

CSS with style

CSS Selector	Description
Inherited styles	Lowest specificity of all selectors - since an inherited style targets the element's parent, and not the HTML element itself.
*	Lowest specificity of all directly targeted selectors
element	Higher specificity than universal selector and inherited styles.
attribute	Higher specificity than element selector
class	Higher specificity than attribute, element and universal selectors.
ID	Higher specificity than class selector.
Combined selectors	Gets the specificity of the selectors combined.
CSS properties set directly on element, inside <code>style</code> attribute.	Stronger specificity than ID selector.

CSS3 Introduction

CSS3 Modules

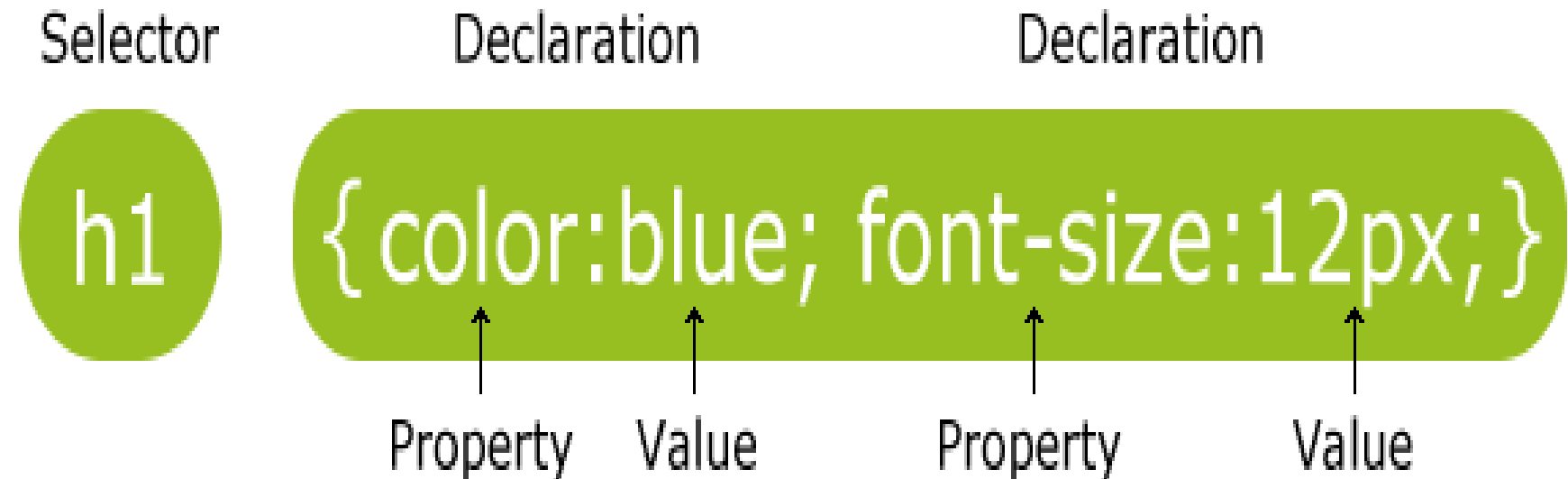
CSS3 is split up into "modules". The old specification has been split into smaller pieces, and new ones are also added.

Some of the most important CSS3 modules are:

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

CSS Syntax

A CSS rule has two main parts: a selector, and one or more declarations:



The selector is normally the HTML element you want to style.

Each declaration consists of a property and a value.

The property is the style attribute you want to change. Each property has a value.

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:



CSS Example

A CSS declaration always ends with a semicolon, and declaration groups are surrounded by curly brackets:

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

To make the CSS more readable, you can put one declaration on each line, like this:

Example

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

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

Property	Browser Support				
border-radius					
box-shadow					
border-image					

CSS3 Backgrounds

CSS3 Backgrounds

CSS3 contains several new background properties, which allow greater control of the background element.

- background-size
- background-origin

CSS3 Fonts

The CSS3 @font-face Rule


Before CSS3, web designers had to use fonts that were already installed on the user's computer.

With CSS3, web designers can use whatever font he/she likes.

When you have found/bought the font you wish to use, include the font file on your web server, and it will be automatically downloaded to the user when needed.

Your "own" fonts are defined in the CSS3 @font-face rule.

Browser Support

Property	Browser Support				
@font-face					

Example

```
<style>
@font-face
{
font-family: myFirstFont;
src: url(sansation_light.woff);
}

div
{
font-family:myFirstFont;
}
</style>
```

CSS3 Animations

With CSS3, we can create animations, which can replace animated images, Flash animations, and JavaScripts in many web pages.

