|  |
| --- |
| Day 2 ( CSS ) |

1. **CSS Overview**

**What is CSS?**



**C**ascading **S**tyle **S**heets, is a simple design language intendedto simplify the process of making web pages presentable.

CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, as well as a variety of other effects.

CSS is easy to learn and understand but it provides a powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

**Advantages of CSS**



* **CSS saves time** - You can write CSS once and then reuse the same sheet inmultiple HTML pages. You can define a style for each HTML element and apply it to as many web pages as you want.
* **Pages load faster** - If you are using CSS, you do not need to write HTML tagattributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So, less code means faster download times.
* **Easy maintenance** - To make a global change, simply change the style, and allthe elements in all the web pages will be updated automatically.
* **Superior styles to HTML** - CSS has a much wider array of attributes than HTML,so you can give a far better look to your HTML page in comparison to HTML attributes.
* **Multiple Device Compatibility** - Style sheets allow content to be optimized formore than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cellphones or for printing.
* **Global web standards** –Now HTML attributes are being deprecated and it isbeing recommended to use CSS. So it’s a good idea to start using CSS in all the HTML pages to make them compatible with future browsers.

**CSS Versions**



Each level of CSS builds upon the last, typically adding new features and typically denoted as CSS1, CSS2, and CSS3.

Profiles are typically a subset of All levels of CSS built for a particular device or user interface. Currently there are profiles for mobile devices, printers, and television sets. Profiles should not be confused with media types which were added in CSS2.

**CSS1**

Among its capabilities are support for:

* Font properties such as typeface and emphasis
* Color of text, backgrounds, and other elements
* Text attributes such as spacing between words, letters, and lines of text
* Alignment of text, images, tables and other elements
* Margin, border, padding, and positioning for most elements
* Unique identification and generic classification of groups of attributes

**CSS2**

* A superset of CSS1, CSS2 includes a number of new capabilities like absolute, relative, and fixed positioning of elements, the concept of media types, support for aural style sheets and bidirectional text, and new font properties such as shadows.
* CSS level 2 revision 1 or CSS 2.1 fixes errors in CSS2, removes poorly-supported features and adds already-implemented browser extensions to the specification.

**CSS3**

* CSS3 is divided into many different documents called Modules. Every module adds new capability or extends features defined in CSS2 over preserving backward compatibility. Work on CSS3 started around the time of publication of the original CSS2 recommendation.
* Because of the modularization in CSS3, every modules has different stability and is in different status.
* The CSS3 version supports many more browsers than CSS2.
* CSS3 has other added features such as new combinator, new CSS selectors, new pseudo-elements and new style properties.

1. **CSS ─ Syntax**

A style rule is made of three parts:

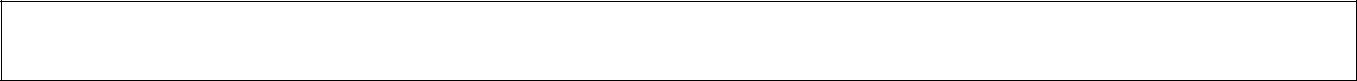
* **Selector:** A selector is an HTML tag at which a style will be applied. This could beany tag like <h1> or <table> etc.
* **Property:** A property is a type of attribute of HTML tag. Put simply, all the HTMLattributes are converted into CSS properties. They could be *color*, *border*, etc.
* **Value:** Values are assigned to properties. For example,*color*property can havethe value either *red* or *#F1F1F1* etc.

You can put CSS Style Rule Syntax as follows:



selector { property: value }

**Example:** You can define a table border as follows:



table{ border :1px solid #C00; }

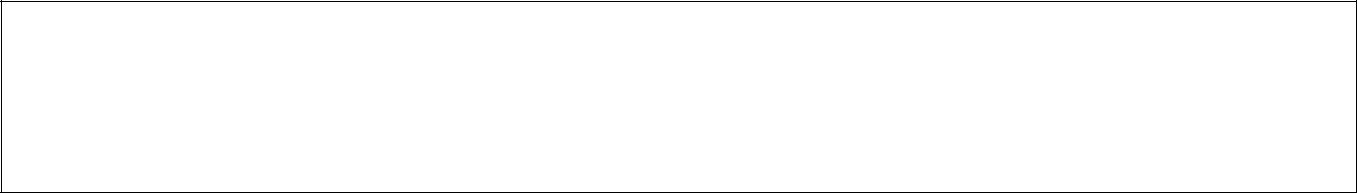
Here table is a selector and border is a property and the given value *1px solid #C00* is the value of that property.

You can define selectors in various simple ways based on your comfort. Let me put these selectors one by one.

**The Type Selectors**



This is the same selector we have seen above. Again, one more example to give a color to all level 1 headings:



h1 {

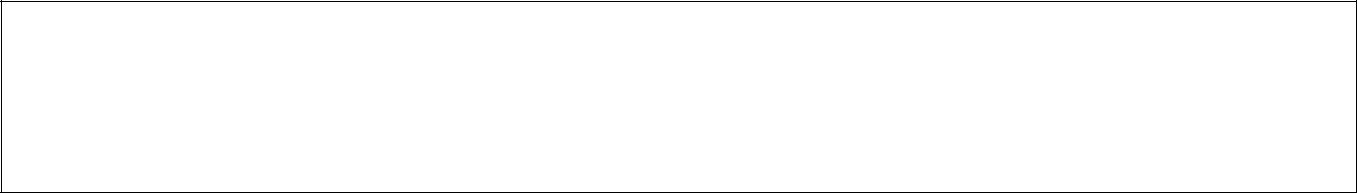
color: #36CFFF;

}

**The Descendant Selectors**



Suppose you want to apply a style rule to a particular element only when it lies inside a particular element. As given in the following example, the style rule will apply to <em> element only when it lies inside the <ul> tag.



ul em {

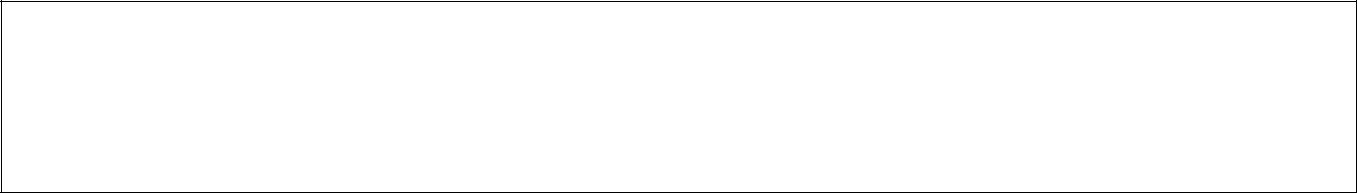
color: #000000;

}

**The Class Selectors**



You can define style rules based on the class attribute of the elements. All the elements having that class will be formatted according to the defined rule.

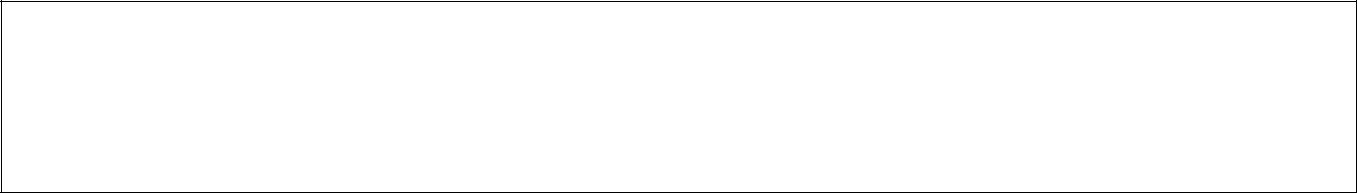


.black {

color: #000000;

}

This rule renders the content in black for every element with class attribute set to *black* in our document. You can make it a bit more particular. For example:



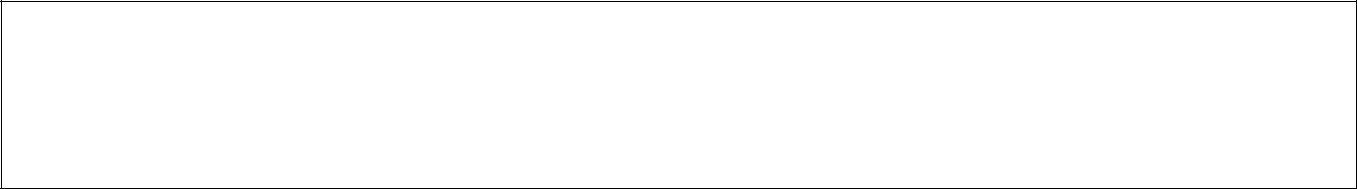
h1.black {

color: #000000;

}

This rule renders the content in black for only <h1> elements with class attribute set to *black*.

You can apply more than one class selectors to a given element. Consider the following example:

<p class="center bold">

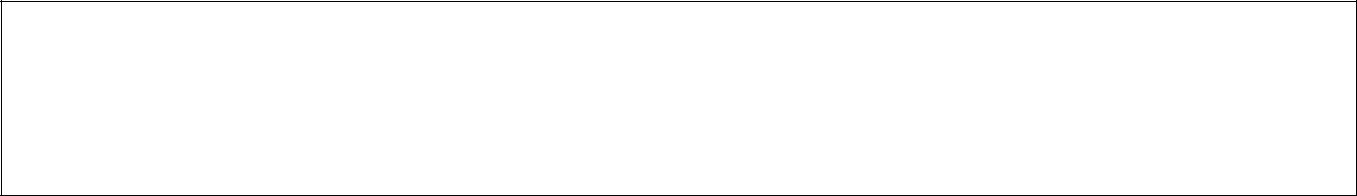
This para will be styled by the classes center and bold.

</p>

**The ID Selectors**



You can define style rules based on the *id* attribute of the elements. All the elements having that *id* will be formatted according to the defined rule.

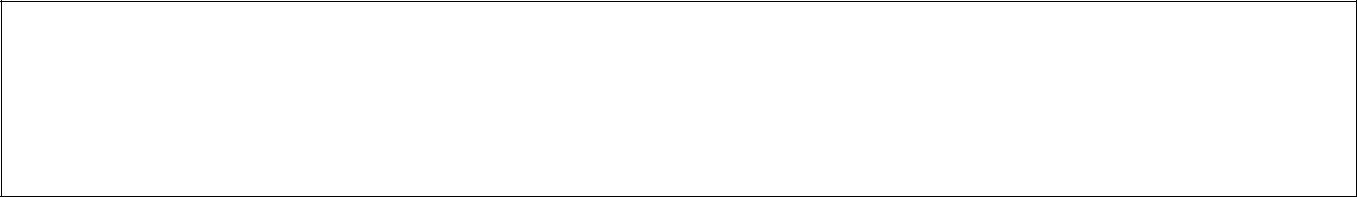


#black {

color: #000000;

}

This rule renders the content in black for every element with *id* attribute set to *black* in our document. You can make it a bit more particular. For example:



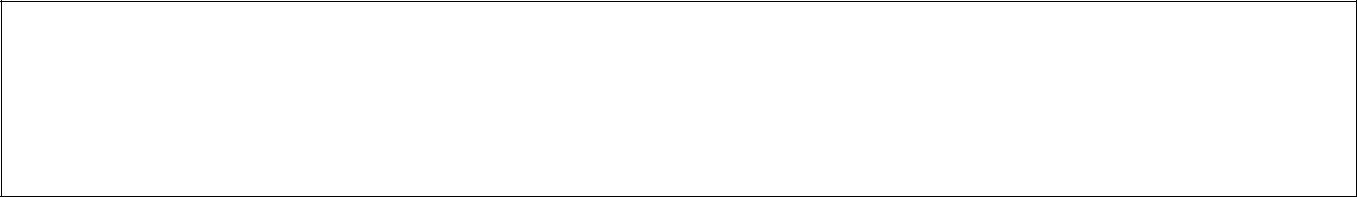
h1#black {

color: #000000;

}

This rule renders the content in black for only <h1> elements with *id* attribute set to *black*.

The true power of *id* selectors is when they are used as the foundation for descendant selectors. For example:



#black h2 {

color: #000000;

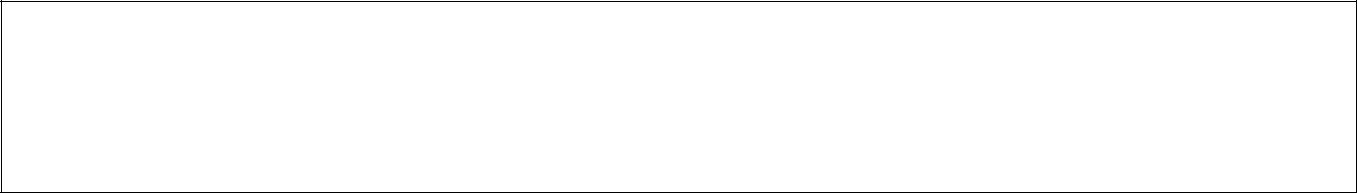
}

In this example, all level 2 headings will be displayed in black color when those headings will lie within tags having *id* attribute set to *black*.

**The Child Selectors**



You have seen the descendant selectors. There is one more type of selector, which is very similar to descendants but have different functionality. Consider the following example:



body > p {

color: #000000;

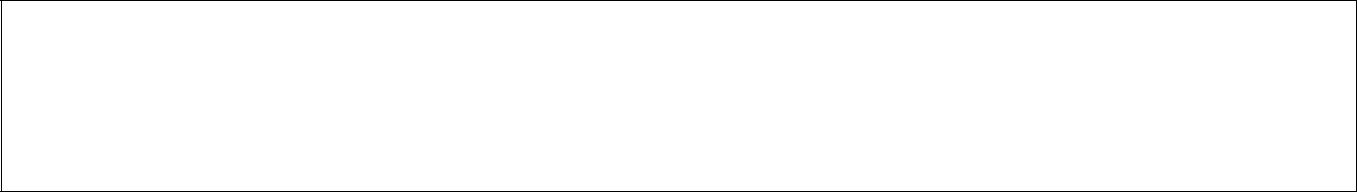
}

This rule will render all the paragraphs in black if they are a direct child of the <body> element. Other paragraphs put inside other elements like <div> or <td> would not have any effect of this rule.

**The Attribute Selectors**



You can also apply styles to HTML elements with particular attributes. The style rule below will match all the input elements having a type attribute with a value of *text*:



input[type="text"]{

color: #000000;

}

The advantage to this method is that the <input type="submit" /> element is unaffected, and the color applied only to the desired text fields.

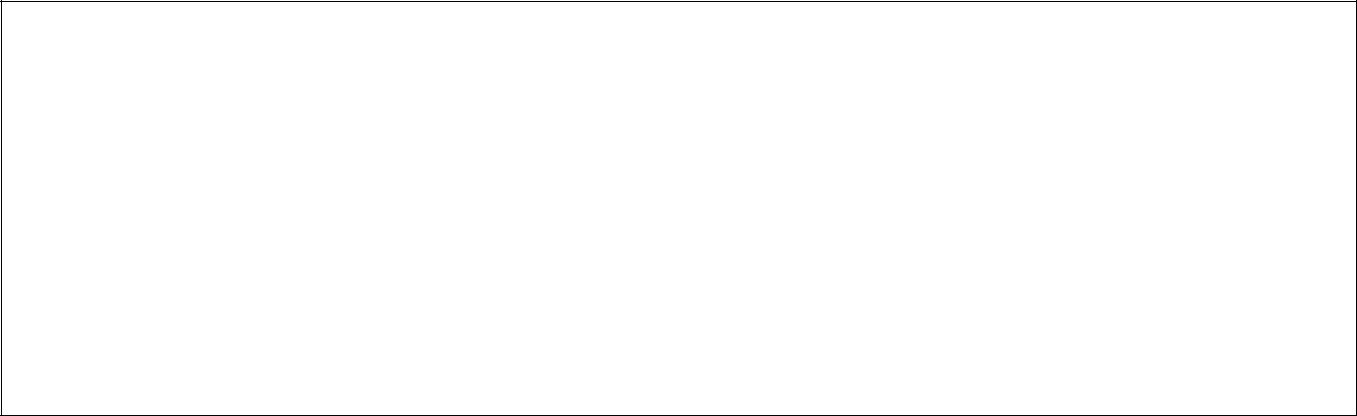
There are following rules applied to attribute selector.

* **p[lang]** - Selects all paragraph elements with a*lang*attribute.
* **p[lang="fr"]** - Selects all paragraph elements whose*lang*attribute has a valueof exactly "fr".
* **p[lang~="fr"]** - Selects all paragraph elements whose*lang*attribute containsthe word "fr".
* **p[lang|="en"]** - Selects all paragraph elements whose*lang*attribute containsvalues that are exactly "en", or begin with "en-".

**Multiple Style Rules**



You may need to define multiple style rules for a single element. You can define these rules to combine multiple properties and corresponding values into a single block as defined in the following example:



h1 {

color: #36C; font-weight: normal; letter-spacing: .4em; margin-bottom: 1em;

text-transform: lowercase;

}

Here all the property and value pairs are separated by a **semicolon (;)**. You can keep them in a single line or multiple lines.

For better readability, we keep them in separate lines.

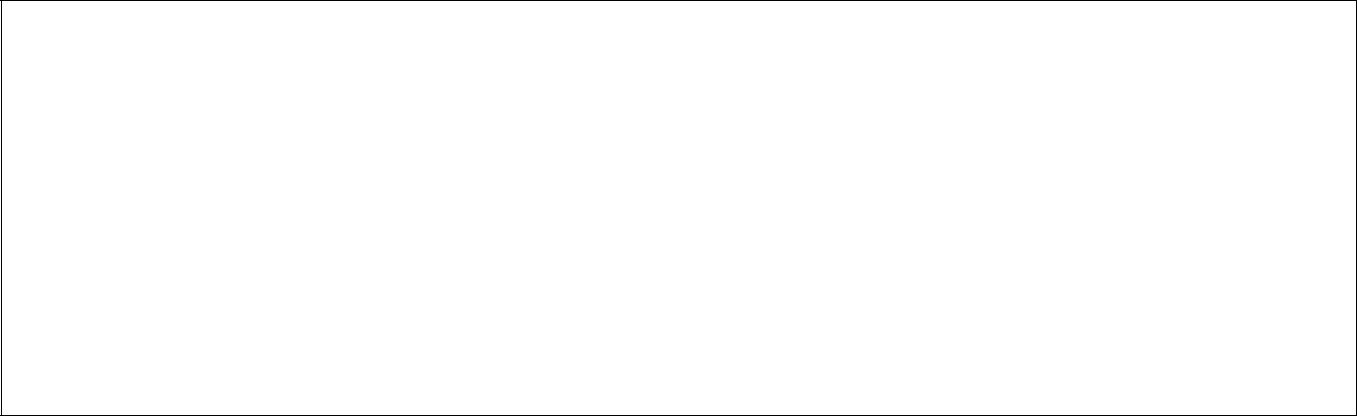
For a while, don't bother about the properties mentioned in the above block.

