



CSS

cascading style sheets

tutorialspoint

SIMPLY EASY LEARNING

www.tutorialspoint.com



<https://www.facebook.com/tutorialspointindia>



<https://twitter.com/tutorialspoint>

About the Tutorial

CSS is used to control the style of a web document in a simple and easy way. CSS stands for Cascading Style Sheets. This tutorial covers both the versions CSS1 and CSS2 and gives a complete understanding of CSS, starting from its basics to advanced concepts.

Audience

This tutorial will help both students as well as professionals who want to make their websites or personal blogs more attractive.

Prerequisites

You should be familiar with:

- Basic word processing using any text editor.
- How to create directories and files.
- How to navigate through different directories.
- Internet browsing using popular browsers like Internet Explorer or Firefox.
- Developing simple Web Pages using HTML or XHTML.

If you are new to HTML and XHTML, then we would suggest you to go through our HTML Tutorial or XHTML Tutorial first.

Copyright & Disclaimer

© Copyright 2015 by Tutorials Point (I) Pvt. Ltd.

All the content and graphics published in this e-book are the property of Tutorials Point (I) Pvt. Ltd. The user of this e-book is prohibited to reuse, retain, copy, distribute or republish any contents or a part of contents of this e-book in any manner without written consent of the publisher.

We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. Tutorials Point (I) Pvt. Ltd. provides no guarantee regarding the accuracy, timeliness or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at contact@tutorialspoint.com

Table of Contents

About the Tutorial	i
--------------------------	---

Audience.....	i
Prerequisites.....	i
Copyright & Disclaimer	i
Table of Contents.....	i
1. OVERVIEW	1
What is CSS?	1
Advantages of CSS	1
Who Creates and Maintains CSS?.....	2
CSS Versions	2
2. SYNTAX	3
The Type Selectors	3
The Universal Selectors.....	4
The Descendant Selectors	4
The Class Selectors.....	4
The ID Selectors	5
The Child Selectors.....	5
The Attribute Selectors	6
Multiple Style Rules	6
Grouping Selectors.....	7
3. INCLUSION	8
Embedded CSS - The <style> Element	8
Attributes	8
Inline CSS - The <i>style</i> Attribute	9
Attributes	9
External CSS - The <link> Element	10

Attributes	10
Imported CSS - @import Rule	11
CSS Rules Overriding	12
Handling Old Browsers.....	12
CSS Comments	12
4. MEASUREMENT UNITS.....	14
5. COLORS.....	16
CSS Colors - Hex Codes	16
CSS Colors - Short Hex Codes	17
CSS Colors - RGB Values	18
Building Color Codes	19
Browser Safe Colors	19
6. BACKGROUND.....	22
Set the Background Color.....	22
Set the Background Image	23
Repeat the Background Image	23
Set the Background Image Position.....	24
Set the Background Attachment	25
Shorthand Property	25
7. FONTS.....	26
Set the Font Family	26
Set the Font Style.....	26
Set the Font Variant.....	27
Set the Font Weight	27
Set the Font Size	28

Set the Font Size Adjust	29
Set the Font Stretch	29
Shorthand Property	30
8. TEXT	31
Set the Text Color	31
Set the Text Direction	32
Set the Space between Characters.....	32
Set the Space between Words	32
Set the Text Indent	33
Set the Text Alignment	33
Decorating the Text	34
Set the Text Cases	35
Set the White Space between Text	35
Set the Text Shadow	36
9. IMAGES.....	37
The Image Border Property	37
The Image Height Property	38
The Image Width Property	38
The -moz-opacity Property.....	39
10. LINKS.....	41
Set the Color of Links	41
Set the Color of Visited Links.....	42
Change the Color of Links when Mouse is Over.....	42
Change the Color of Active Links	43
11. TABLES	44