CSS3

Animation

CSS3 @keyframes Rule

To create animations in CSS3, you will have to learn about the @keyframes rule.

The @keyframes rule is where the animation is created. Specify a CSS style inside the @keyframes rule and the animation will gradually change from the current style to the new style.

Browser Support

Property	Browser Support				
@keyframes					
animation					

Internet Explorer 10, Firefox, and Opera supports the @keyframes rule and animation property.

CSS Pseudo-classes

CSS pseudo-classes are used to add special effects to some selectors.

Syntax

The syntax of pseudo-classes:

```
selector:pseudo-class {property:value;}
```

CSS classes can also be used with pseudo-classes:

```
selector.class:pseudo-class {property:value;}
```

Anchor Pseudo-classes

Links can be displayed in different ways in a CSS-supporting browser:

Example

```
a:link {color:#FF0000;}      /* unvisited link */  
a:visited {color:#00FF00;}  /* visited link */  
a:hover {color:#FF00FF;}    /* mouse over link */  
a:active {color:#0000FF;}   /* selected link */
```

All CSS Pseudo Classes/Elements

Selector	Example	Example description
<u>:link</u>	a:link	Selects all unvisited links
<u>:visited</u>	a:visited	Selects all visited links
<u>:active</u>	a:active	Selects the active link
<u>:hover</u>	a:hover	Selects links on mouse over
<u>:focus</u>	input:focus	Selects the input element which has focus
<u>:first-letter</u>	p:first-letter	Selects the first letter of every <p> element
<u>:first-line</u>	p:first-line	Selects the first line of every <p> element
<u>:first-child</u>	p:first-child	Selects every <p> elements that is the first child of its parent
<u>:before</u>	p:before	Insert content before every <p> element
<u>:after</u>	p:after	Insert content after every <p> element
<u>:lang(<i>language</i>)</u>	p:lang(it)	Selects every <p> element with a lang attribute value starting with "it"

Example

CSS - The :first-child Pseudo-class

The :first-child pseudo-class matches a specified element that is the first child of another element.

Match the first <p> element

In the following example, the selector matches any <p> element that is the first child of any element:

```
<!DOCTYPE html>
<html>
<head>
<style type="text/css">
p:first-child
{
color:blue;
}
</style>
</head>

<body>
<p>Hi Welcome in CSS Program.</p>
<p>Hi Welcome in CSS Program.</p>
<p><b>Note:</b> For :DOCTYPE must be declared.</p>
</body>
</html>
```

Example

Match the first `<i>` element in all `<p>` elements

In the following example, the selector matches the first `<i>` element in all `<p>` elements:

```
<!DOCTYPE html>
<html>
<head>
<style type="text/css">
p > i:first-child
{
color:blue;
}
</style>
</head>

<body>
<p>I am a <i>strong</i> man. I am a <i>strong</i> man.</p>
<p>I am a <i>strong</i> man. I am a <i>strong</i> man.</p>
<p><b>Note:</b> For :first-child to work in IE8 and earlier, a DOCTYPE must be
declared.</p>
</body>
</html>
```

OutPut

I am a *strong* man. I am a *strong* man.

I am a *strong* man. I am a *strong* man.

Example

Match all `<i>` elements in all first child `<p>` elements

In the following example, the selector matches all `<i>` elements in `<p>` elements that are the first child of another element:

```
<!DOCTYPE html>
<html>
<head>
<style type="text/css">
p:first-child i
{
color:blue;
}
</style>
</head>

<body>
<p>I am a <i>strong</i> man. I am a <i>strong</i> man.</p>
<p>I am a <i>strong</i> man. I am a <i>strong</i> man.</p>
<p><b>Note:</b> For :first-child to work in IE8 and earlier, a DOCTYPE must be
declared.</p>
</body>
</html>
```

OutPut

I am a *strong* man. I am a *strong* man.

I am a *strong* man. I am a *strong* man.

CSS Pseudo-elements

CSS pseudo-elements are used to add special effects to some selectors.

Syntax

The syntax of pseudo-elements:

```
selector:pseudo-element {property:value;}
```

CSS classes can also be used with pseudo-elements:

```
selector.class:pseudo-element {property:value;}
```


Multiple Pseudo-elements

Several pseudo-elements can also be combined.