These properties will be explained in the coming sections and you can find the complete detail about properties in CSS References.

**Grouping Selectors**



You can apply a style to many selectors if you like. Just separate the selectors with a comma, as given in the following example:



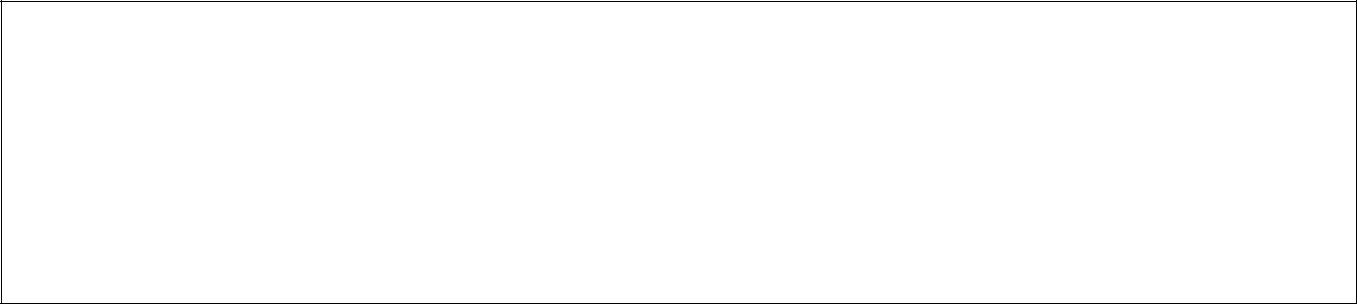
h1, h2, h3 { color: #36C;

font-weight: normal; letter-spacing: .4em; margin-bottom: 1em; text-transform: lowercase;

}

This define style rule will be applicable to h1, h2 and h3 element as well. The order of the list is irrelevant. All the elements in the selector will have the corresponding declarations applied to them.

You can combine the various *class* selectors together as shown below:



#content, #footer, #supplement {

position: absolute;

left: 510px;

width: 200px;

}

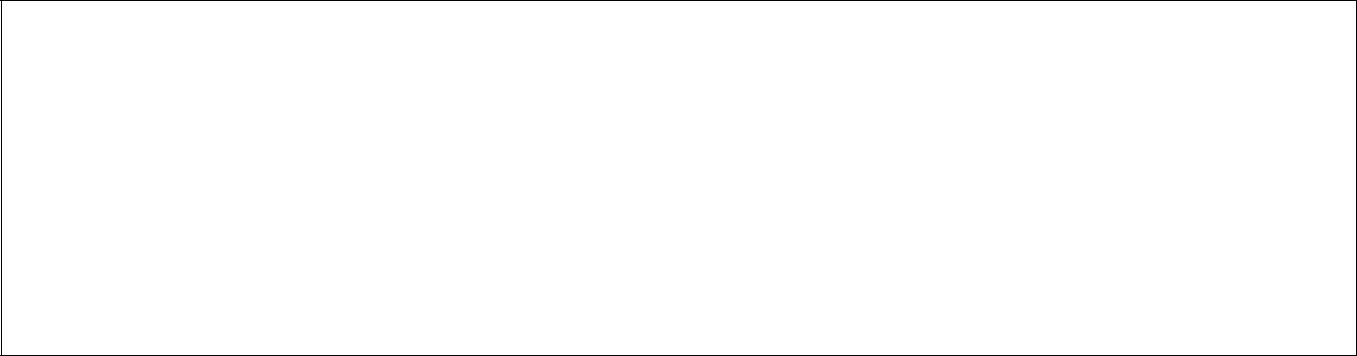
1. **CSS ─ Inclusion**

There are four ways to associate styles with your HTML document. Most commonly used methods are inline CSS and External CSS.

**Embedded CSS - The <style> Element**



You can put your CSS rules into an HTML document using the <style> element. This tag is placed inside the <head>...</head> tags. Rules defined using this syntax will be applied to all the elements available in the document. Here is the generic syntax:



<head>

<style type="text/css" media="...">

Style Rules

............

</style>

</head>

**Attributes**

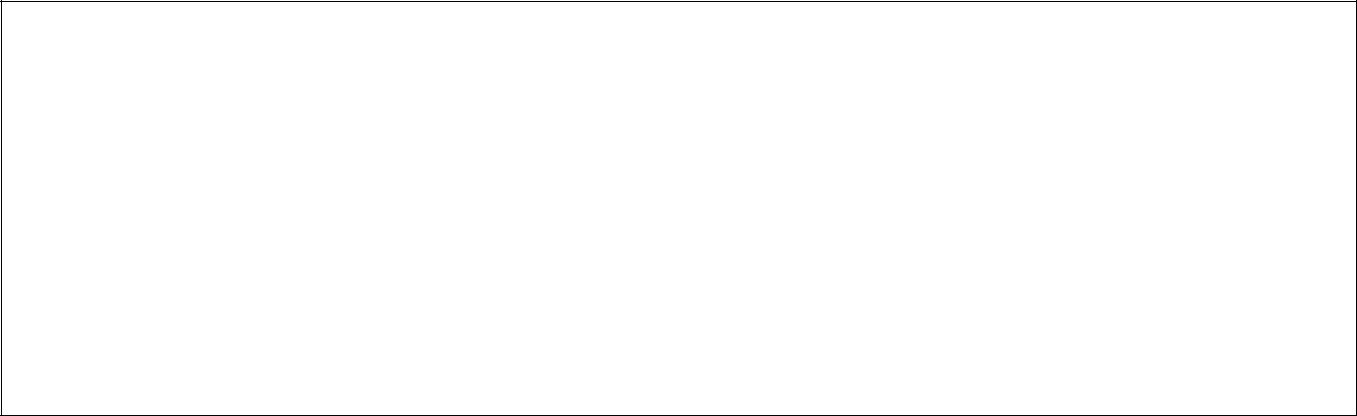


Attributes associated with <style> elements are:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Attribute** |  |  | **Value** |  |  | **Description** |  |  |
|  |  |  |  |  |  |  |
| type | | |  | text/css | |  | Specifies the style sheet language as a content-type | |  |
|  |  |  |  |  |  |  | (MIME type). This is a required attribute. | |  |
|  | | |  |  | |  |  | |  |
| media | | |  | screen | |  | Specifies the device, the document will be displayed | |  |
|  |  |  |  | tty | |  | on.  Default value is *all*. This is an optional attribute. | |  |
|  |  |  |  | tv | |  |  |  |  |
|  |  |  |  | projection | |  |  |  |  |
|  |  |  |  | handheld | |  |  |  |  |
|  |  |  |  | print | |  |  |  |  |
|  |  |  |  | braille | |  |  |  |  |
|  |  |  |  | aural | |  |  |  |  |
|  |  |  |  | all | |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

**Example**

Following is an example of embed CSS based on the above syntax:



<head>

<style type="text/css" media="all">

h1{

color: #36C;

}

</style>

</head>

**Inline CSS - The *style* Attribute**



You can use *style* attribute of any HTML element to define style rules. These rules will be applied to that element only. Here is the generic syntax:



<element style="...style rules....">

**Attributes**



|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| style | style | The value of *style* attribute is a combination of style |
|  | rules | declarations separated by semicolon (;). |
|  |  |  |

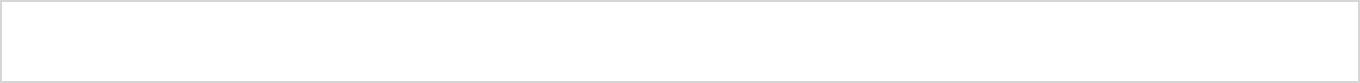
**Example**

Following is the example of inline CSS based on the above syntax:



<h1 style ="color:#36C;"> This is inline CSS </h1>

It will produce the following result:



**This is inline CSS**

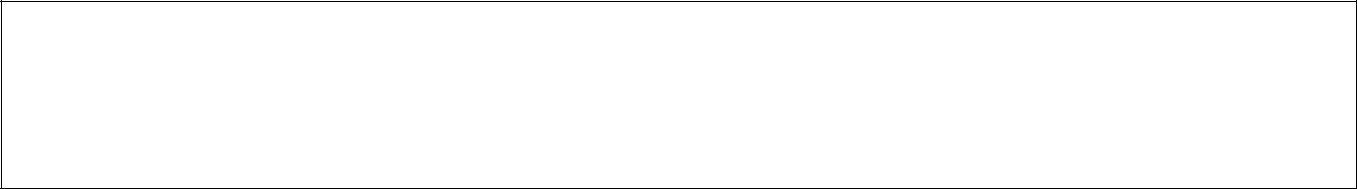
**External CSS - The <link> Element**



The <link> element can be used to include an external stylesheet file in your HTML document.

An external style sheet is a separate text file with **.css** extension. You define all the Style rules within this text file and then you can include this file in any HTML document using <link> element.

Here is the generic syntax of including external CSS file:



<head>

<link type="text/css" href="..." media="..." />

</head>

**Attributes**

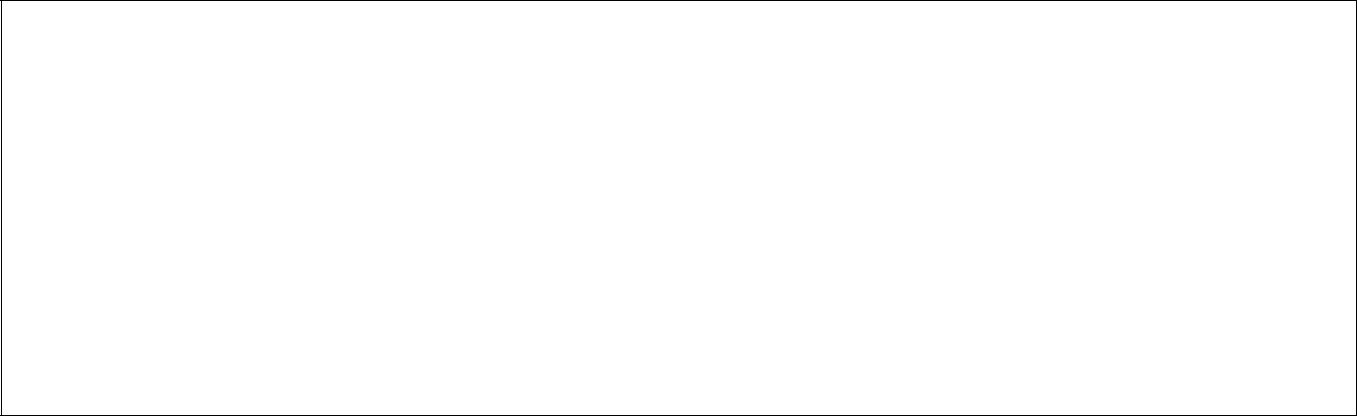


Attributes associated with <style> elements are:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| type | text/css | Specifies the style sheet language as a content-type (MIME |
|  |  | type). This attribute is required. |
|  |  |  |
| href | URL | Specifies the style sheet file having Style rules. This attribute |
|  |  | is a required. |
|  |  |  |
| media | screen | Specifies the device the document will be displayed on. |
|  | tty | Default value is *all*. This is an optional attribute. |
|  | tv |  |
|  | projection |  |
|  | handheld |  |
|  | print |  |
|  | braille |  |
|  | aural |  |
|  | all |  |
|  |  |  |

**Example**

Consider a simple style sheet file with a name *mystyle.css* having the following rules:

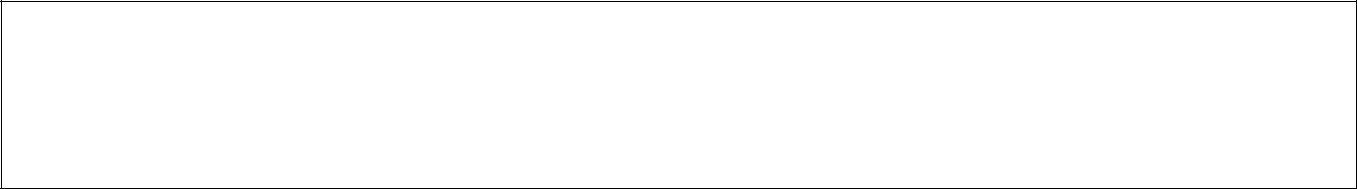


h1, h2, h3 { color: #36C;

font-weight: normal; letter-spacing: .4em; margin-bottom: 1em; text-transform: lowercase;

}

Now you can include this file *mystyle.css* in any HTML document as follows:



<head>

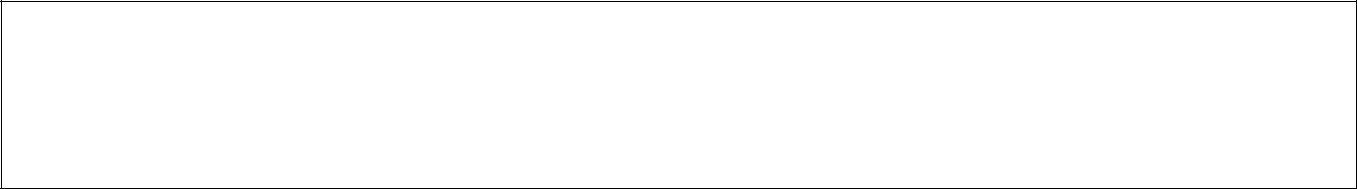
<link type="text/css" href="mystyle.css" media="all" />

</head>

**Imported CSS - @import Rule**



@import is used to import an external stylesheet in a manner similar to the <link> element. Here is the generic syntax of @import rule.

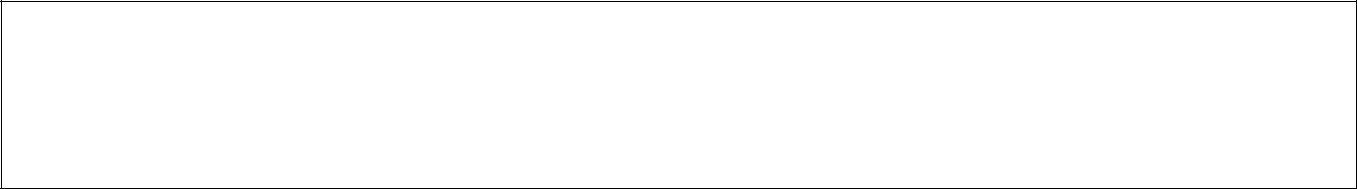


<head>

<@import "URL";

</head>

Here URL is the URL of the style sheet file having style rules. You can use another syntax as well:



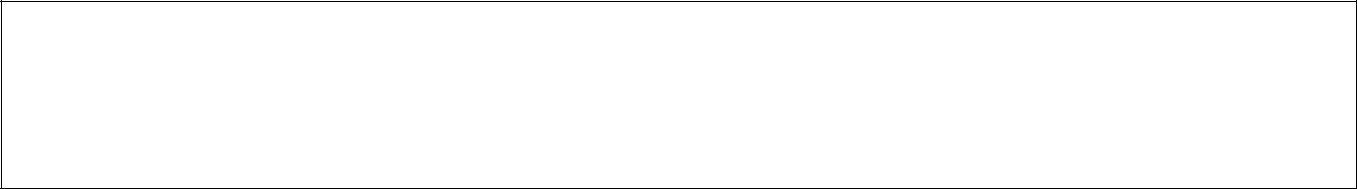
<head>

<@import url("URL");

</head>

**Example**

Following is the example showing you how to import a style sheet file into an HTML document:



<head>

@import "mystyle.css";

</head>

**CSS Rules Overriding**



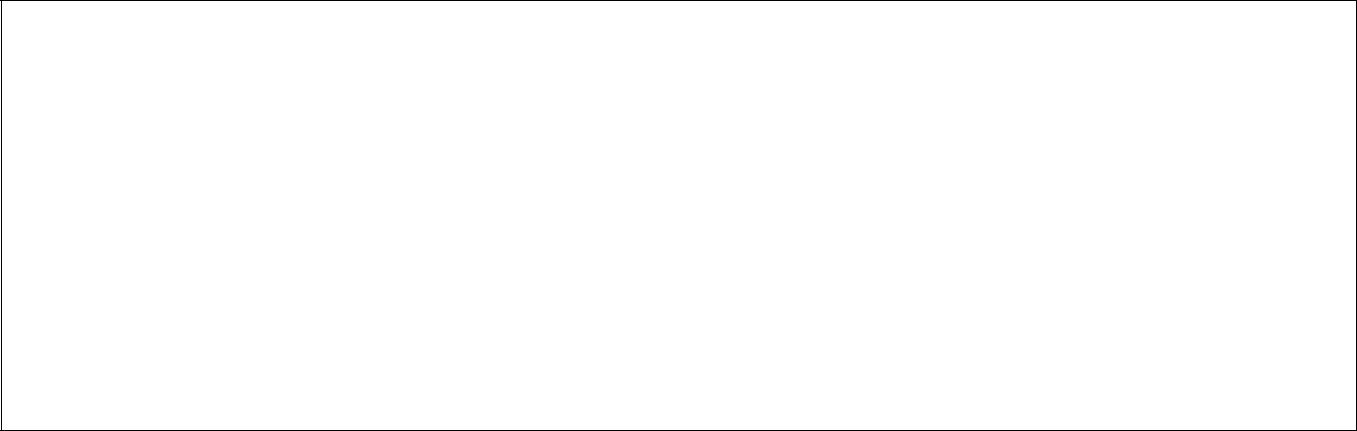
We have discussed four ways to include style sheet rules in an HTML document. Here is the rule to override any Style Sheet Rule.

* Any inline style sheet takes the highest priority. So, it will override any rule defined in <style>...</style> tags or the rules defined in any external style sheet file.
* Any rule defined in <style>...</style> tags will override the rules defined in any external style sheet file.
* Any rule defined in the external style sheet file takes the lowest priority, and the rules defined in this file will be applied only when the above two rules are not applicable.

