11.0	16.0 4.0 - webkit-	15.0 3.5 -moz-	6.0 3.1 - webkit-	15.0 11.0 -o-

## • CSS3 The border-radius Property - Rounded Corners

- Adding rounded corners in CSS2 was tricky. We had to use different images for each corner.
- In CSS3, creating rounded corners is easy.
- In CSS3, the border-radius property is used to create rounded corners:
  - This box has rounded corners!

- Example
- Add rounded corners to a <div> element:
- ```
div {
  border: 2px solid;
  border-radius: 25px;
}
```

• CSS3 The border-radius Property - Rounded Corners

- Adding rounded corners in CSS2 was tricky. We had to use different images for each corner.
- In CSS3, creating rounded corners is easy.
- In CSS3, the border-radius property is used to create rounded corners:
 - This box has rounded corners!

• Example

- Add rounded corners to a <div> element:

```
div {  
    border: 2px solid;  
    border-radius: 25px;  
}
```

• CSS3 The box-shadow Property

- In CSS3, the box-shadow property is used to add shadow to boxes:

• Example

- Add a box-shadow to a <div> element:

```
div {  
    box-shadow: 10px 10px 5px #888888;  
}
```

CSS3 The border-image Property

With the CSS3 border-image property you can use an image to create a border:

The border-image property allows you to specify an image as a border!

The original image used to create the border above:



Example

Use an image to create a border around a <div> element:

```
div {  
    -webkit-border-image: url(border.png) 30 30 round; /* Safari 3.1-5  
*/
```

```

    -o-border-image: url(border.png) 30 30 round; /* Opera 11-12.1 */
    border-image: url(border.png) 30 30 round;
}

```

CSS3 Gradients



CSS3 gradients let you display smooth transitions between two or more specified colors.

Earlier, you had to use images for these effects. However, by using CSS3 gradients you can reduce download time and bandwidth usage. In addition, elements with gradients look better when zoomed, because the gradient is generated by the browser.

CSS3 defines two types of gradients:

- **Linear Gradients (goes down/up/left/right/diagonally)**
- **Radial Gradients (defined by their center)**

Browser Support

The numbers in the table specify the first browser version that fully supports the property.

Numbers followed by -webkit-, -moz-, or -o- specify the first version that worked with a prefix.

| Property | IE | Chrome | Firefox | Safari | Opera |
|-----------------|------|---------------------------|-----------------------|-------------------------|----------------------|
| linear-gradient | 10.0 | 26.0
10.0 -
webkit- | 16.0
3.6 -
moz- | 6.1
5.1 -
webkit- | 12.1
11.1 -
o- |

| | | | | | |
|---------------------------|------|---------------------------|-----------------------|-------------------------|----------------------|
| radial-gradient | 10.0 | 26.0
10.0 -
webkit- | 16.0
3.6 -
moz- | 6.1
5.1 -
webkit- | 12.1
11.6 -
o- |
| repeating-linear-gradient | 10.0 | 26.0
10.0 -
webkit- | 16.0
3.6 -
moz- | 6.1
5.1 -
webkit- | 12.1
11.1 -
o- |
| repeating-radial-gradient | 10.0 | 26.0
10.0 -
webkit- | 16.0
3.6 -
moz- | 6.1
5.1 -
webkit- | 12.1
11.6 -
o- |

CSS3 Linear Gradients

To create a linear gradient you must define at least two color stops. Color stops are the colors you want to render smooth transitions among. You can also set a starting point and a direction (or an angle) along with the gradient effect.

Example of Linear Gradient:



Syntax

```
background: linear-gradient(direction, color-stop1, color-stop2, ...);
```

Linear Gradient - Top to Bottom (this is default)

The following example shows a linear gradient that starts at the top. It starts red, transitioning to blue:

Example

A linear gradient from top to bottom:

```
#grad {
  background: -webkit-linear-gradient(red, blue); /* For Safari 5.1 to
```

```

6.0 */
    background: -o-linear-gradient(red, blue); /* For Opera 11.1 to 12.0
*/
    background: -moz-linear-gradient(red, blue); /* For Firefox 3.6 to 15
*/
    background: linear-gradient(red, blue); /* Standard syntax */
}

```

Linear Gradient - Left to Right

The following example shows a linear gradient that starts from the left. It starts red, transitioning to blue:

Example

A linear gradient from left to right:

```

#grad {
    background: -webkit-linear-gradient(left, red , blue); /* For Safari
5.1 to 6.0 */
    background: -o-linear-gradient(right, red, blue); /* For Opera 11.1
to 12.0 */
    background: -moz-linear-gradient(right, red, blue); /* For Firefox
3.6 to 15 */
    background: linear-gradient(to right, red , blue); /* Standard syntax
*/
}

```

Linear Gradient - Diagonal

You can make a gradient diagonally by specifying both the horizontal and vertical starting positions.