In the following example, the first letter of a paragraph will be red, in an xx-large font size. The rest of the first line will be blue, and in small-caps. The rest of the paragraph will be the default font size and color:

```
<!DOCTYPE html>
<html>
<head>
<style type="text/css">
p:first-letter
{
color:#ff0000;
font-size:xx-large;
}
p:first-line
{
color:#0000ff;
font-variant:small-caps;
}
</style>
</head>

<body>
<p>You can combine the :first-letter and :first-line pseudo-elements to add a
special effect to the first letter and the first line of a text!</p>
</body>
</html>
```

YOU CAN COMBINE THE :FIRST-LETTER AND :FIRST-LINE PSEUDO-ELEMENTS TO ADD A SPECIAL effect to the first letter and the first line of a text!

All CSS Pseudo Classes/Elements

Selector	Example	Example description
<u>:link</u>	a:link	Selects all unvisited links
<u>:visited</u>	a:visited	Selects all visited links
<u>:active</u>	a:active	Selects the active link
<u>:hover</u>	a:hover	Selects links on mouse over
<u>:focus</u>	input:focus	Selects the input element which has focus
<u>:first-letter</u>	p:first-letter	Selects the first letter of every <p> element
<u>:first-line</u>	p:first-line	Selects the first line of every <p> element
<u>:first-child</u>	p:first-child	Selects every <p> elements that is the first child of its parent
<u>:before</u>	p:before	Insert content before every <p> element
<u>:after</u>	p:after	Insert content after every <p> element

CSS Attribute Selectors

Style HTML Elements With Specific Attributes

It is possible to style HTML elements that have specific attributes, not just class and id.

Attribute Selector

The example below styles all elements with a title attribute:

Example

```
[title]
{
  color:blue;
}
```

Attribute and Value Selector

```
<!DOCTYPE html>
<html>
<head>
<style type="text/css">
[title=test]
{
border:5px solid green;
}
</style>
</head>

<body>
<h2>Will apply to:</h2>

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

Styling Forms

The attribute selectors are particularly useful for styling forms without class or ID:

```
<!DOCTYPE html>
<html>
<head>
<style>
input[type="text"]
{
width:150px;
display:block;|
margin-bottom:10px;
background-color:yellow;
}
input[type="button"]
{
width:120px;
margin-left:35px;
display:block;
}
</style>
</head>
<body>

<form name="input" action="" method="get">
Firstname:<input type="text" name="Name" value="Peter" size="20">
Lastname:<input type="text" name="Name" value="Griffin" size="20">
<input type="button" value="Example Button">

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

Firstname:

Peter

Lastname:

Griffin

Example Button

Grouping Selectors

In style sheets there are often elements with the same style.

```
h1
{
color:green;
}
h2
{
color:green;
}
p
{
color:green;
}
```

To minimize the code, you can 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
{
color:green;
}
```

@Media

Media Types allow you to specify how documents will be presented in different media. The document can be displayed differently on the screen, on the paper, with an aural browser, etc.

The @media Rule

The @media rule allows different style rules for different media in the same style sheet.

The style in the example below tells the browser to display a 14 pixels Verdana font on the screen. But if the page is printed, it will be in a 10 pixels Times font. Notice that the font-weight is set to bold, both on screen and on paper:

```
<html>
<head>
<style>
@media screen
{
  p.test {font-family:verdana,sans-serif;font-size:14px;}
}
@media print
{
  p.test {font-family:times,serif;font-size:10px;}
}
@media screen,print
{
  p.test {font-weight:bold;}
}
</style>
</head>

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

- The following media types are available:
 - `all` (the default; styles are applied to all devices and outputs)
 - `aural` (for screen readers and browsers with voice capability)
 - `braille` (for output to a Braille tactile feedback device)
 - `embossed` (for paged Braille printers)
 - `handheld` (for mobile devices such as cell phones and PDAs)
 - `print` (for print output)
 - `projection` (for full-screen presentation slides or kiosks)
 - `screen` (for desktop browsers; the iPhone also uses this media type)
 - `speech` (added in CSS2.1 as a replacement for the `aural` media type, which was deprecated)
 - `tty` (for devices with a fixed-pitch character grid, such as a terminal screen)
 - `tv` (for televisions or similar devices)

@Rule in CSS

The `@import` rule is another way of loading a CSS file. You can use it in two ways. The

simplest is within the header of your document, like so:

```
1 <style>
2     @import url('/css/styles.css');
3 </style>
```

second

```
1 <link rel="stylesheet" type="text/css" href="/css/styles.css" />
```

- The `@import` rule isn't recognized by the really old browsers out there... Netscape Navigator 4 skips over `@imports` entirely, and Internet Explorer 4 ignores them if you don't use the (entirely optional, but generally used) parentheses. Of course, not too many people these days use either of these archaic browsers. And this problem can actually be *useful* if you'd like to hide some or all of your CSS from the browsers that can't really support it.
- Your `@imports` **must** come before all other content in your CSS. And I mean *all* of your content. Even putting comments before the `@import` tag will cause your imports to fail. So be sure to do your imports before you do anything else.