**Handling Old Browsers**



There are still many old browsers who do not support CSS. So, we should take care while writing our Embedded CSS in an HTML document. The following snippet shows how to use comment tags to hide CSS from older browsers:



<style type="text/css">

<!--

body, td {

color: blue;

}

-->

</style>

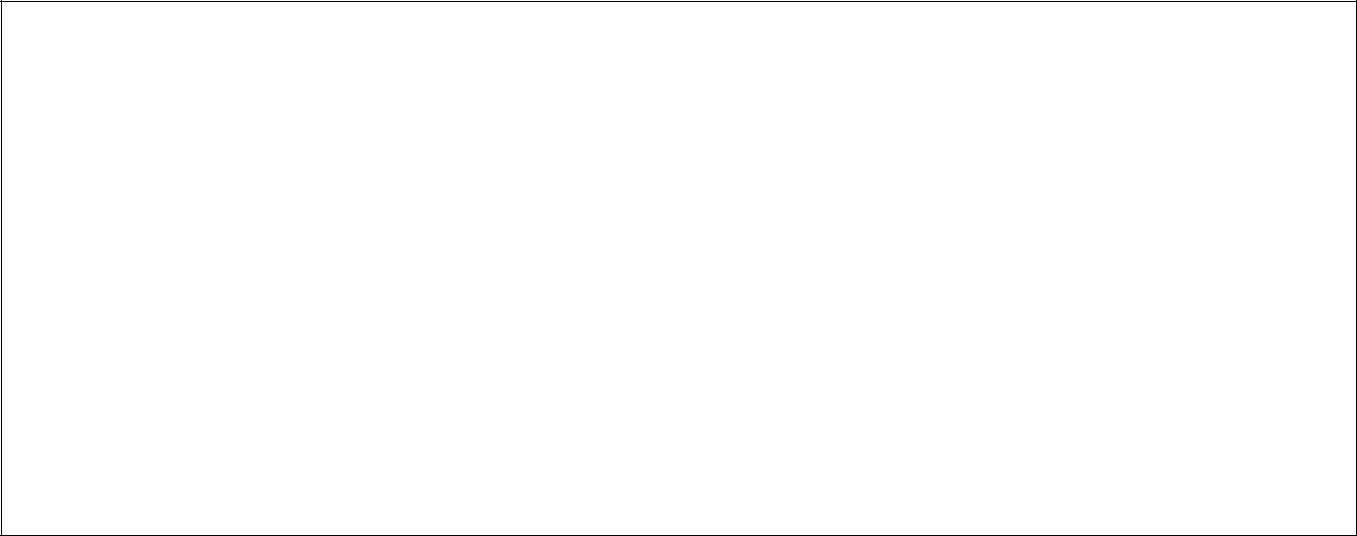
**CSS Comments**



Many times, you may need to put additional comments in your style sheet blocks. So, it is very easy to comment any part in the style sheet. You can simply put your comments inside /\*.....this is a comment in style sheet.....\*/.

You can use /\* ....\*/ to comment multi-line blocks in similar way you do in C and C++ programming languages.

**Example**



/\* This is an external style sheet file \*/

h1, h2, h3 { color: #36C;

font-weight: normal; letter-spacing: .4em; margin-bottom: 1em; text-transform: lowercase;

}

/\* end of style rules. \*/

1. **CSS ─ Measurement Units 4. CSS ─ Measurement Units**

Before we start the actual exercise, we would like to give a brief idea about the CSS Measurement Units. CSS supports a number of measurements including absolute units such as inches, centimeters, points, and so on, as well as relative measures such as percentages and em units. You need these values while specifying various measurements in your Style rules e.g. **border="1px solid red"**.

We have listed out all the CSS Measurement Units along with proper Examples:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Unit** |  |  | **Description** |  |  | **Example** | |  |  |
|  |  |  | Defines a measurement as a percentage |  | p | {font-size: | 16pt; | line-height: |  |
| % |  |  | relative to another value, typically an |  |  |
|  |  | 125%;} | | |  |  |  |
|  |  |  | enclosing element. |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  | |  |  |  |  | | | |  |
| cm | |  | Defines a measurement in centimeters. |  | div {margin-bottom: 2cm;} | | | |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  | A relative measurement for the height of |  |  |  |  |  |  |
|  |  |  | a font in em spaces. Because an em unit |  |  |  |  |  |  |
| em | |  | is equivalent to the size of a given font, |  | p {letter-spacing: 7em;} | | |  |  |
|  | if you assign a font to 12pt, each "em" |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  | unit would be 12pt; thus, 2em would be |  |  |  |  |  |  |
|  |  |  | 24pt. |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  | This value defines a measurement |  |  |  |  |  |  |
| ex | |  | relative to a font's x-height. The x- |  | p | {font-size: | 24pt; | line-height: |  |
|  | height is determined by the height of the |  | 3ex;} | |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  | font's lowercase letter x. |  |  |  |  |  |  |
|  | |  |  |  |  | | | |  |
| in | |  | Defines a measurement in inches. |  | p {word-spacing: .15in;} | | | |  |
|  | |  |  |  |  | | | |  |
| mm | |  | Defines a measurement in millimeters. |  | p {word-spacing: 15mm;} | | | |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  | Defines a measurement in picas. A pica |  |  |  |  |  |  |
| pc | |  | is equivalent to 12 points; thus, there |  | p {font-size: 20pc;} | | |  |  |
|  |  |  | are 6 picas per inch. |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| pt | |  | Defines a measurement in points. A |  | body {font-size: 18pt;} | | |  |  |
|  | point is defined as 1/72nd of an inch. |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  | |  |  |  |  | | |  |  |
| px | |  | Defines a measurement in screen pixels. |  | p {padding: 25px;} | | |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

1. **CSS ─ Colors CS Colors**

CSS uses color values to specify a color. Typically, these are used to set a color either for the foreground of an element (i.e., its text) or for the background of the element. They can also be used to affect the color of borders and other decorative effects.

You can specify your color values in various formats. Following table lists all the possible formats:

|  |  |  |
| --- | --- | --- |
| **Format** | **Syntax** | **Example** |
| Hex Code | #RRGGBB | p{color:#FF0000;} |
|  |  |  |
| Short Hex Code | #RGB | p{color:#6A7;} |
|  |  |  |
| RGB % | rgb(rrr%,ggg%,bbb%) | p{color:rgb(50%,50%,50%);} |
|  |  |  |
| RGB Absolute | rgb(rrr,ggg,bbb) | p{color:rgb(0,0,255);} |
|  |  |  |
| keyword | aqua, black, etc. | p{color:teal;} |
|  |  |  |

These formats are explained in more detail in the following sections:

**CSS Colors - Hex Codes**



A hexadecimal is a 6 digit representation of a color. The first two digits (RR) represent a red value, the next two are a green value (GG), and the last are the blue value (BB).

A hexadecimal value can be taken from any graphics software like Adobe Photoshop, Jasc Paintshop Pro, or even using Advanced Paint Brush.

Each hexadecimal code will be preceded by a pound or hash sign ‘#’. Following are the examples to use Hexadecimal notation.



|  |  |  |
| --- | --- | --- |
| **Color** |  | **Color HEX** |

#000000

#FF0000

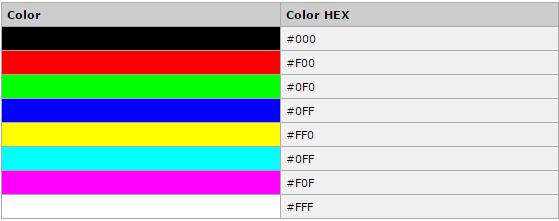
**CSS Colors - Short Hex Codes**



This is a shorter form of the six-digit notation. In this format, each digit is replicated to arrive at an equivalent six-digit value. For example: #6A7 becomes #66AA77.

A hexadecimal value can be taken from any graphics software like Adobe Photoshop, Jasc Paintshop Pro or even using Advanced Paint Brush.

Each hexadecimal code will be preceded by a pound or hash sign #. Following are the examples to use the Hexadecimal notation.



**CSS Colors - RGB Values**



This color value is specified using the **rgb( )** property. This property takes three values, one each for red, green, and blue. The value can be an integer between 0 and 255 or a percentage.

**NOTE:** All the browsers does not support rgb() property of color, so it is recommendednot to use it.

Following is the example to show few colors using RGB values.

|  |  |
| --- | --- |
| **Color** | **Color RGB** |
|  | rgb(0,0,0) |
|  |  |
|  |  |
|  | rgb(255,0,0) |
|  |  |
|  | rgb(0,255,0) |
|  |  |
|  |  |
|  | rgb(0,0,255) |
|  |  |
|  |  |
|  | rgb(255,255,0) |
|  |  |
|  |  |
|  | rgb(0,255,255) |
|  |  |
|  |  |
|  | rgb(255,0,255) |
|  |  |
|  |  |
|  | rgb(192,192,192) |
|  |  |
|  |  |
|  | rgb(255,255,255) |
|  |  |

**Building Color Codes**



You can build millions of color codes using our Color Code Builder. Check our **HTML** **Color Code Builder**. To use this tool, you would need a Java Enabled Browser.

**Browser Safe Colors**



Here is the list of 4 colors, which are supposed to be most safe and computer independent colors. These colors vary from hexa code 000000 to FFFFFF.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 000000 |  |  | 000033 |  |  | 000066 |  |  | 000099 |  |  | 0000CC |  |  | 0000FF |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 003300 |  |  | 003333 |  |  | 003366 |  |  | 003399 |  |  | 0033CC |  |  | 0033FF |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 006600 |  |  | 006633 |  |  | 006666 |  |  | 006699 |  |  | 0066CC |  |  | 0066FF |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 009900 |  |  | 009933 |  |  | 009966 |  |  | 009999 |  |  | 0099CC |  |  | 0099FF |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 00CC00 |  |  | 00CC33 |  |  | 00CC66 |  |  | 00CC99 |  |  | 00CCCC |  |  | 00CCFF |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

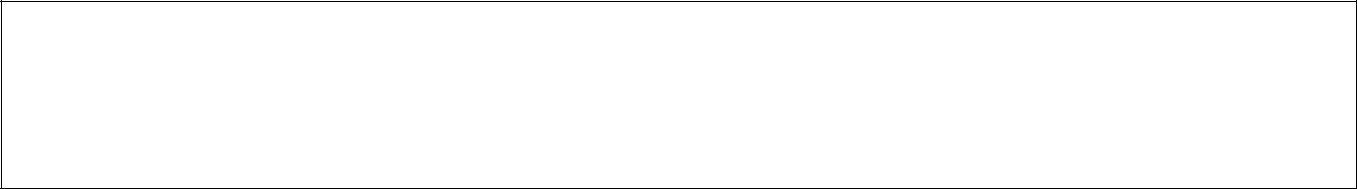
1. **CSS ─ Background**

This section teaches you how to set backgrounds of various HTML elements. You can set the following background properties of an element:

* The **background-color** property is used to set the background color of an element.
* The **background-image** property is used to set the background image of an element.
* The **background-repeat** property is used to control the repetition of an image in the background.
* The **background-position** property is used to control the position of an image in the background.
* The **background-attachment** property is used to control the scrolling of an image in the background.
* The **background** property is used as a shorthand to specify a number of other background properties.

**Set the Background Color**

Following is the example, which demonstrates how to set the background color for an element.



<p style="background-color:yellow;">

This text has a yellow background color.

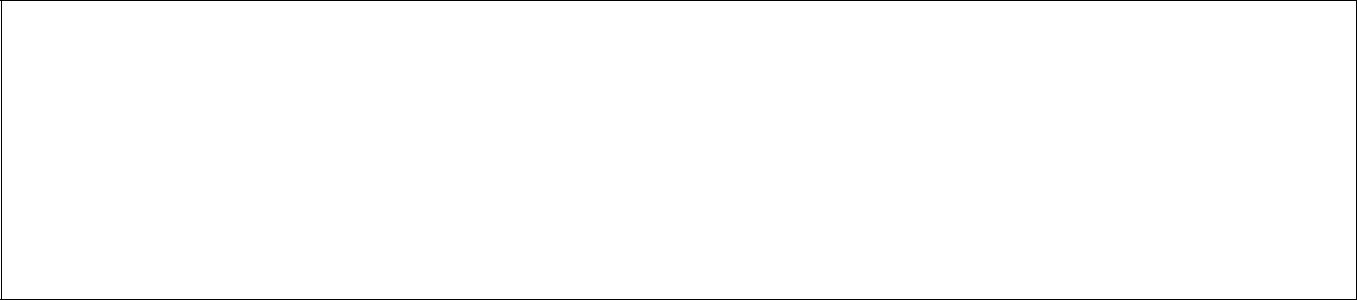
</p>

It will produce the following result:



This text has a yellow background color.

**Set the Background Image**



<table style="background-image:url(/images/pattern1.gif);">

<tr><td>

This table has background image set.

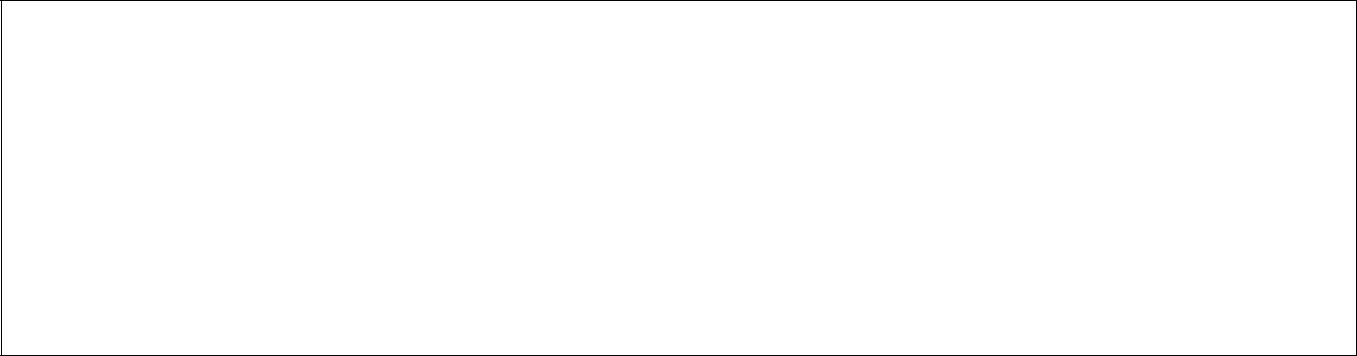
</td></tr>

</table>

**Repeat the Background Image**

The following example demonstrates how to repeat the background image if an image is small. You can use *no-repeat* value for the *background-repeat* property if you don't want to repeat an image. In this case, the image will display only once.

By default, the *background-repeat* property will have a *repeat* value.

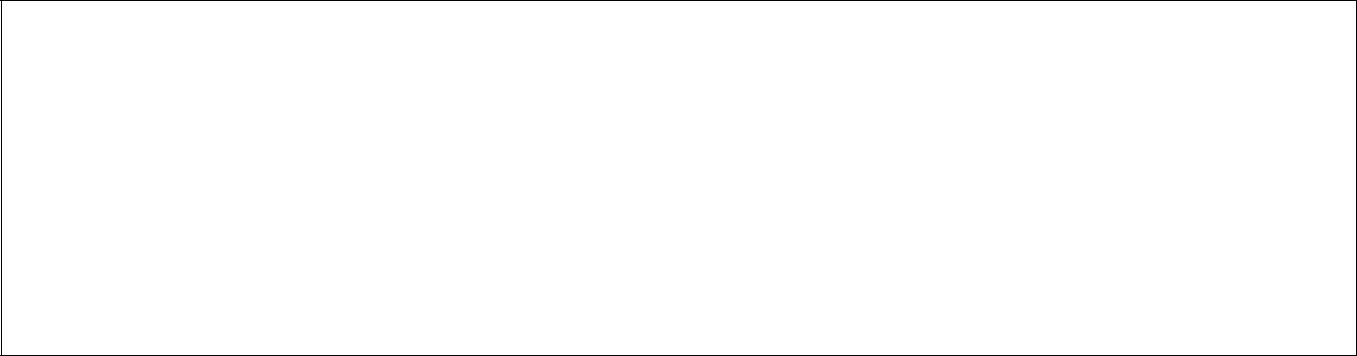
<table style="background-image:url(/images/pattern1.gif); background-repeat: repeat;">

<tr><td>

This table has background image which repeats multiple times. </td></tr>

</table>

The following example which demonstrates how to repeat the background image vertically.



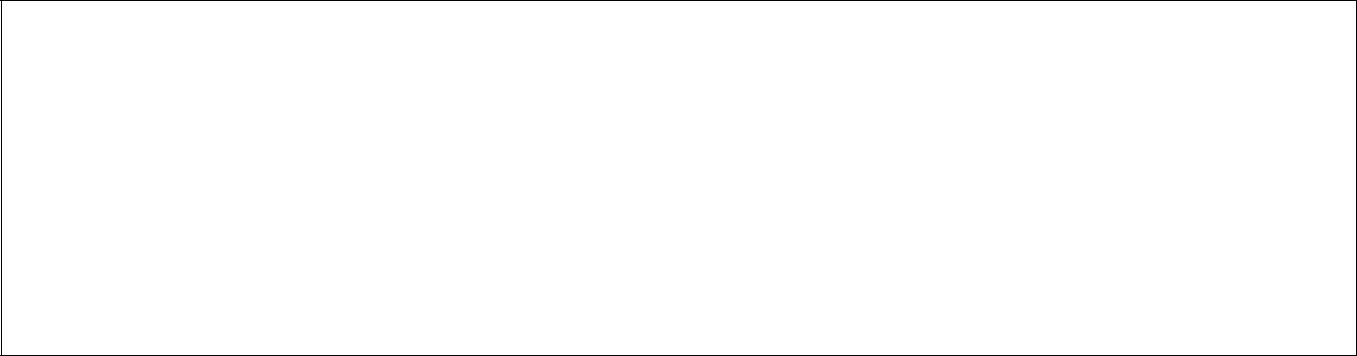
<table style="background-image:url(/images/pattern1.gif); background-repeat: repeat-y;">

<tr><td>

This table has background image set which will repeat vertically. </td></tr>

</table>

The following example demonstrates how to repeat the background image horizontally.



<table style="background-image:url(/images/pattern1.gif); background-repeat: repeat-x;">

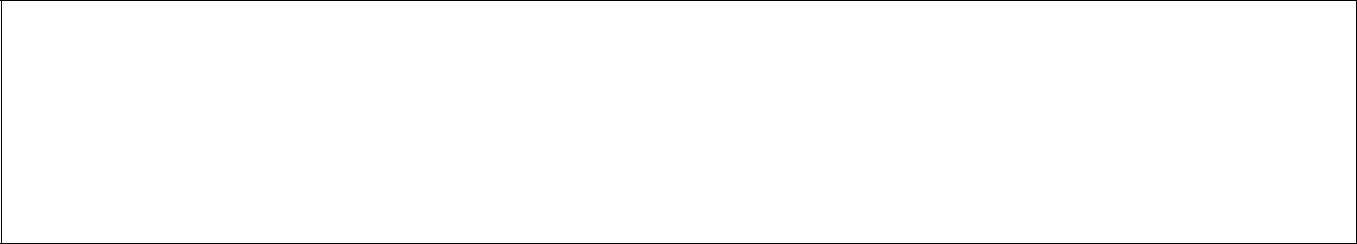
<tr><td>

This table has background image set which will repeat horizontally. </td></tr>

</table>

**Set the Background Image Position**

The following example demonstrates how to set the background image position 100 pixels away from the left side.