The following example shows a linear gradient that starts at top left (and goes to bottom right). It starts red, transitioning to blue:

Example

A linear gradient that starts at top left (and goes to bottom right):

```

#grad {
    background: -webkit-linear-gradient(left top, red , blue); /* For
Safari 5.1 to 6.0 */
    background: -o-linear-gradient(bottom right, red, blue); /* For Opera
11.1 to 12.0 */
    background: -moz-linear-gradient(bottom right, red, blue); /* For
Firefox 3.6 to 15 */
    background: linear-gradient(to bottom right, red , blue); /* Standard
syntax */
}

```


Using Angles

If you want more control over the direction of the gradient, you can define an angle, instead of the predefined directions (to bottom, to top, to right, to left, to bottom right, etc.).

Syntax

```
background: linear-gradient(angle, color-stop1, color-stop2);
```

The angle is specified as an angle between a horizontal line and the gradient line, going counter-clockwise. In other words, 0deg creates a bottom to top gradient, while 90deg generates a left to right gradient.

The following example shows how to use angles on linear gradients:

Example

A linear gradient with a specified angle:

```
#grad {  
  background: -webkit-linear-gradient(180deg, red, blue); /* For Safari  
5.1 to 6.0 */  
  background: -o-linear-gradient(180deg, red, blue); /* For Opera 11.1  
to 12.0 */  
  background: -moz-linear-gradient(180deg, red, blue); /* For Firefox  
3.6 to 15 */  
  background: linear-gradient(180deg, red, blue); /* Standard syntax */  
}
```

Using Multiple Color Stops

The following example shows how to set multiple color stops:

Example

A linear gradient from top to bottom with multiple color stops:

```
#grad {  
  background: -webkit-linear-gradient(red, green, blue); /* For Safari  
5.1 to 6.0 */  
  background: -o-linear-gradient(red, green, blue); /* For Opera 11.1  
to 12.0 */  
  background: -moz-linear-gradient(red, green, blue); /* For Firefox  
3.6 to 15 */  
  background: linear-gradient(red, green, blue); /* Standard syntax */  
}
```

The following example shows how to create a linear gradient with the color of the rainbow and some text:

Example

```
#grad {
  /* For Safari 5.1 to 6.0 */
  background: -webkit-linear-
gradient(left,red,orange,yellow,green,blue,indigo,violet);
  /* For Opera 11.1 to 12.0 */
  background: -o-linear-
gradient(left,red,orange,yellow,green,blue,indigo,violet);
  /* For Fx 3.6 to 15 */
  background: -moz-linear-
gradient(left,red,orange,yellow,green,blue,indigo,violet);
  /* Standard syntax */
  background: linear-gradient(to right,
red,orange,yellow,green,blue,indigo,violet);
}
```

Using Transparency

CSS3 gradients also support transparency, which can be used to create fading effects.

To add transparency, we use the `rgba()` function to define the color stops. The last parameter in the `rgba()` function can be a value from 0 to 1, and it defines the transparency of the color: 0 indicates full transparency, 1 indicates full color (no transparency).

The following example shows a linear gradient that starts from the left. It starts fully transparent, transitioning to full color red:

Example

A linear gradient from left to right, with transparency:

```
#grad {
  background: -webkit-linear-
gradient(left,rgba(255,0,0,0),rgba(255,0,0,1)); /*Safari 5.1-6*/
  background: -o-linear-
gradient(right,rgba(255,0,0,0),rgba(255,0,0,1)); /*Opera 11.1-12*/
  background: -moz-linear-
gradient(right,rgba(255,0,0,0),rgba(255,0,0,1)); /*Fx 3.6-15*/
  background: linear-gradient(to right, rgba(255,0,0,0),
rgba(255,0,0,1)); /*Standard*/
}
```

Repeating a linear-gradient

The repeating-linear-gradient() function is used to repeat linear gradients:

Example

A repeating linear gradient:

```
#grad {  
  /* Safari 5.1 to 6.0 */  
  background: -webkit-repeating-linear-gradient(red, yellow 10%, green 20%);  
  /* Opera 11.1 to 12.0 */  
  background: -o-repeating-linear-gradient(red, yellow 10%, green 20%);  
  /* Firefox 3.6 to 15 */  
  background: -moz-repeating-linear-gradient(red, yellow 10%, green 20%);  
  /* Standard syntax */  
  background: repeating-linear-gradient(red, yellow 10%, green 20%);  
}
```

CSS3 Radial Gradients

A radial gradient is defined by its center.