The order-collapse Property	44
The border-spacing Property	46
The caption-side Property.....	47
The empty-cells Property	49
The table-layout Property	51
12. BORDERS.....	53
The border-color Property	53
The border-style Property	54
The border-width Property	56
Border Properties Using Shorthand	58
13. MARGINS	59
The Margin Property.....	59
The margin-bottom Property	61
The margin-top Property.....	61
The margin-left Property.....	62
The margin-right Property	63
14. LISTS	64
The list-style-type Property.....	64
The list-style-position Property	67
The list-style-image Property	69
The list-style Property.....	70
The marker-offset Property	71
15. PADDINGS.....	72
The padding-bottom Property	72
The padding-top Property.....	73

The padding-left Property	73
The padding-right Property	74
The Padding Property	74
16. CURSORS.....	76
17. OUTLINES.....	79
The outline-width Property.....	79
The outline-style Property	80
The outline-color Property.....	81
The Outline Property	82
18. DIMENSION.....	84
The Height and Width Properties	84
The line-height Property	85
The max-height Property	86
The min-height Property	86
The max-width Property	87
The min-width Property.....	88
19. SCROLLBARS.....	90
20. VISIBILITY	93
21. POSITIONING	95
Relative Positioning	95
Absolute Positioning.....	96
Fixed Positioning.....	96
22. LAYERS	98
23. PSEUDO CLASSES	100

The :link pseudo-class	101
The :visited pseudo-class	101
The :hover pseudo-class.....	102
The :active pseudo-class	102
The :focus pseudo-class	102
The :first-child pseudo-class.....	103
The :lang pseudo-class	104
24. PSEUDO ELEMENTS.....	106
The :first-line pseudo-element.....	106
The :first-letter pseudo-element	107
The :before pseudo-element.....	108
The :after pseudo-element	108
25. @ RULES	110
The @import Rule.....	110
The @charset Rule	111
The @font-face Rule	111
The !important Rule.....	112
26. FILTERS	114
Alpha Channel.....	114
Motion Blur	115
Chroma Filter	117
Drop Shadow Effect	118
Flip Effect	120
Glow Effect	121
Grayscale Effect	122

Invert Effect	123
Mask Effect	124
Shadow Filter	125
Wave Effect.....	127
X-Ray Effect	128
27. MEDIA TYPES	130
The @media rule	130
The Document Language.....	131
Recognized Media Types.....	131
28. PAGED MEDIA.....	133
Defining Pages: The @page Rule	133
Setting Page Size	134
Left, Right, and First Pages	135
Controlling Pagination	136
Controlling Widows and Orphans	137
29. AURAL MEDIA	139
The azimuth Property	141
The elevation Property	142
The cue-after Property	142
The cue-before Property	143
The cue Property.....	143
The pause-after Property	144
The pause-before Property	144
The pause Property	144
The pitch Property	145

The pitch-range Property	145
The play-during Property	145
The richness Property	146
The speak Property	146
The speak-numeral Property.....	147
The speak-punctuation Property.....	147
The speech-rate Property.....	147
The stress Property	147
The voice-family Property	148
The volume Property	148
 30. PRINTING	 150
 31. LAYOUTS	 152
Sample Column Layout	153
 32. VALIDATIONS	 157
Why Validate Your HTML Code?	157
 33. CSS2 REFERENCE GUIDE.....	 159
Pseudo-classes & Pseudo-elements	167
 34. COLOR REFERENCES.....	 168

1. OVERVIEW

What is CSS?

Cascading Style Sheets, fondly referred to as **CSS**, is a simple design language intended to 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 in multiple 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 tag attributes 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 all the 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 for more 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 is being 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.

Who Creates and Maintains CSS?

CSS is created and maintained through a group of people within the W3C called the CSS Working Group. The CSS Working Group creates documents called **specifications**. When a specification has been discussed and officially ratified by the W3C members, it becomes a recommendation.

These ratified specifications are called recommendations because the W3C has no control over the actual implementation of the language. Independent companies and organizations create that software.

NOTE: The World Wide Web Consortium or W3C is a group that makes recommendations about how the Internet works and how it should evolve.

CSS Versions

Cascading Style Sheets level 1 (CSS1) came out of W3C as a recommendation in December 1996. This version describes the CSS language as well as a simple visual formatting model for all the HTML tags.