<table style="background-image:url(/images/pattern1.gif); background-position:100px;">

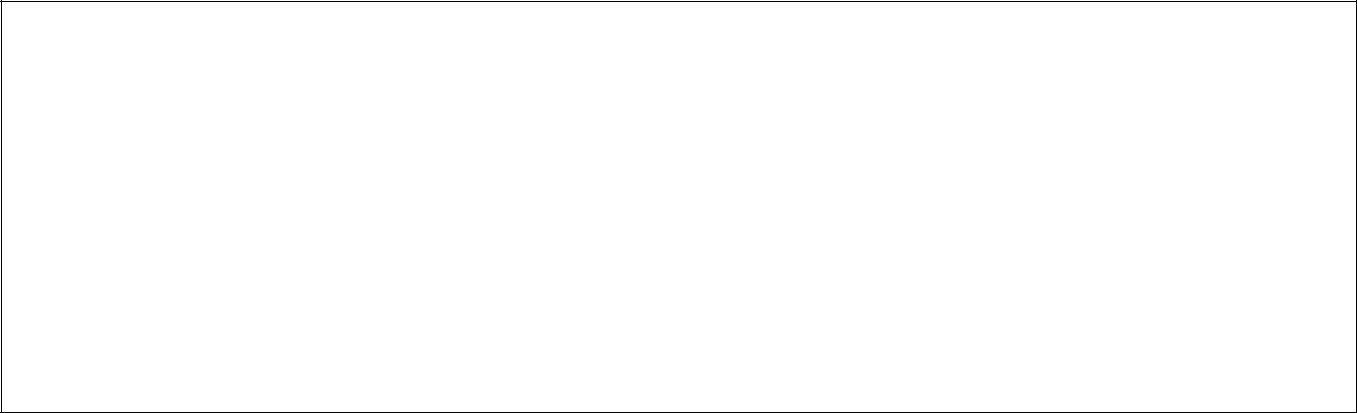
<tr><td>

Background image positioned 100 pixels away from the left.

</td></tr>

</table>

The following example demonstrates how to set the background image position 100 pixels away from the left side and 200 pixels down from the top.



<table style="background-image:url(/images/pattern1.gif); background-position:100px 200px;">

<tr><td>

This table has background image positioned 100

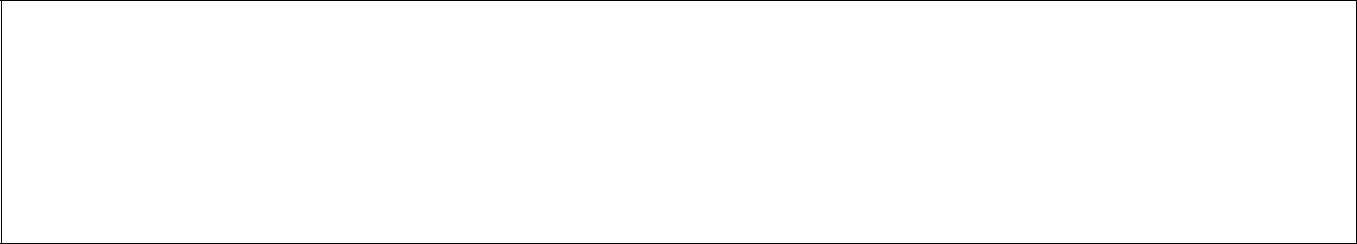
pixels away from the left and 200 pixels from the top. </td></tr>

</table>

**Set the Background Attachment**

Background attachment determines whether a background image is fixed or scrolls with the rest of the page.

The following example demonstrates how to set the fixed background image.



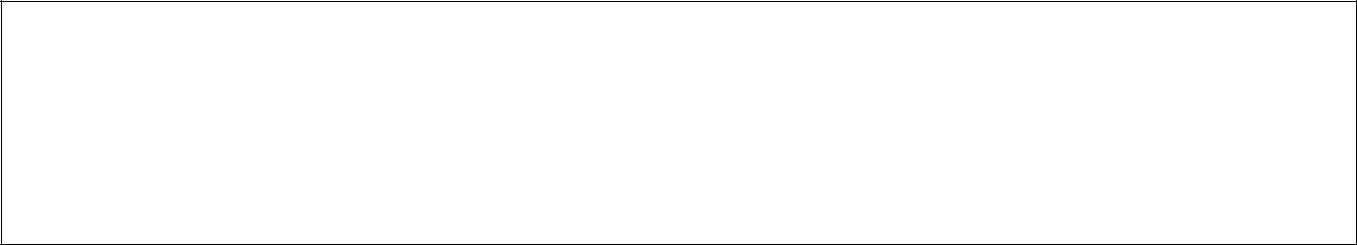
<p style="background-image:url(/images/pattern1.gif);

background-attachment:fixed;">

This parapgraph has fixed background image.

</p>

The following example demonstrates how to set the scrolling background image.



<p style="background-image:url(/images/pattern1.gif);

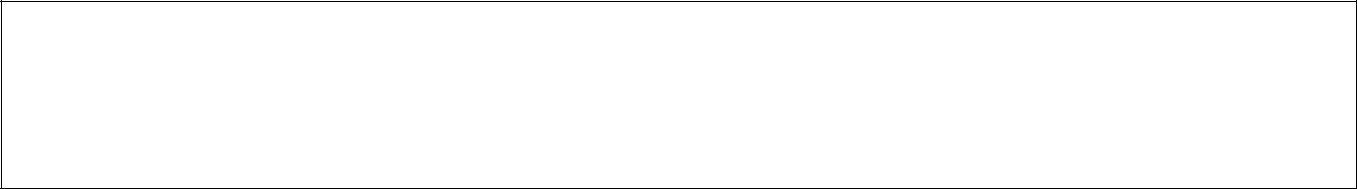
background-attachment:scroll;">

This parapgraph has scrolling background image.

</p>

**Shorthand Property**

You can use the *background* property to set all the background properties at once. For example:



<p style="background:url(/images/pattern1.gif) repeat fixed;">

This parapgraph has fixed repeated background image.

</p>

1. **CSS – Font**

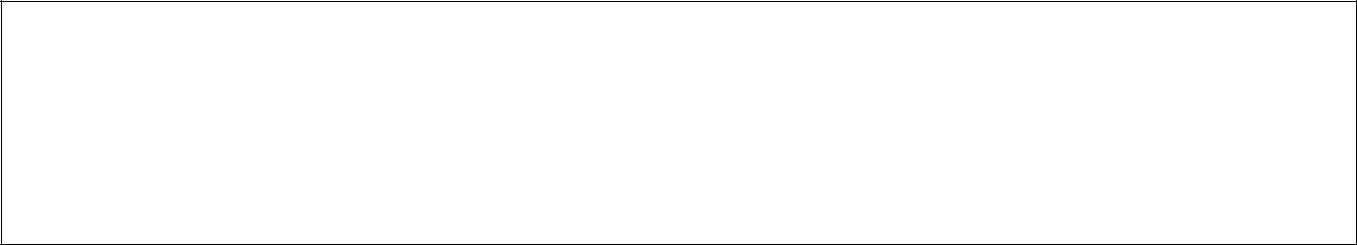
This section teaches you how to set fonts of a content, available in an HTML element. You can set the following font properties of an element:

* The **font-family** property is used to change the face of a font.
* The **font-style** property is used to make a font italic or oblique.
* The **font-variant** property is used to create a small-caps effect.
* The **font-weight** property is used to increase or decrease how bold or light a font appears.
* The **font-size** property is used to increase or decrease the size of a font.
* The **font** property is used as shorthand to specify a number of other font properties.

**Set the Font Family**



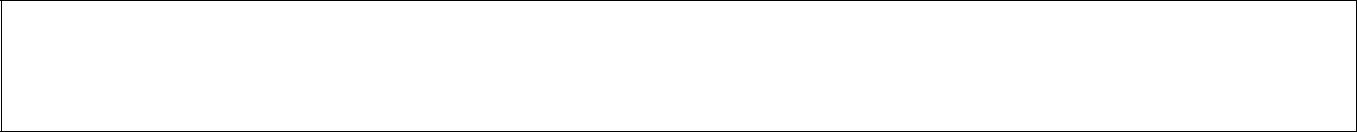
Following is the example, which demonstrates how to set the font family of an element. Possible value could be any font family name.



<p style="font-family:georgia,garamond,serif;">

This text is rendered in either georgia, garamond, or the default serif font depending on which font you have at your system. </p>

It will produce the following result:

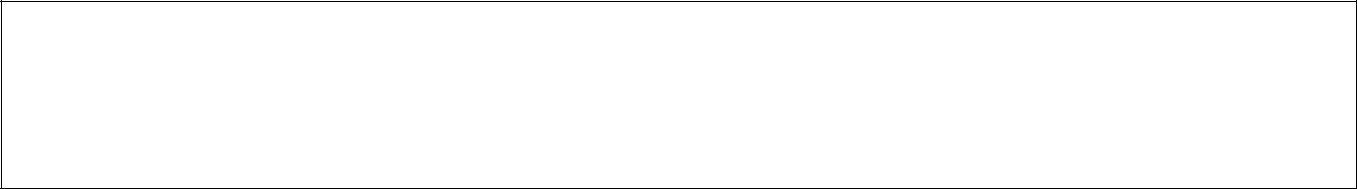


This text is rendered in either georgia, garamond, or the default serif font depending on which font you have at your system.

**Set the Font Style**



The following example demonstrates how to set the font style of an element. Possible values are *normal, italic and oblique*.



<p style="font-style:italic;">

This text will be rendered in italic style

</p>

It will produce the following result:

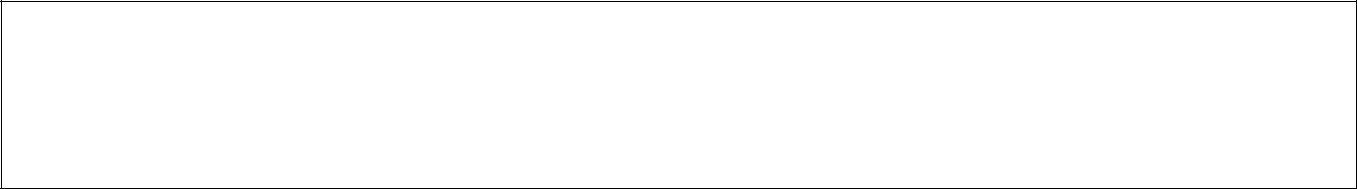


*This text will be rendered in italic style*

**Set the Font Variant**



The following example demonstrates how to set the font variant of an element. Possible values are *normal and small-caps*.



<p style="font-variant:small-caps;">

This text will be rendered as small caps

</p>

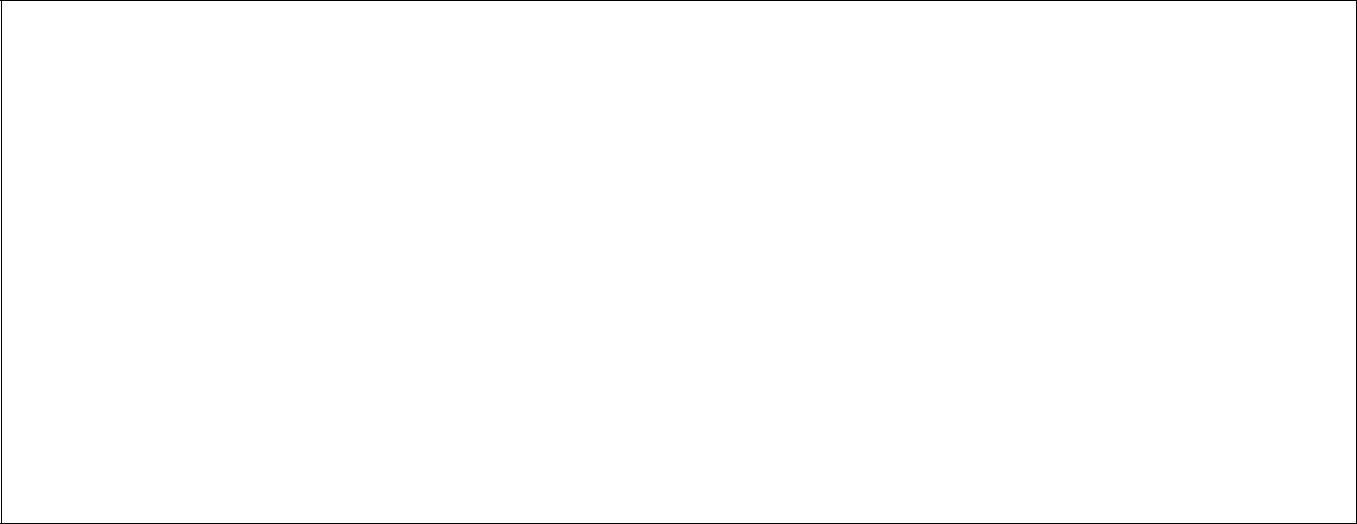
It will produce the following result:

THIS TEXT WILL BE RENEDERED AS SMALL CAPS

**Set the Font Weight**



The following example demonstrates how to set the font weight of an element. The font-weight property provides the functionality to specify how bold a font is. Possible values could be *normal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800, 900*.



<p style="font-weight:bold;"> This font is bold.

</p>

<p style="font-weight:bolder;"> This font is bolder.

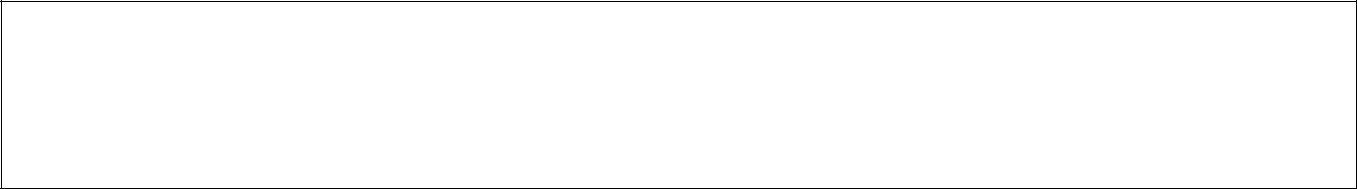
</p>

<p style="font-weight:900;"> This font is 900 weight. </p>

**Set the Font Size**



The following example demonstrates how to set the font size of an element. The font-size property is used to control the size of fonts. Possible values could be *xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger, size in pixels or in %.*



<p style="font-size:20px;">

This font size is 20 pixels

</p>

<p style="font-size:small;">

This font size is small

</p>

<p style="font-size:large;">

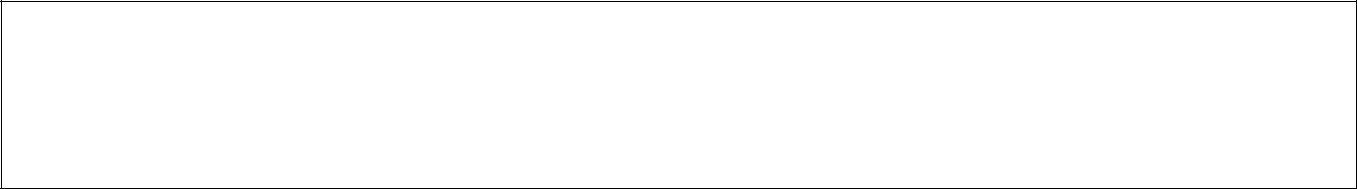
This font size is large

</p>

**Shorthand Property**



You can use the *font* property to set all the font properties at once. For example:



<p style="font:italic small-caps bold 15px georgia;">

Applying all the properties on the text at once.

</p>

It will produce the following result:

***APPLYING ALL THE PROPERTIES ON THE TEXT AT ONCE.***

1. **CSS - Text**

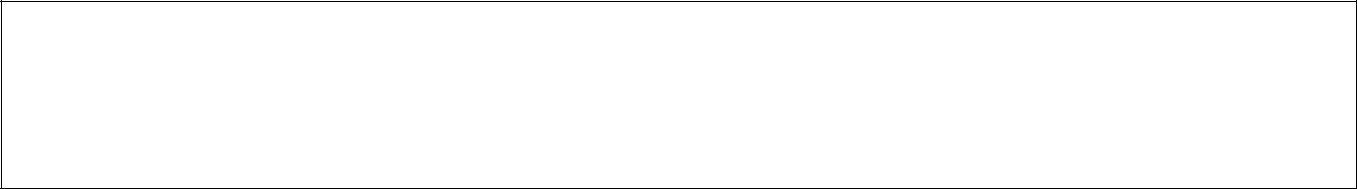
This section teaches you how to manipulate text using CSS properties. You can set the following text properties of an element:

* The **color** property is used to set the color of a text.
* The **direction** property is used to set the text direction.
* The **letter-spacing** property is used to add or subtract space between the letters that make up a word.
* The **word-spacing** property is used to add or subtract space between the words of a sentence.
* The **text-indent** property is used to indent the text of a paragraph.
* The **text-align** property is used to align the text of a document.
* The **text-decoration** property is used to underline, overline, and strikethrough text.
* The **text-transform** property is used to capitalize text or convert text to uppercase or lowercase letters.
* The **white-space** property is used to control the flow and formatting of text.
* The **text-shadow** property is used to set the text shadow around a text.

**Set the Text Color**



The following example demonstrates how to set the text color. Possible value could be any color name in any valid format.



<p style="color:red;">

This text will be written in red.

</p>

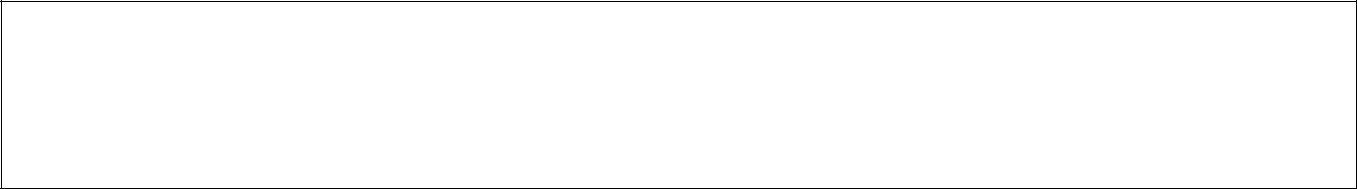
It will produce the following result:

This text will be written in red.

**Set the Text Direction**



The following example demonstrates how to set the direction of a text. Possible values are *ltr or rtl*.

<p style="direction:rtl;">

This text will be renedered from right to left

</p>

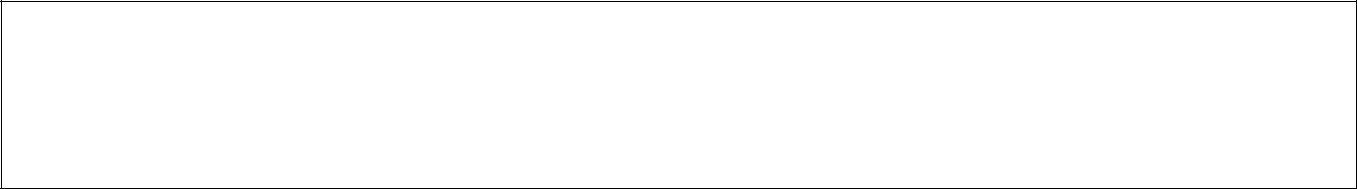
It will produce the following result:

This text will be renedered from right to left

**Set the Space between Characters**



The following example demonstrates how to set the space between characters. Possible values are *normal or a number specifying space.*



<p style="letter-spacing:5px;">

This text is having space between letters.