CSS BOX Model

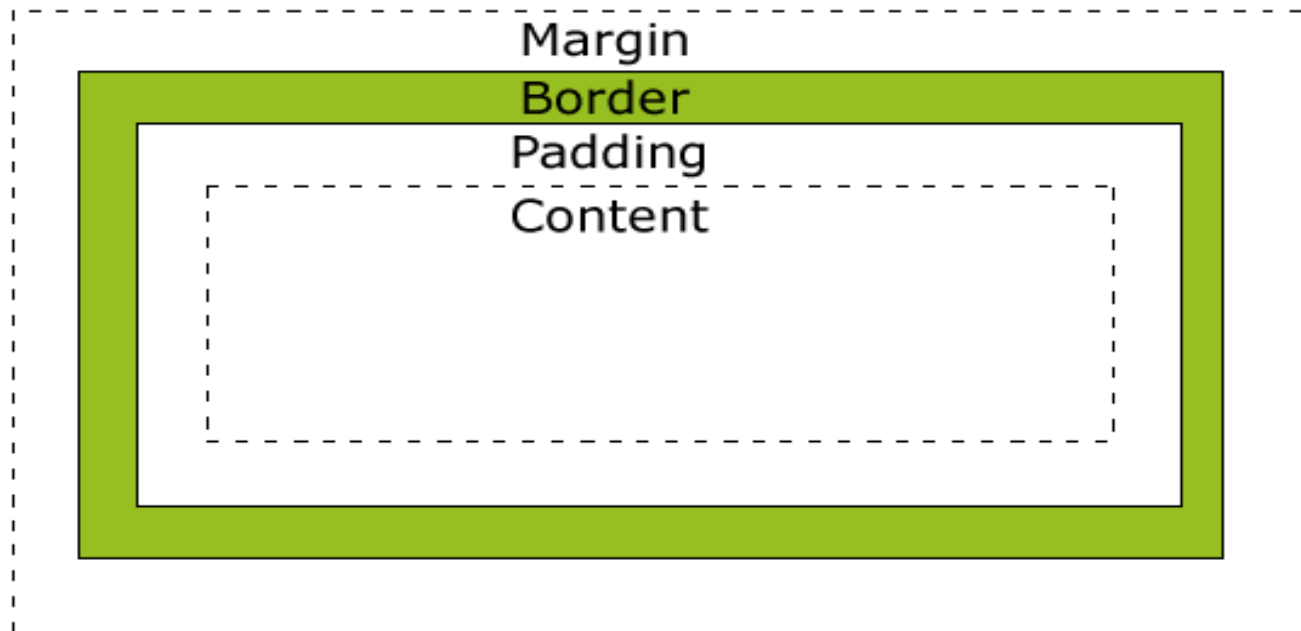
The CSS Box Model

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 place a border around elements and space elements in relation to other elements.

The image below illustrates the box model:



Explanation of the different parts:

- **Margin** - Clears an area around the border. The margin does not have a background color, it is completely transparent
- **Border** - A border that goes around the padding and content. The border is affected by the background color of the box
- **Padding** - Clears an area around the content. The padding is affected by the background color of the box
- **Content** - The content of the box, where text and images appear

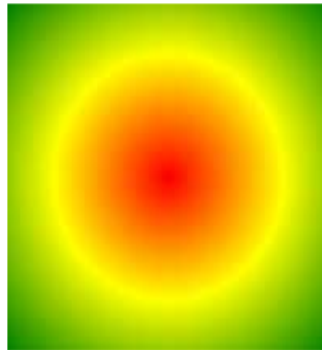
CSS3 Gradients

- CSS3 defines two types of gradients:
- **Linear Gradients** (goes down/up/left/right/diagonally)
- **Radial Gradients** (defined by their center)

Example of Linear Gradient:



Example of Radial Gradient:



Syntax

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

Syntax

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

Linear Gradients

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 */  
}
```

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 */  
}
```

Continue..

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 */  
}
```

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 */  
}
```

Using Transparency

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*/  
}
```

CSS3 Radial Gradients

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 */  
}
```


Continue...

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 */  
}
```