CSS2 became a W3C recommendation in May 1998 and builds on CSS1. This version adds support for media-specific style sheets e.g. printers and aural devices, downloadable fonts, element positioning and tables.

2. SYNTAX

A CSS comprises of style rules that are interpreted by the browser and then applied to the corresponding elements in your document. A style rule is made of three parts:

- **Selector:** A selector is an HTML tag at which a style will be applied. This could be any tag like `<h1>` or `<table>` etc.
- **Property:** A property is a type of attribute of HTML tag. Put simply, all the HTML attributes are converted into CSS properties. They could be *color*, *border*, etc.
- **Value:** Values are assigned to properties. For example, *color* property can have the 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 Universal Selectors

Rather than selecting elements of a specific type, the universal selector quite simply matches the name of any element type:

```
* {  
    color: #000000;  
}
```

This rule renders the content of every element in our document in black.

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 `` element only when it lies inside the `` 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 value of exactly "fr".
- **p[lang~="fr"]** - Selects all paragraph elements whose *lang* attribute contains the word "fr".
- **p[lang|="en"]** - Selects all paragraph elements whose *lang* attribute contains values 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 chapters 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;  
}
```

3. 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 tty tv projection handheld print braille	Specifies the device, the document will be displayed on. Default value is <i>all</i> . This is an optional attribute.

	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 rules	The value of <i>style</i> attribute is a combination of style 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 tty tv projection handheld print braille aural all	Specifies the device the document will be displayed on. Default value is <i>all</i> . This is an optional attribute.

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. */
```

4. 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 relative to another value, typically an enclosing element.	p {font-size: 16pt; line-height: 125%;}
cm	Defines a measurement in centimeters.	div {margin-bottom: 2cm;}
em	A relative measurement for the height of a font in em spaces. Because an em unit is equivalent to the size of a given font, if you assign a font to 12pt, each "em" unit would be 12pt; thus, 2em would be 24pt.	p {letter-spacing: 7em;}
ex	This value defines a measurement relative to a font's x-height. The x-height is determined by the height of the font's lowercase letter x.	p {font-size: 24pt; line-height: 3ex;}
in	Defines a measurement in inches.	p {word-spacing: .15in;}
mm	Defines a measurement in millimeters.	p {word-spacing: 15mm;}

pc	Defines a measurement in picas. A pica is equivalent to 12 points; thus, there are 6 picas per inch.	p {font-size: 20pc;}
pt	Defines a measurement in points. A point is defined as 1/72nd of an inch.	body {font-size: 18pt;}
px	Defines a measurement in screen pixels.	p {padding: 25px;}

5. 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;}

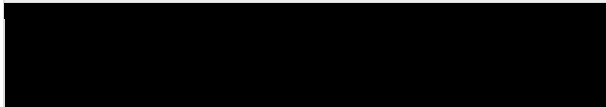








These formats are explained in more detail in the following sections:

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.

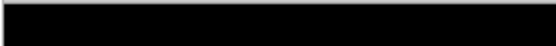







Color	Color HEX
	#000000
	#FF0000
	#00FF00
	#0000FF
	#FFFF00
	#00FFFF
	#FF00FF
	#C0C0C0
	#FFFFFF

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.

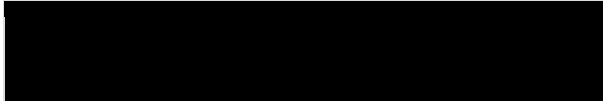





Color	Color HEX
	#000
	#F00
	#0F0
	#0FF
	#FF0
	#0FF
	#F0F
	#FFF

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 recommended not 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 216 colors, which are supposed to be most safe and computer independent colors. These colors vary from hexa code 000000 to FFFFFF. These colors are safe to use because they ensure that all computers would display the colors correctly when running a 256 color palette:

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
00FF00	00FF33	00FF66	00FF99	00FFCC	00FFFF
330000	330033	330066	330099	3300CC	3300FF