</p>

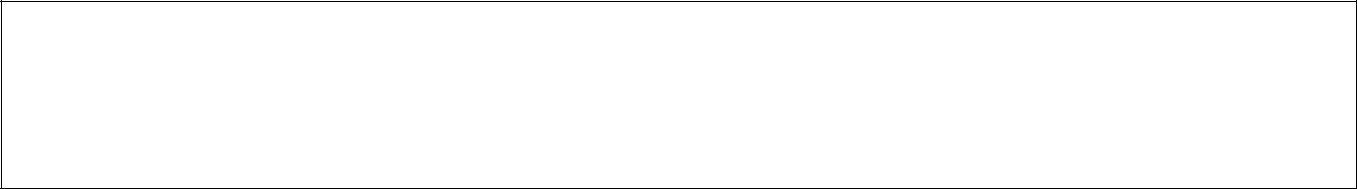
It will produce the following result:

T h i s t e x t i s h a v i n g s p a c e b e t w e e n l e t t e r s .

**Set the Space between Words**



The following example demonstrates how to set the space between words. Possible values are *normal or a number specifying space.*



<p style="word-spacing:5px;">

This text is having space between words.

</p>

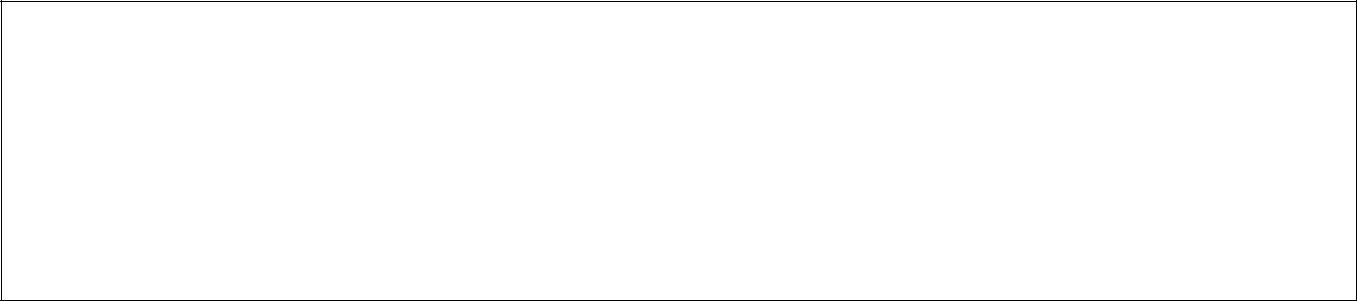
It will produce the following result:

This text is having space between words.

**Set the Text Indent**



The following example demonstrates how to indent the first line of a paragraph. Possible values are *% or a number specifying indent space.*



<p style="text-indent:1cm;">

This text will have first line indented by 1cm and this line will remain at its actual position this is done by CSS text-indent property. </p>

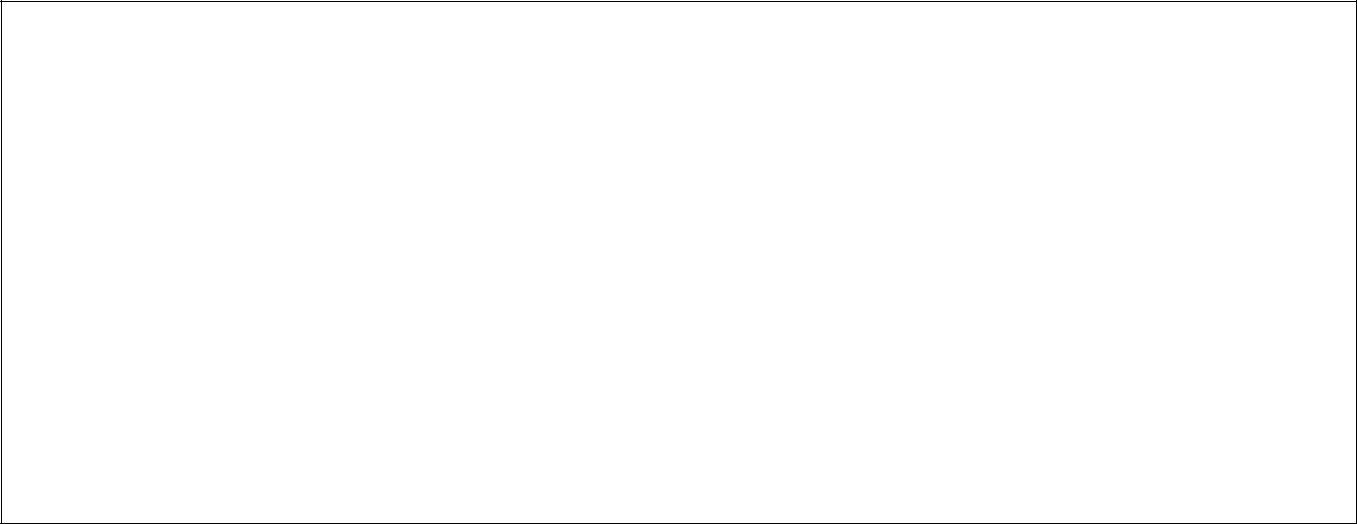
It will produce the following result:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| This | text | will | have | first | line |  | indented | by | 1cm |
| and | this | line | will | remain | at | its | actual |  | position |
| this is done by CSS text-indent property. | | | |  |  |  |  |  |  |

**Set the Text Alignment**



The following example demonstrates how to align a text. Possible values are *left, right,* *center, justify.*



<p style="text-align:right;"> This will be right aligned. </p>

<p style="text-align:center;"> This will be center aligned. </p>

<p style="text-align:left;"> This will be left aligned. </p>

It will produce the following result:

This will be right aligned.

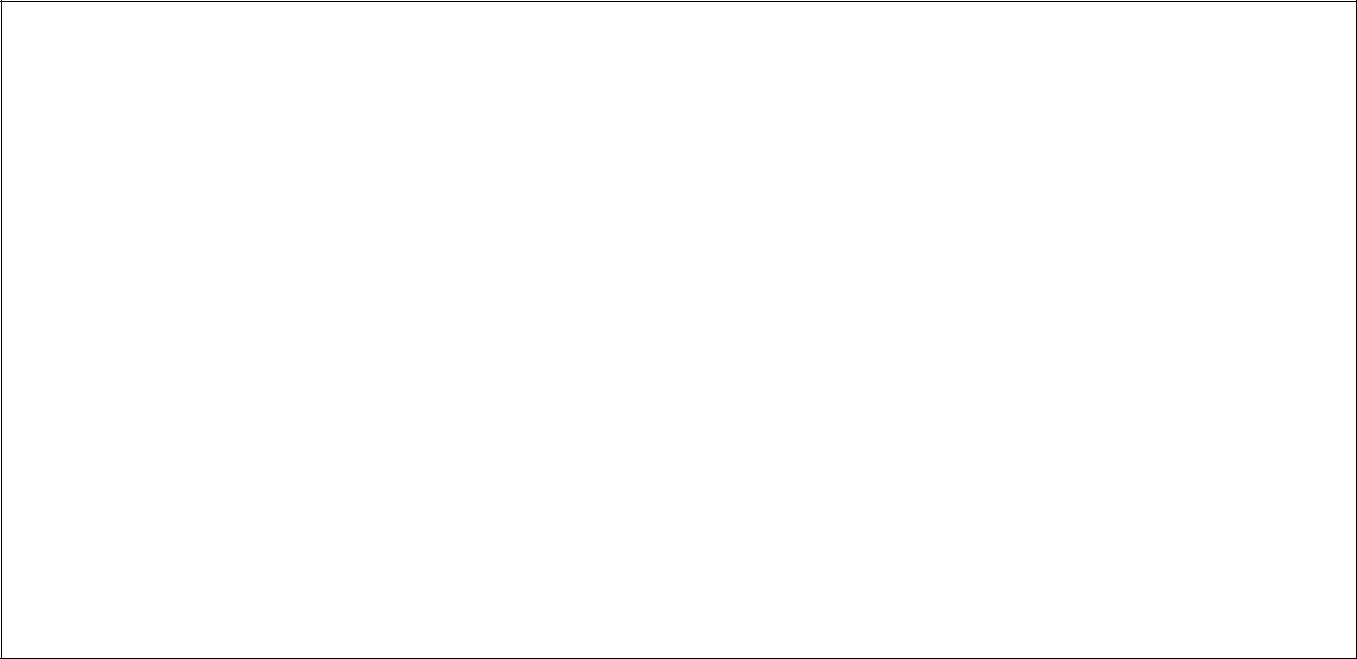
This will be center aligned.

This will be left aligned.

**Decorating the Text**



The following example demonstrates how to decorate a text. Possible values are *none,* *underline, overline, line-through, blink.*



<p style="text-decoration:underline;">This will be underlined </p>

<p style="text-decoration:line-through;"> This will be striked through. </p>

<p style="text-decoration:overline;"> This will have an over line.</p>

It will produce the following result:

This will be underlined

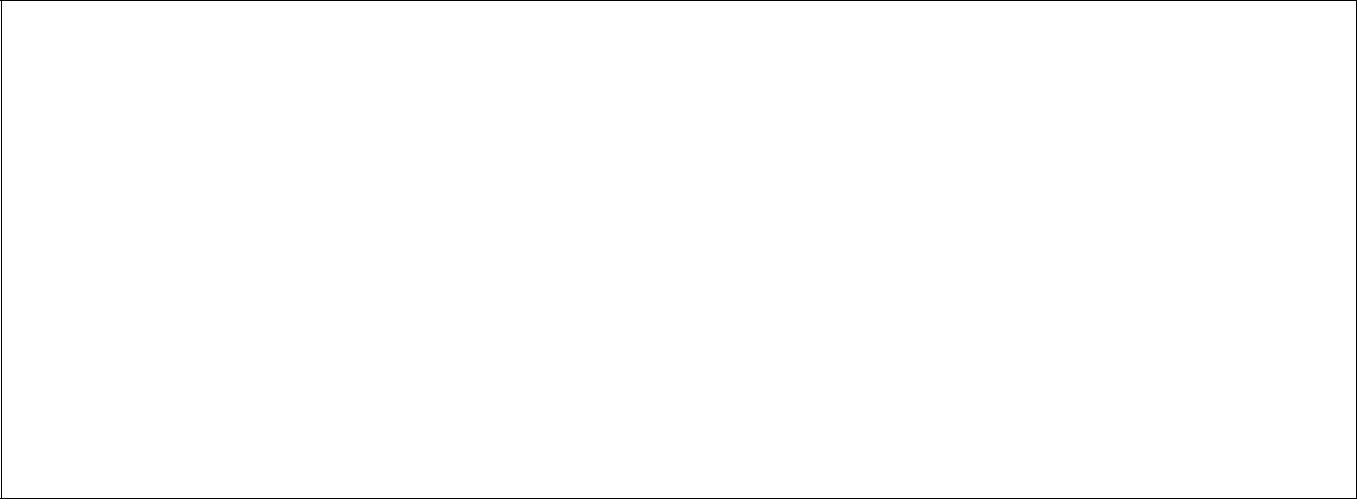
~~This will be striked through.~~

This will have an over line.

**Set the Text Cases**



The following example demonstrates how to set the cases for a text. Possible values are *none, capitalize, uppercase, lowercase.*



<p style="text-transform:capitalize;"> This will be capitalized

</p>

<p style="text-transform:uppercase;"> This will be in uppercase

</p>

<p style="text-transform:lowercase;"> This will be in lowercase

</p>

It will produce the following result:

This Will Be Capitalized

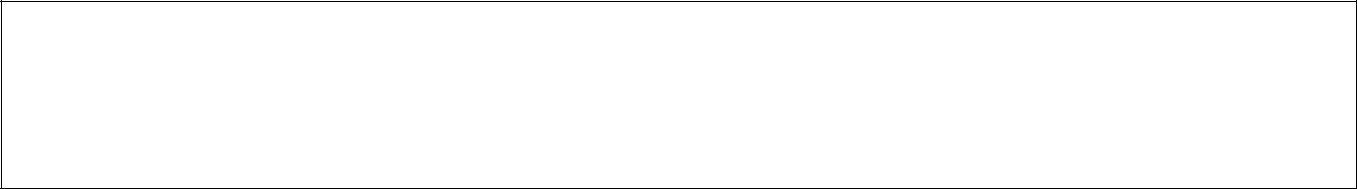
THIS WILL BE IN UPPERCASE

this will be in lowercase

**Set the White Space between Text**



The following example demonstrates how white space inside an element is handled. Possible values are *normal, pre, nowrap*.



<p style="white-space:pre;">This text has a line break

and the white-space pre setting tells the browser to honor it just like the HTML pre tag.</p>

It will produce the following result:

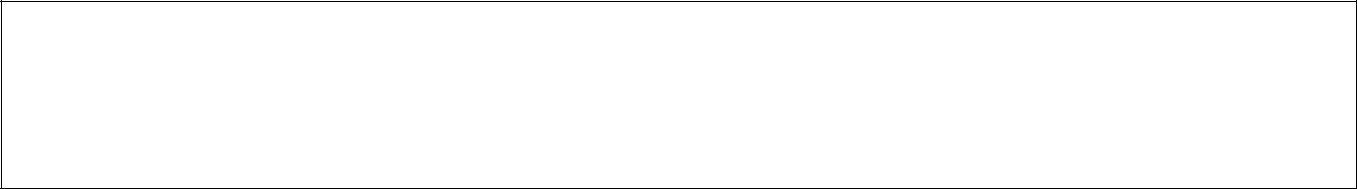
This text has a line break

and the white-space pre setting tells the browser to honor it just like the HTML pre tag.

**Set the Text Shadow**



The following example demonstrates how to set the shadow around a text. This may not be supported by all the browsers.



<p style="text-shadow:4px 4px 8px blue;">

If your browser supports the CSS text-shadow property, this text will have a blue shadow.</p>

1. **CSS - Images**

Images play an important role in any webpage. Though it is not recommended to include a lot of images, but it is still important to use good images wherever required.

CSS plays a good role to control image display. You can set the following image properties using CSS.

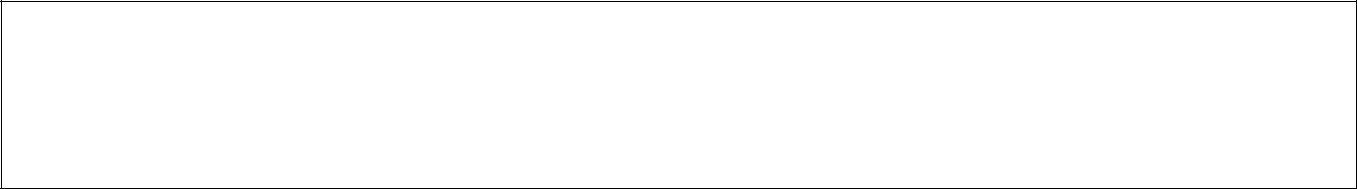
* The **border** property is used to set the width of an image border.
* The **height** property is used to set the height of an image.
* The **width** property is used to set the width of an image.
* The **-moz-opacity** property is used to set the opacity of an image.

**The Image Border Property**



The *border* property of an image is used to set the width of an image border. This property can have a value in length or in %. A width of zero pixels means no border.

Here is an example:



<img style="border:0px;" src="/images/css.gif" />

<br />

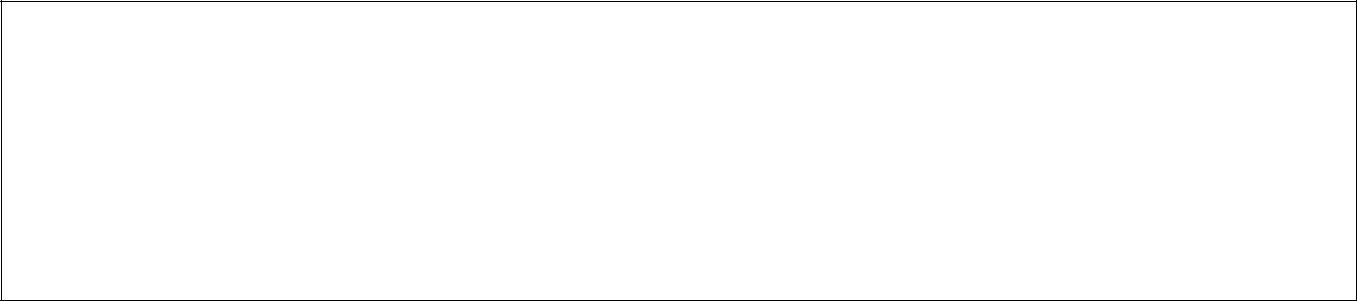
<img style="border:3px dashed red;" src="/images/css.gif" />

**The Image Height Property**



The *height* property of an image is used to set the height of an image. This property can have a value in length or in %. While giving value in %, it applies it in respect of the box in which an image is available.

Here is an example:



<img style="border:1px solid red; height:100px;" src="/images/css.gif" />

<br />

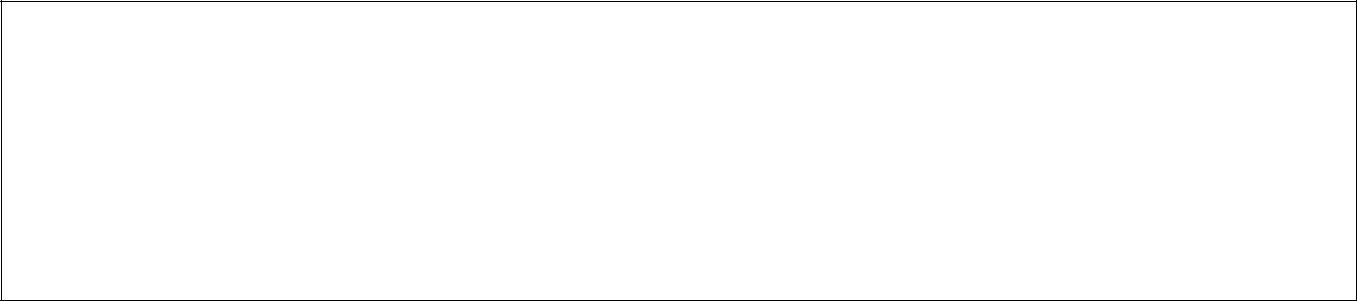
<img style="border:1px solid red; height:50%;" src="/images/css.gif" />

**The Image Width Property**



The *width* property of an image is used to set the width of an image. This property can have a value in length or in %. While giving value in %, it applies it in respect of the box in which an image is available.