To create a radial gradient you must also define at least two color stops.

Example of Radial Gradient:



Syntax

```
background: radial-gradient(shape size at position, start-color, ..., last-color);
```

By default, shape is ellipse, size is farthest-corner, and position is center.

Radial Gradient - Evenly Spaced Color Stops (this is default)

Example

A radial gradient with evenly spaced color stops:

```
#grad {  
    background: -webkit-radial-gradient(red, green, blue); /* Safari 5.1  
to 6.0 */  
    background: -o-radial-gradient(red, green, blue); /* For Opera 11.6  
to 12.0 */  
    background: -moz-radial-gradient(red, green, blue); /* For Firefox  
3.6 to 15 */  
    background: radial-gradient(red, green, blue); /* Standard syntax */  
}
```

Radial Gradient - Differently Spaced Color Stops

Example

A radial gradient with differently spaced color stops:

```
#grad {  
    background: -webkit-radial-gradient(red 5%, green 15%, blue 60%); /*  
Safari 5.1-6.0 */  
    background: -o-radial-gradient(red 5%, green 15%, blue 60%); /* For  
Opera 11.6-12.0 */  
    background: -moz-radial-gradient(red 5%, green 15%, blue 60%); /* For  
Firefox 3.6-15 */  
    background: radial-gradient(red 5%, green 15%, blue 60%); /* Standard  
syntax */  
}
```

Set Shape

The shape parameter defines the shape. It can take the value circle or ellipse. The default value is ellipse.

Example

A radial gradient with the shape of a circle:

```
#grad {  
    background: -webkit-radial-gradient(circle, red, yellow, green); /*  
Safari */  
    background: -o-radial-gradient(circle, red, yellow, green); /* Opera  
11.6 to 12.0 */  
    background: -moz-radial-gradient(circle, red, yellow, green); /*  
Firefox 3.6 to 15 */  
    background: radial-gradient(circle, red, yellow, green); /* Standard  
syntax */  
}
```

Use of Different Size Keywords

The size parameter defines the size of the gradient. It can take four values:

- **closest-side**
- **farthest-side**
- **closest-corner**
- **farthest-corner**

Example

A radial gradient with different size keywords:

```
#grad1 {
  /* Safari 5.1 to 6.0 */
  background: -webkit-radial-gradient(60% 55%, closest-
side,blue,green,yellow,black);
  /* For Opera 11.6 to 12.0 */
  background: -o-radial-gradient(60% 55%, closest-
side,blue,green,yellow,black);
  /* For Firefox 3.6 to 15 */
  background: -moz-radial-gradient(60% 55%, closest-
side,blue,green,yellow,black);
  /* Standard syntax */
  background: radial-gradient(closest-side at 60%
55%,blue,green,yellow,black);
}

#grad2 {
  /* Safari 5.1 to 6.0 */
  background: -webkit-radial-gradient(60% 55%, farthest-
side,blue,green,yellow,black);
  /* Opera 11.6 to 12.0 */
  background: -o-radial-gradient(60% 55%, farthest-
side,blue,green,yellow,black);
  /* For Firefox 3.6 to 15 */
  background: -moz-radial-gradient(60% 55%, farthest-
side,blue,green,yellow,black);
  /* Standard syntax */
  background: radial-gradient(farthest-side at 60%
55%,blue,green,yellow,black);
}
```

Repeating a radial-gradient

The repeating-radial-gradient() function is used to repeat radial gradients:

Example

A repeating radial gradient:

```
#grad {  
  /* For Safari 5.1 to 6.0 */  
  background: -webkit-repeating-radial-gradient(red, yellow 10%, green  
15%);  
  /* For Opera 11.6 to 12.0 */  
  background: -o-repeating-radial-gradient(red, yellow 10%, green 15%);  
  /* For Firefox 3.6 to 15 */  
  background: -moz-repeating-radial-gradient(red, yellow 10%, green  
15%);  
  /* Standard syntax */  
  background: repeating-radial-gradient(red, yellow 10%, green 15%);  
}
```