Here is an example:



<img style="border:1px solid red; width:100px;" src="/images/css.gif" />

<br />

<img style="border:1px solid red; width:100%;" src="/images/css.gif" />

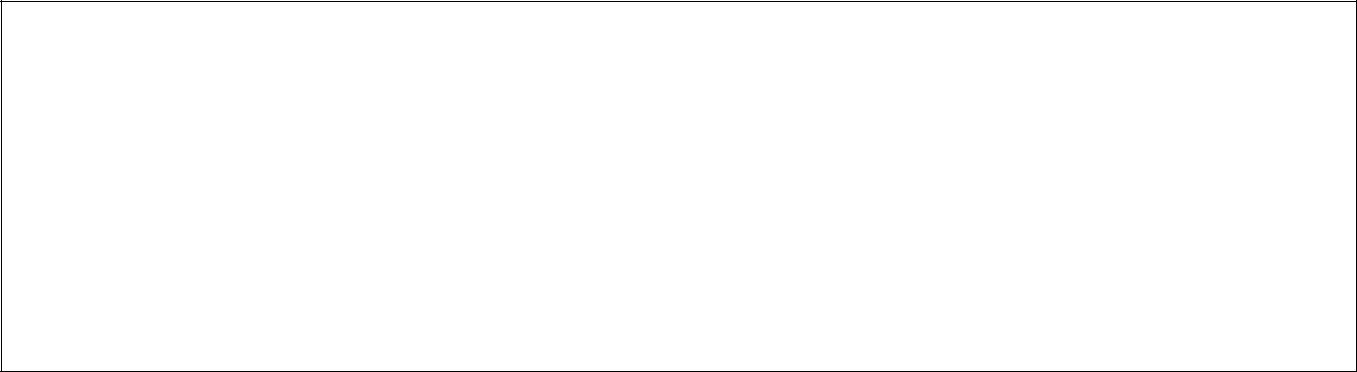
1. **CSS - Link**

This section teaches you how to set different properties of a hyper link using CSS. You can set the following properties of a hyperlink:

* We will revisit the same properties when we will discuss Pseudo-Classes of CSS.
* The **:link** signifies unvisited hyperlinks.
* The **:visited** signifies visited hyperlinks.
* The **:hover** signifies an element that currently has the user's mouse pointer hovering over it.
* The **:active** signifies an element on which the user is currently clicking.

Usually, all these properties are kept in the header part of the HTML document.

Remember a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective. Also, a:active MUST come after a:hover in the CSS definition as follows:



<style type="text/css">

a:link {color: #000000}

a:visited {color: #006600}

a:hover {color: #FFCC00}

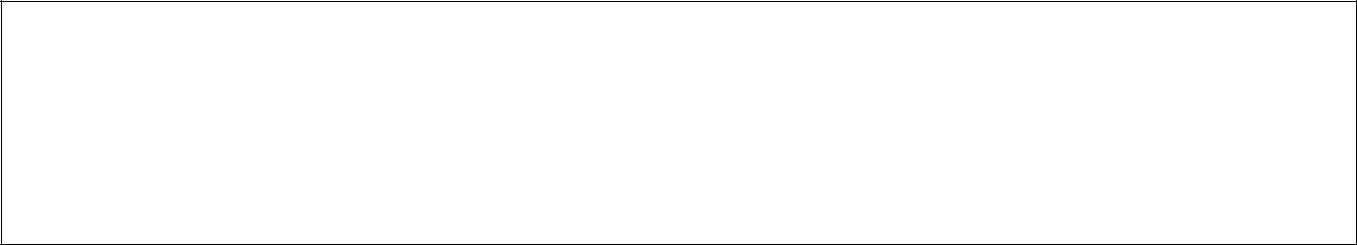
a:active {color: #FF00CC}

</style>

**Set the Color of Links**



The following example demonstrates how to set the link color. Possible values could be any color name in any valid format.



<style type="text/css">

a:link {color:#000000}

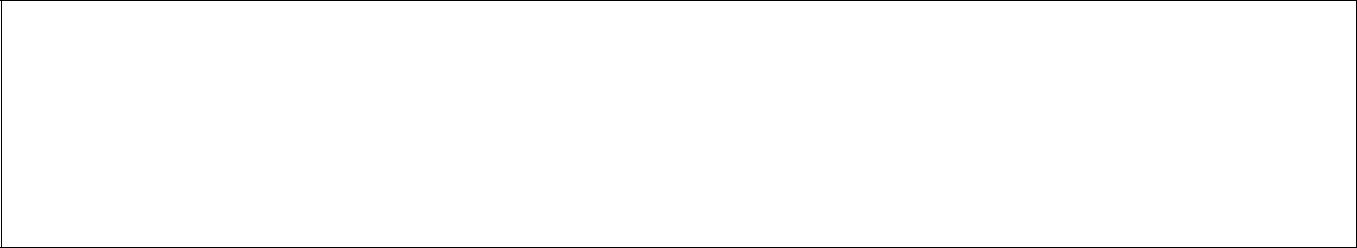
</style>

<a href="/html/index.htm">Black Link</a>

**Set the Color of Visited Links**



The following example demonstrates how to set the color of the visited links. Possible values could be any color name in any valid format.

<style type="text/css">

a:visited {color: #006600}

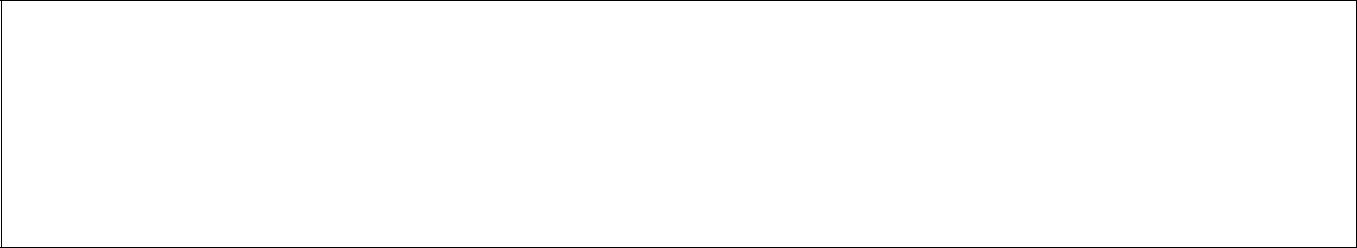
</style>

<a href="/html/index.htm">Click this link</a>

**Change the Color of Links when Mouse is Over**



The following example demonstrates how to change the color of links when we bring a mouse pointer over that link. Possible values could be any color name in any valid format.



<style type="text/css">

a:hover {color: #FFCC00}

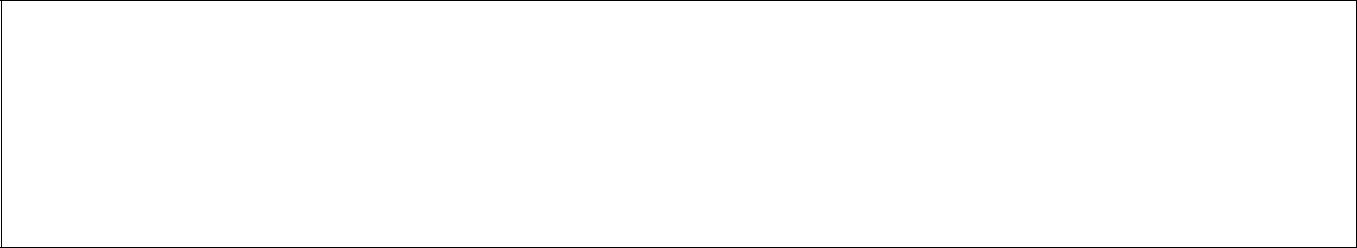
</style>

<a href="/html/index.htm">Bring Mouse Here</a>

**Change the Color of Active Links**



The following example demonstrates how to change the color of active links. Possible values could be any color name in any valid format.



<style type="text/css">

a:active {color: #FF00CC}

</style>

<a href="/html/index.htm">Click This Link</a>

1. **CSS – Tables**

This section teaches you how to set different properties of an HTML table using CSS. You can set the following properties of a table:

* The **table-layout** allows browsers to speed up the layout of a table by using the first width properties it comes across for the rest of a column rather than having to load the whole table before rendering it.

**The table-layout Property**



The table-layout property is supposed to help you control how a browser should render or lay out a table.

This property can have one of the three values: *fixed, auto*, or *inherit*.

The following example shows the difference between these properties.

**NOTE:** This property is not supported by many browsers, so do not rely on this property.



<table class="auto" border="1" width="100%"> <tr> <td width="20%">1000000000000000000000000000</td> <td width="40%">10000000</td>

<td width="40%">100</td> </tr> </table>

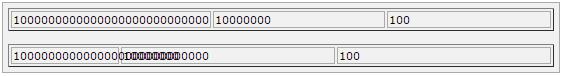
<table class="fixed" border="1" width="100%"> <tr> <td width="20%">1000000000000000000000000000</td> <td width="40%">10000000</td>

<td width="40%">100</td>

</tr>

</table>

It will produce the following result:



1. **CSS - Borders****2**

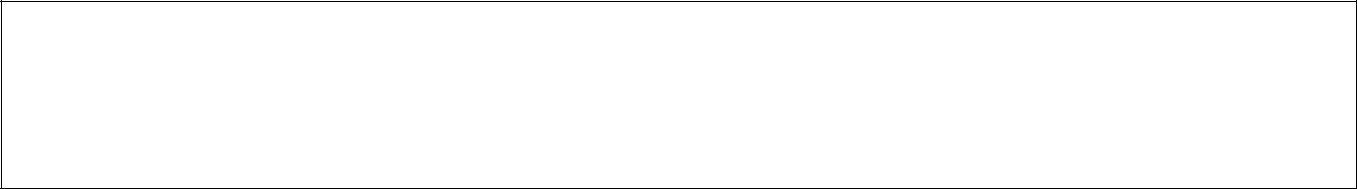
The *border* properties allow you to specify how the border of the box representing an element should look. There are three properties of a border you can change:

**Border Properties Using Shorthand**



The border property allows you to specify color, style, and width of lines in one property:

The following example shows how to use all the three properties into a single property. This is the most frequently used property to set border around any element.



<p style="border:4px solid red;">

This example is showing shorthand property for border.

</p>

It will produce the following result:

**13. CSS ─ Margins**

The *margin* property defines the space around an HTML element. It is possible to use negative values to overlap content.

The values of the margin property are not inherited by the child elements. Remember that the adjacent vertical margins (top and bottom margins) will collapse into each other so that the distance between the blocks is not the sum of the margins, but only the greater of the two margins or the same size as one margin if both are equal.

We have the following properties to set an element margin.

* The **margin** specifies a shorthand property for setting the margin properties in one declaration.
* The **margin-bottom** specifies the bottom margin of an element.
* The **margin-top** specifies the top margin of an element.
* The **margin-left** specifies the left margin of an element.
* The **margin-right** specifies the right margin of an element.

**The Margin Property**



The margin property allows you to set all of the properties for the four margins in one declaration. Here is the syntax to set margin around a paragraph:

Here is an example:



<p style="margin: 15px; border:1px solid black;">

all four margins will be 15px

</p>

<p style="margin:10px 2%; border:1px solid black;">

top and bottom margin will be 10px, left and right margin will be 2% of the total width of the document.

</p>

<p style="margin: 10px 2% -10px; border:1px solid black;">

top margin will be 10px, left and right margin will be 2% of the total width of the document, bottom margin will be -10px

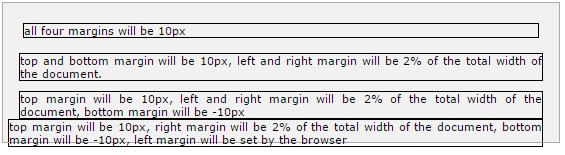
</p>

<p style="margin: 10px 2% -10px auto; border:1px solid black;">

top margin will be 10px, right margin will be 2% of the total width of the document, bottom margin will be -10px, left margin will be set by the browser

</p>

It will produce the following result:



**14. CSS ─ Paddings**

The *padding* property allows you to specify how much space should appear between the content of an element and its border:

The value of this attribute should be either a length, a percentage, or the word *inherit*. If the value is *inherit*, it will have the same padding as its parent element. If a percentage is used, the percentage is of the containing box.

The following CSS properties can be used to control lists. You can also set different values for the padding on each side of the box using the following properties:

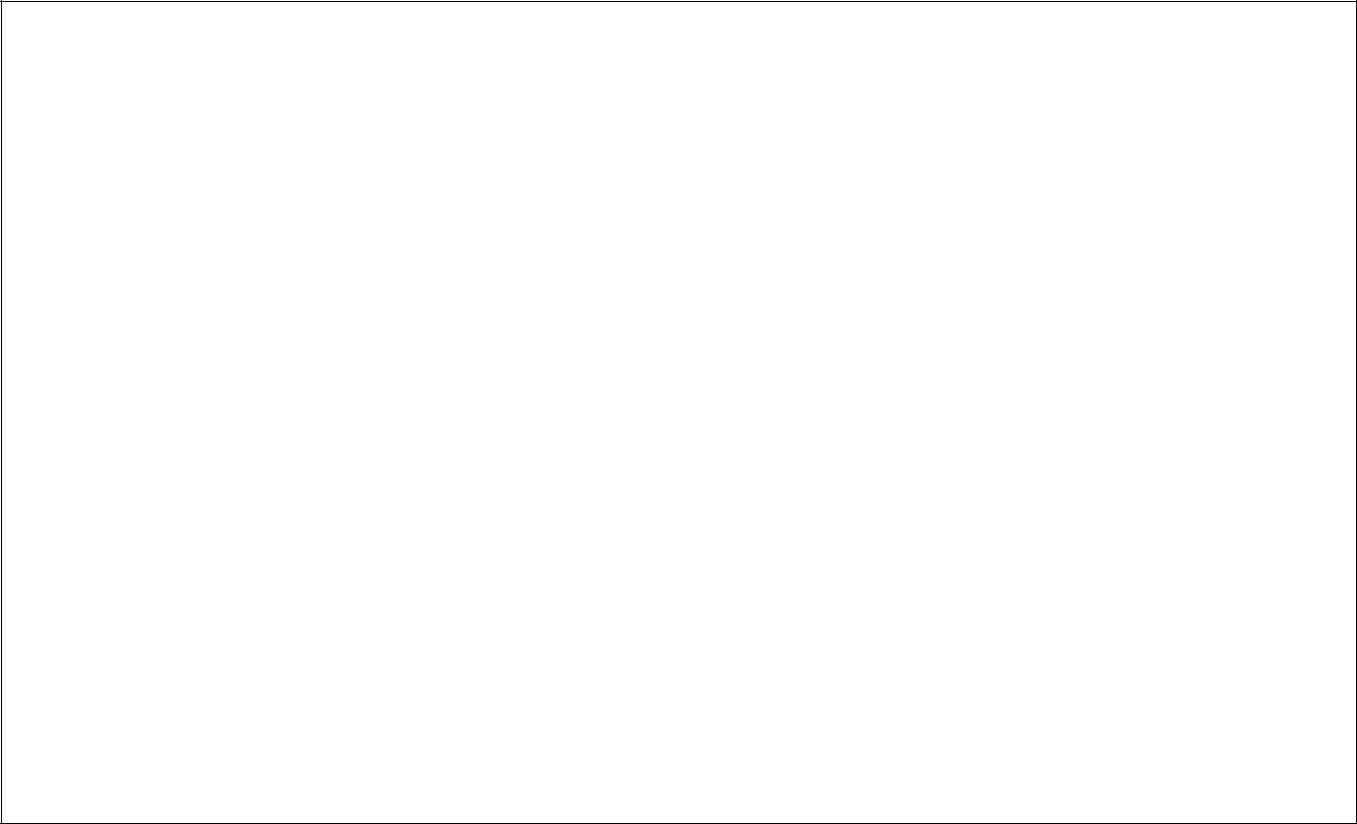
* The **padding-bottom** specifies the bottom padding of an element.
* The **padding-top** specifies the top padding of an element.
* The **padding-left** specifies the left padding of an element.
* The **padding-right** specifies the right padding of an element.
* The **padding** serves as shorthand for the preceding properties.

**The Padding Property**



The *padding* property sets the left, right, top and bottom padding (space) of an element. This can take a value in terms of length of %.

Here is an example:



<p style="padding: 15px; border:1px solid black;">

all four padding will be 15px

</p>

<p style="padding:10px 2%; border:1px solid black;">

top and bottom padding will be 10px, left and right padding will be 2% of the total width of the document.

</p>

<p style="padding: 10px 2% 10px; border:1px solid black;">

top padding will be 10px, left and right padding will be 2% of the total width of the document, bottom padding will be 10px

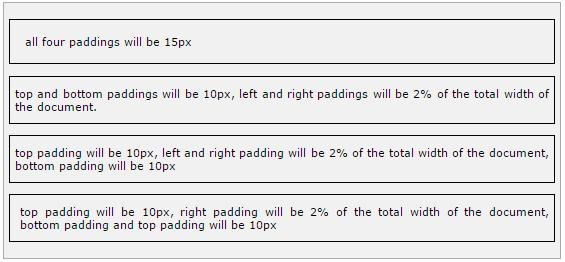
</p>

<p style="padding: 10px 2% 10px 10px; border:1px solid black;">

top padding will be 10px, right padding will be 2% of the total width of the document, bottom padding and top padding will be 10px

</p>

It will produce the following result:



**15. CSS ─ Cursors**

The *cursor* property of CSS allows you to specify the type of cursor that should be displayed to the user.

One good usage of this property is in using images for submit buttons on forms. By default, when a cursor hovers over a link, the cursor changes from a pointer to a hand. However, it does not change form for a submit button on a form. Therefore, whenever someone hovers over an image that is a submit button, it provides a visual clue that the image is clickable.

The following table shows the possible values for the cursor property:

|  |  |
| --- | --- |
| **Value** | **Description** |
|  |  |
| auto | Shape of the cursor depends on the context area it is over. For example, |
|  | an ‘I’ over text, a ‘hand’ over a link, and so on. |
|  |  |
| crosshair | A crosshair or plus sign. |
|  |  |
| default | An arrow. |
|  |  |
| pointer | A pointing hand (in IE 4 this value is hand). |
|  |  |
| move | The ‘I’ bar. |
|  |  |
| e-resize | The cursor indicates that an edge of a box is to be moved right (east). |
|  |  |
| ne-resize | The cursor indicates that an edge of a box is to be moved up and right |
|  | (north/east). |
|  |  |
| nw- | The cursor indicates that an edge of a box is to be moved up and left |
| resize | (north/west). |
|  |  |
| n-resize | The cursor indicates that an edge of a box is to be moved up (north). |
|  |  |
| se-resize | The cursor indicates that an edge of a box is to be moved down and right |
|  | (south/east). |
|  |  |
| sw- | The cursor indicates that an edge of a box is to be moved down and left |
| resize | (south/west). |
|  |  |
| s-resize | The cursor indicates that an edge of a box is to be moved down (south). |
|  |  |
| w-resize | The cursor indicates that an edge of a box is to be moved left (west). |
|  |  |
| text | The I bar. |
|  |  |
| wait | An hour glass. |
|  |  |
| help | A question mark or balloon, ideal for use over help buttons. |
|  |  |
| <url> | The source of a cursor image file. |
|  |  |

**NOTE:** You should try to use only these values to add helpful information for users, andin places, they would expect to see that cursor. For example, using the crosshair when someone hovers over a link can confuse the visitors.

Here is an example:



<p>Move the mouse over the words to see the cursor change:</p> <div style="cursor:auto">Auto</div>

<div style="cursor:crosshair">Crosshair</div> <div style="cursor:default">Default</div> <div style="cursor:pointer">Pointer</div> <div style="cursor:move">Move</div>

<div style="cursor:e-resize">e-resize</div> <div style="cursor:ne-resize">ne-resize</div> <div style="cursor:nw-resize">nw-resize</div> <div style="cursor:n-resize">n-resize</div> <div style="cursor:se-resize">se-resize</div> <div style="cursor:sw-resize">sw-resize</div> <div style="cursor:s-resize">s-resize</div> <div style="cursor:w-resize">w-resize</div> <div style="cursor:text">text</div>



<div style="cursor:wait">wait</div> <div style="cursor:help">help</div>

**16) CSS ─ Dimension**

You have seen the border that surrounds every box i.e. element, the padding that can appear inside each box, and the margin that can go around them. In this section, we will learn how to change the dimensions of boxes.

We have the following properties that allow you to control the dimensions of a box.

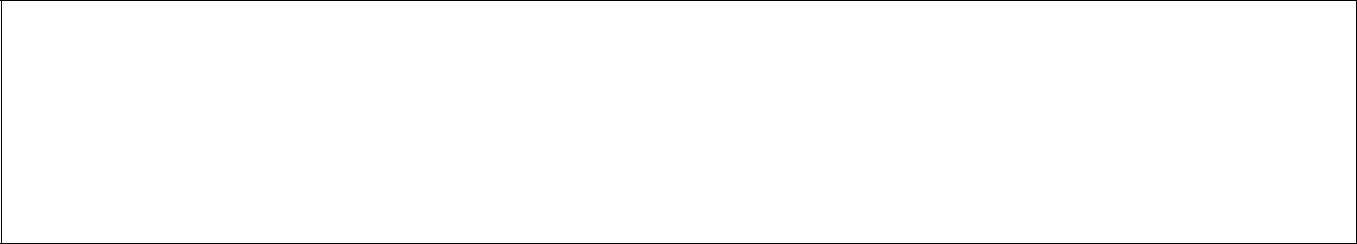
* The **height** property is used to set the height of a box.
* The **width** property is used to set the width of a box.
* The **line-height** property is used to set the height of a line of text.
* The **max-height** property is used to set a maximum height that a box can be .
* The **min-height** property is used to set the minimum height that a box can be .
* The **max-width** property is used to set the maximum width that a box can be .
* The **min-width** property is used to set the minimum width that a box can be .

**The Height and Width Properties**



The *height* and *width* properties allow you to set the height and width for boxes. They can take values of a length, a percentage, or the keyword auto.

Here is an example:



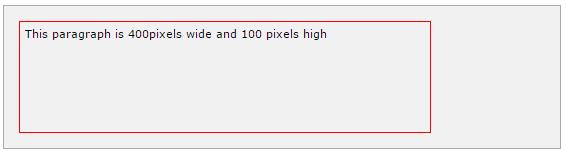
<p style="width:400px; height:100px;border:1px solid red;

padding:5px; margin:10px;">

This paragraph is 400pixels wide and 100 pixels high

</p>

It will produce the following result:

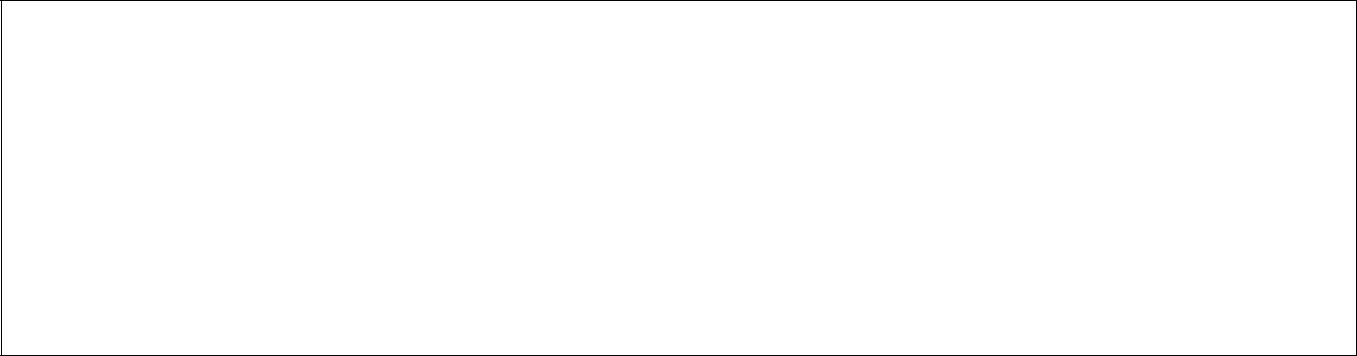


**The line-height Property**



The *line-height* property allows you to increase the space between lines of text. The value of the line-height property can be a number, a length, or a percentage.

Here is an example:

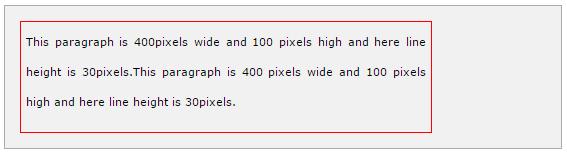


<p style="width:400px; height:100px;border:1px solid red; padding:5px; margin:10px;line-height:30px;">

This paragraph is 400pixels wide and 100 pixels high

and here line height is 30pixels.This paragraph is 400 pixels wide and 100 pixels high and here line height is 30pixels. </p>

It will produce the following result:



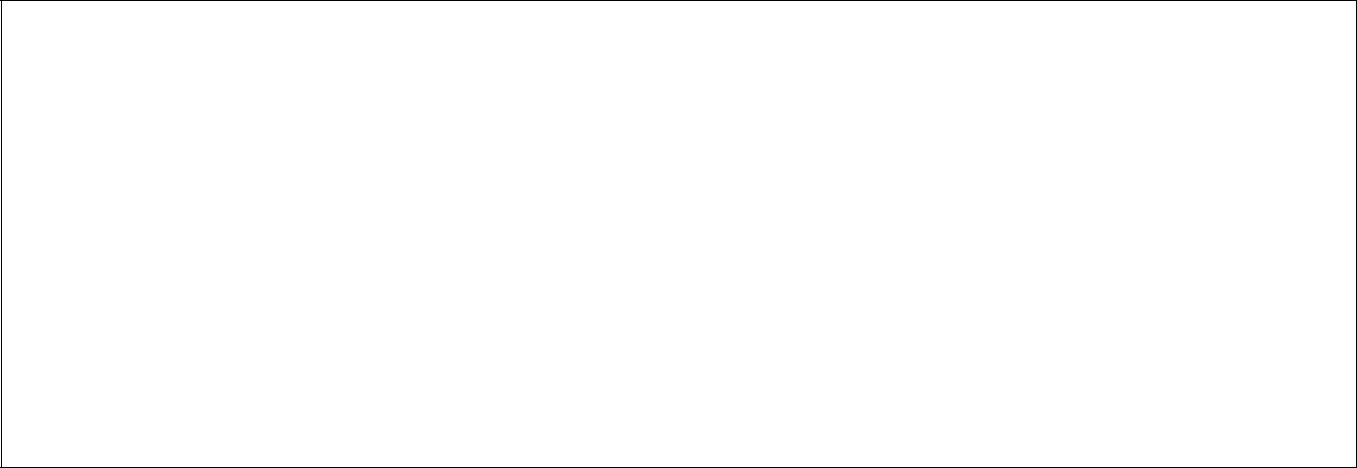
**The max-height Property**



The *max-height* property allows you to specify the maximum height of a box. The value of the max-height property can be a number, a length, or a percentage.

**NOTE:** This property does not work in either Netscape 7 or IE 6.

Here is an example:



<p style="width:400px; max-height:10px;border:1px solid red; padding:5px; margin:10px;">

This paragraph is 400px wide and max height is 10px This paragraph is 400px wide and max height is 10px This paragraph is 400px wide and max height is 10px This paragraph is 400px wide and max height is 10px </p>

<img alt="logo" src="/images/css.gif" width="95" height="84" />

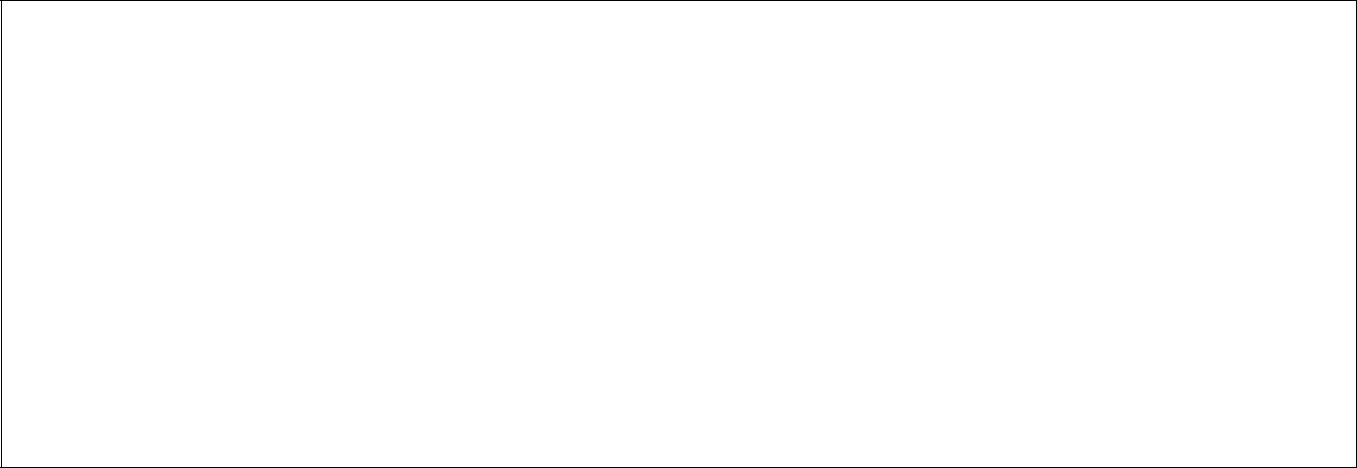
**The min-height Property**



The *min-height* property allows you to specify the minimum height of a box. The value of the min-height property can be a number, a length, or a percentage.

**NOTE:** This property does not work in either Netscape 7 or IE 6.

Here is an example:



<p style="width:400px; min-height:200px;border:1px solid red; padding:5px; margin:10px;">

This paragraph is 400px wide and min height is 200px This paragraph is 400px wide and min height is 200px This paragraph is 400px wide and min height is 200px This paragraph is 400px wide and min height is 200px </p>

<img alt="logo" src="/images/css.gif" width="95" height="84" />

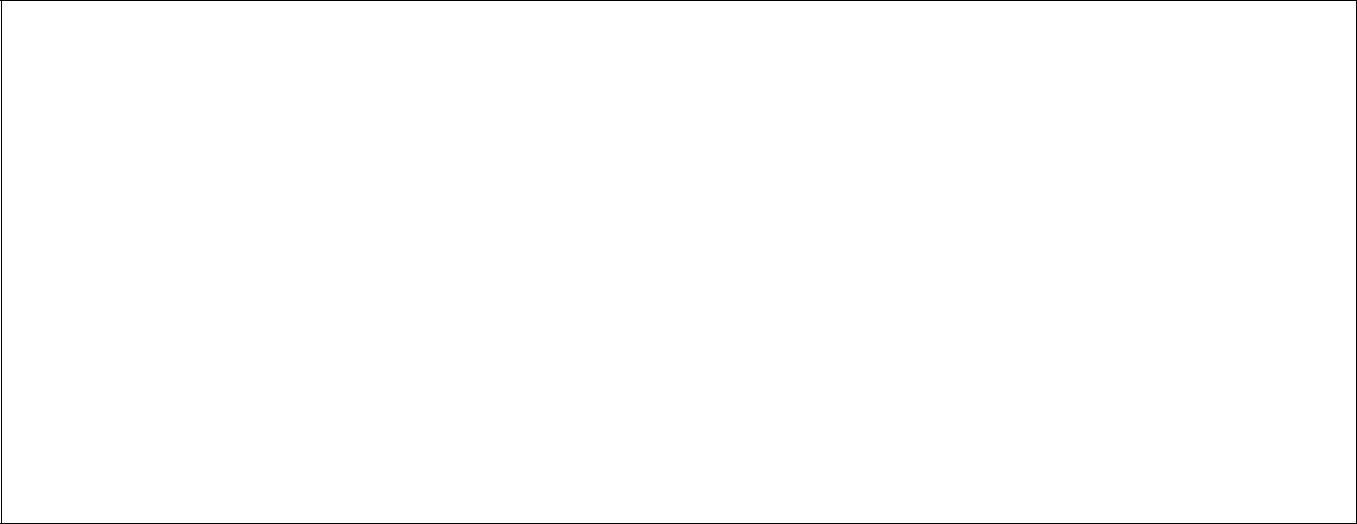
**The max-width Property**



The *max-width* property allows you to specify the maximum width of a box. The value of the max-width property can be a number, a length, or a percentage.

**NOTE:** This property does not work in either Netscape 7 or IE 6.

Here is an example:



<p style="max-width:100px; height:200px;border:1px solid red; padding:5px; margin:10px;">

This paragraph is 200px high and max width is 100px This paragraph is 200px high and max width is 100px This paragraph is 200px high and max width is 100px This paragraph is 200px high and max width is 100px This paragraph is 200px high and max width is 100px </p>

<img alt="logo" src="/images/css.gif" width="95" height="84" />

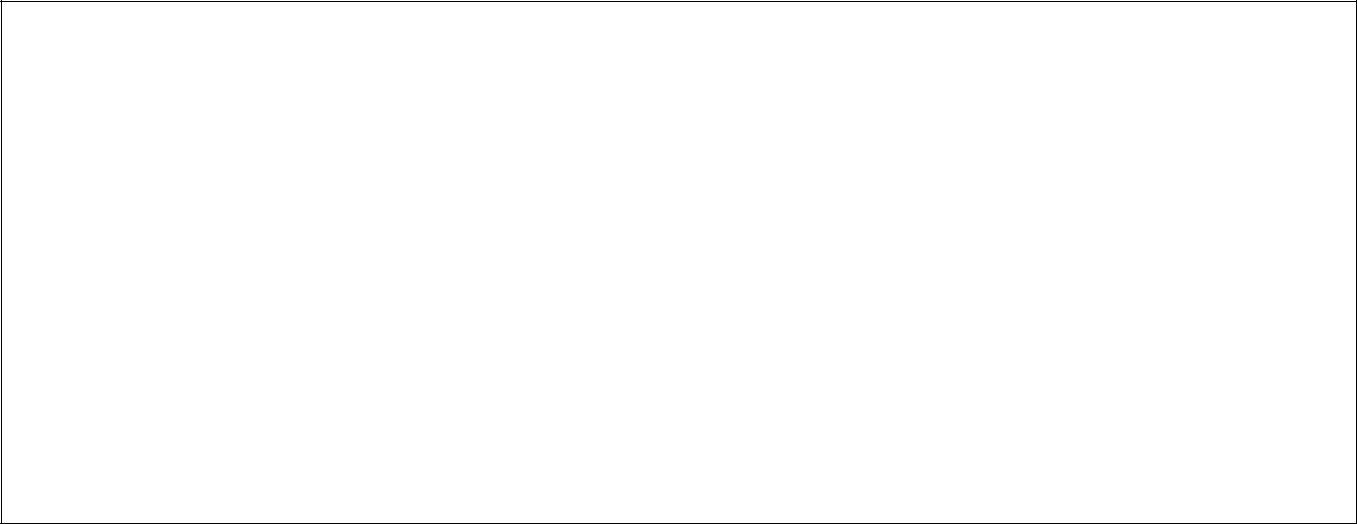
**The min-width Property**



The *min-width* property allows you to specify the minimum width of a box. The value of the min-width property can be a number, a length, or a percentage.

**NOTE:** This property does not work in either Netscape 7 or IE 6.

Here is an example:



<p style="min-width:400px; height:100px;border:1px solid red; padding:5px; margin:10px;">

This paragraph is 100px high and min width is 400px This paragraph is 100px high and min width is 400px This paragraph is 100px high and min width is 400px This paragraph is 100px high and min width is 400px This paragraph is 100px high and min width is 400px </p>

<img alt="logo" src="/images/css.gif" width="95" height="84" />

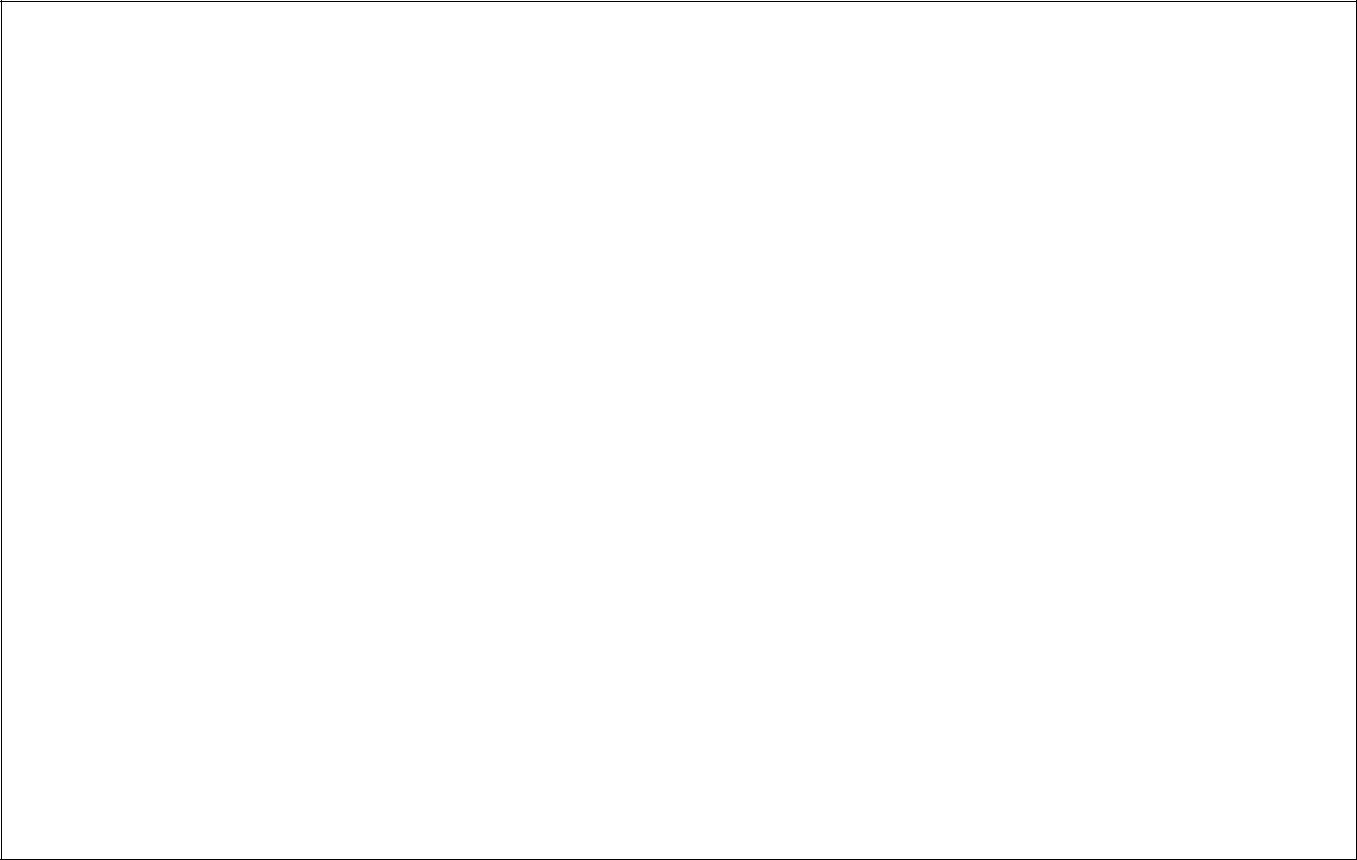
**17. CSS ─ Scrollbars**

There may be a case when an element's content might be larger than the amount of space allocated to it. For example, the given width and height properties do not allow enough room to accommodate the content of the element.

CSS provides a property called *overflow*, which tells the browser what to do if the box's contents is larger than the box itself. This property can take one of the following values:

|  |  |
| --- | --- |
| **Value** | **Description** |
|  |  |
| visible | Allows the content to overflow the borders of its containing element. |
|  |  |
| hidden | The content of the nested element is simply cut off at the border of the |
|  | containing element and no scrollbars is visible. |
|  |  |
| scroll | The size of the containing element does not change, but the scrollbars are |
|  | added to allow the user to scroll to see the content. |
|  |  |
| auto | The purpose is the same as scroll, but the scrollbar will be shown only if the |
|  | content does overflow. |
|  |  |

Here is an example:



<style type="text/css">

.scroll{

display:block; border: 1px solid red; padding:5px; margin-top:5px; width:300px; height:50px; overflow:scroll;

}

.auto{

display:block; border: 1px solid red; padding:5px;

margin-top:5px;

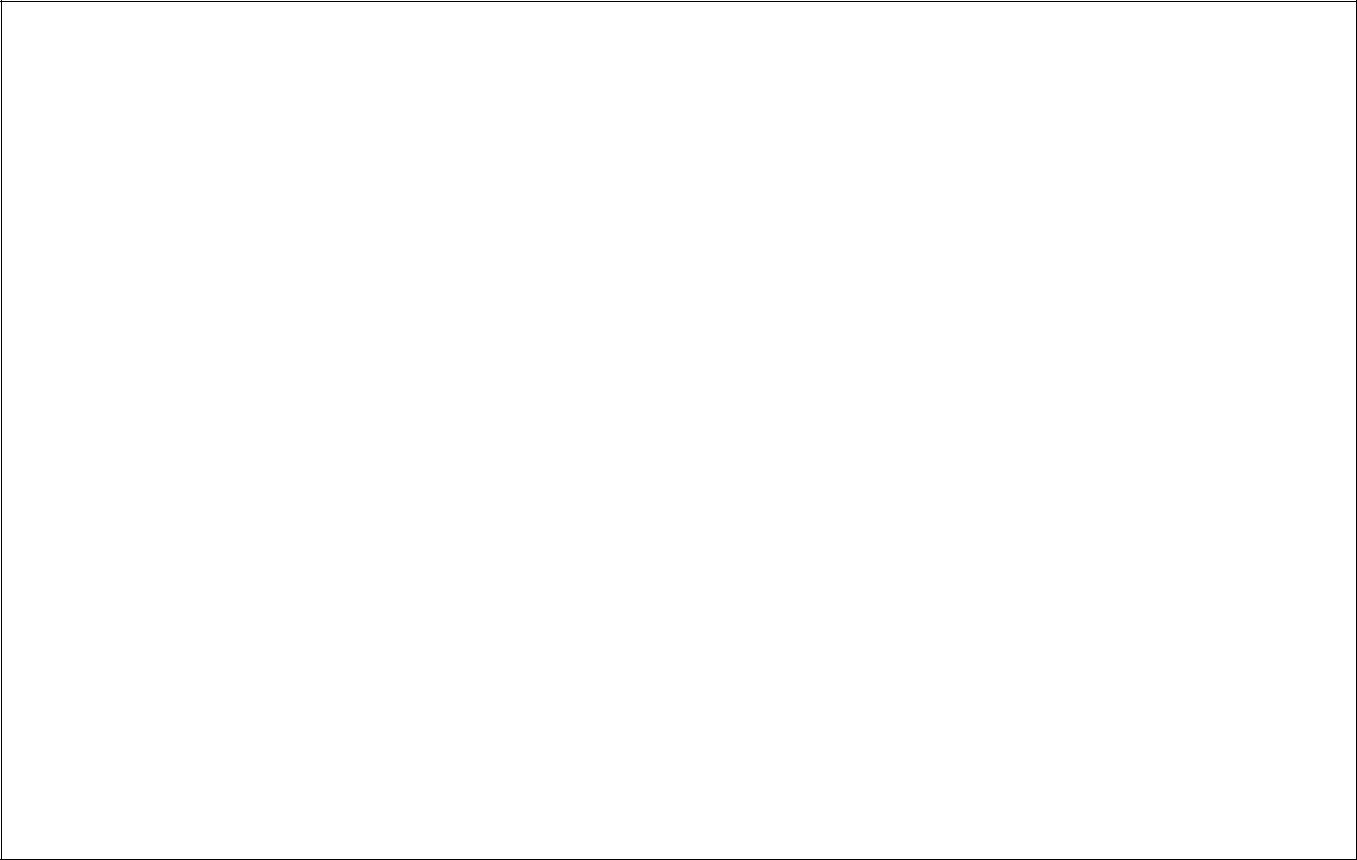
width:300px;

height:50px;

overflow:auto;

}

</style>

<p>Example of scroll value:</p> <div class="scroll">

I am going to keep lot of content here just to show you how scrollbars works if there is an overflow in an element box. This provides your horizontal as well

as vertical scrollbars. </div>

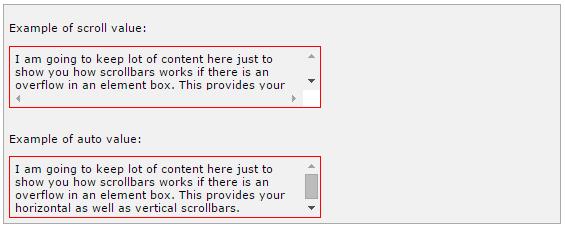
<br />

<p>Example of auto value:</p> <div class="auto">

I am going to keep lot of content here just to show you how scrollbars works if there is an overflow in an element box. This provides your horizontal as well as vertical scrollbars.

</div>

It will produce the following result:



A property called *visibility* allows you to hide an element from view. You can use this property along with JavaScript to create very complex menu and very complex webpage layouts.

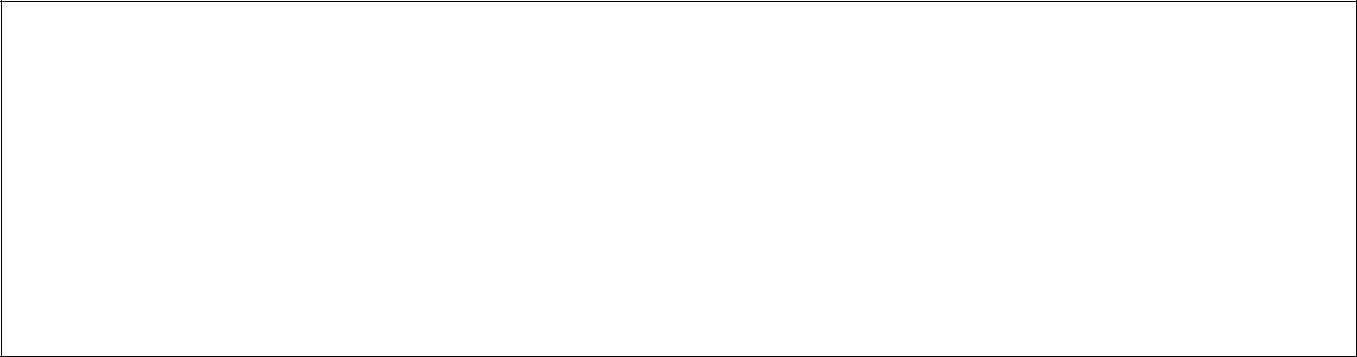
You may choose to use the visibility property to hide error messages that are only displayed if the user needs to see them, or to hide answers to a quiz until the user selects an option.

**NOTE:** Remember that the source code will still contain whatever is in the invisibleparagraph, so you should not use this to hide sensitive information such as credit card details or passwords.

The *visibility* property can take the values listed in the table that follows:

|  |  |
| --- | --- |
| **Value** | **Description** |
|  |  |
| visible | The box and its contents are shown to the user. |
|  |  |
| hidden | The box and its content are made invisible, although they still affect the |
|  | layout of the page. |
|  |  |
| collapse | This is for use only with dynamic table columns and row effects. |
|  |  |

Here is an example:



<p>

This paragraph should be visible in normal way.

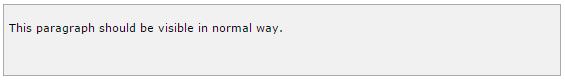
</p>

<p style="visibility:hidden;">

This paragraph should not be visible.

</p>

It will produce the following result:



**18. CSS ─ Media Types**

One of the most important features of style sheets is that they specify how a document is to be presented on different media: on the screen, on paper, with a speech synthesizer, with a braille device, etc.

We have currently two ways to specify media dependencies for style sheets:

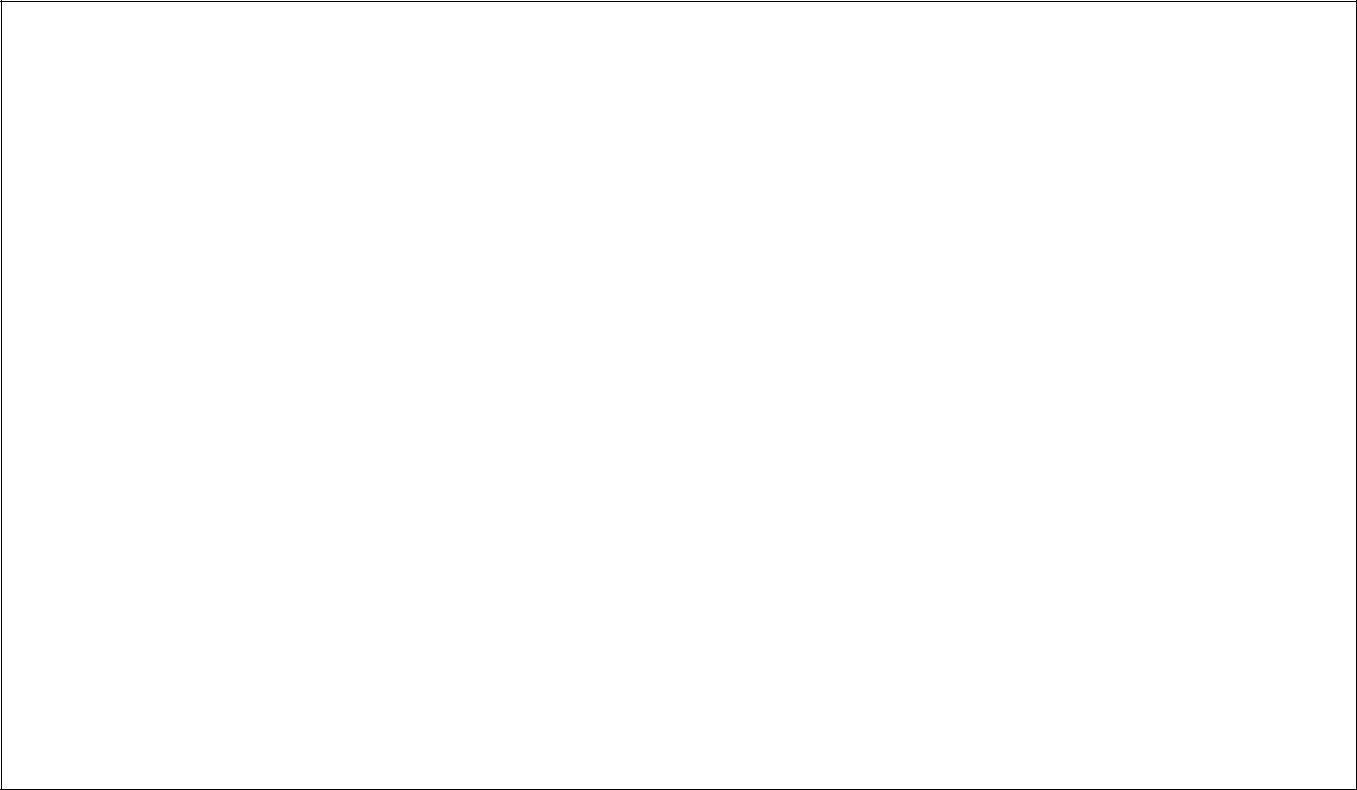
* Specify the target medium from a style sheet with the @media or @import at-rules.
* Specify the target medium within the document language.

**The @media rule**



The *@media* rule specifies the target media types (separated by commas) of a set of rules.

Given below is an example:

<style tyle="text/css">

@media print {

body { font-size: 10pt }

}

@media screen {

body { font-size: 12pt }

}

@media screen, print {

body { line-height: 1.2 }

}

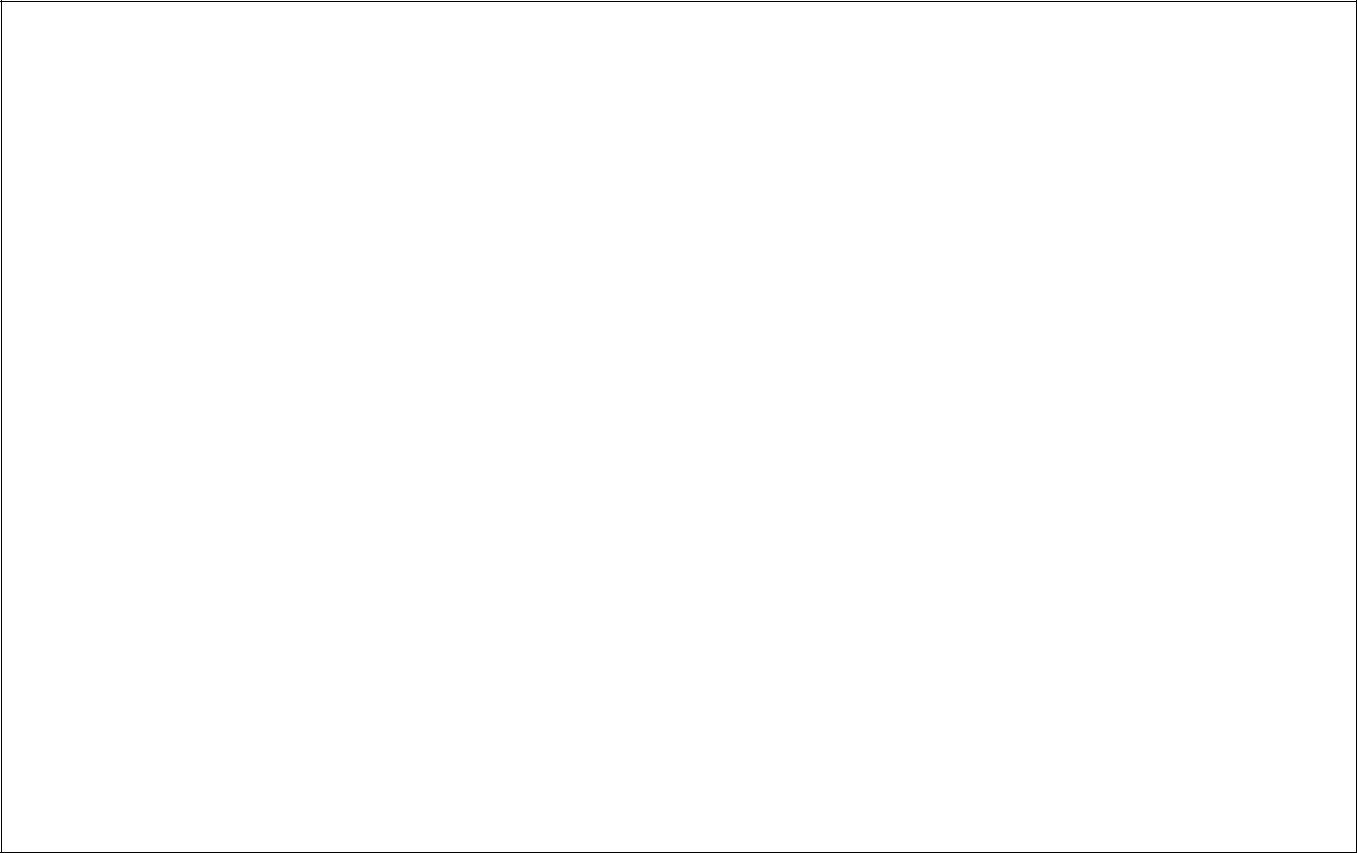
</style>

**The Document Language**



In HTML 4.0, the *media* attribute on the LINK element specifies the target media of an external style sheet.

Following is an example:



<!doctype html public "-//w3c//dtd html 4.0//en">

<html>

<head>

<title>link to a target medium</title> <link rel="stylesheet" type="text/css" media="print, handheld" href="foo.css">

</head>

<body>

<p>the body...</p>

</body>

</html>