333300	333333	333366	333399	3333CC	3333FF
336600	336633	336666	336699	3366CC	3366FF
339900	339933	339966	339999	3399CC	3399FF
33CC00	33CC33	33CC66	33CC99	33CCCC	33CCFF
33FF00	33FF33	33FF66	33FF99	33FFCC	33FFFF
660000	660033	660066	660099	6600CC	6600FF
663300	663333	663366	663399	6633CC	6633FF
666600	666633	666666	666699	6666CC	6666FF
669900	669933	669966	669999	6699CC	6699FF
66CC00	66CC33	66CC66	66CC99	66CCCC	66CCFF
66FF00	66FF33	66FF66	66FF99	66FFCC	66FFFF
990000	990033	990066	990099	9900CC	9900FF
993300	993333	993366	993399	9933CC	9933FF
996600	996633	996666	996699	9966CC	9966FF
999900	999933	999966	999999	9999CC	9999FF

99CC00	99CC33	99CC66	99CC99	99CCCC	99CCFF
99FF00	99FF33	99FF66	99FF99	99FFCC	99FFFF
CC0000	CC0033	CC0066	CC0099	CC00CC	CC00FF
CC3300	CC3333	CC3366	CC3399	CC33CC	CC33FF
CC6600	CC6633	CC6666	CC6699	CC66CC	CC66FF
CC9900	CC9933	CC9966	CC9999	CC99CC	CC99FF
CCCC00	CCCC33	CCCC66	CCCC99	CCCCCC	CCCCFF
CCFF00	CCFF33	CCFF66	CCFF99	CCFFCC	CCFFFF
FF0000	FF0033	FF0066	FF0099	FF00CC	FF00FF
FF3300	FF3333	FF3366	FF3399	FF33CC	FF33FF
FF6600	FF6633	FF6666	FF6699	FF66CC	FF66FF
FF9900	FF9933	FF9966	FF9999	FF99CC	FF99FF
FFCC00	FFCC33	FFCC66	FFCC99	FFCCCC	FFCCFF
FFFF00	FFFF33	FFFF66	FFFF99	FFFFCC	FFFFFF

6. BACKGROUND

This chapter 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 paragraph 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 paragraph 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 paragraph has fixed repeated background image.  
</p>
```

7. FONTS

This chapter 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>
```

It will produce the following result:

This font is bold.
This font is bolder.
This font is 900 weight.

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>
```

It will produce the following result:

This font size is 20 pixels

This font size is small

This font size is large

Set the Font Size Adjust

The following example demonstrates how to set the font size adjust of an element. This property enables you to adjust the x-height to make fonts more legible. Possible value could be any number.

```
<p style="font-size-adjust:0.61;">  
This text is using a font-size-adjust value.  
</p>
```

It will produce the following result:

This text is using a font-size-adjust value.

Set the Font Stretch

The following example demonstrates how to set the font stretch of an element. This property relies on the user's computer to have an expanded or condensed version of the font being used.

Possible values could be *normal*, *wider*, *narrower*, *ultra-condensed*, *extra-condensed*, *condensed*, *semi-condensed*, *semi-expanded*, *expanded*, *extra-expanded*, *ultra-expanded*.

```
<p style="font-stretch:ultra-expanded;">  
If this doesn't appear to work, it is likely that  
your computer doesn't have a condensed or expanded  
version of the font being used.  
</p>
```

It will produce the following result:

If this doesn't appear to work, it is likely that your computer doesn't have a condensed or expanded version of the font being used.

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.

8. TEXT

This chapter 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 rendered from right to left  
</p>
```

It will produce the following result:

This text will be rendered 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 a over line.
</p>
<p style="text-decoration:blink;">
This text will have blinking effect
</p>
```

It will produce the following result:

This will be underlined

~~This will be striked through.~~

This will have a over line.

This text will have blinking effect

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>
```

It will produce the following result:

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

9. 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:

```
  
<br />  

```

It will produce the following result:



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:

```

<br />

```

It will produce the following result:



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:

```

<br />

```


It will produce the following result:



The -moz-opacity Property

The *-moz-opacity* property of an image is used to set the opacity of an image. This property is used to create a transparent image in Mozilla. IE uses **filter:alpha(opacity=x)** to create transparent images.

In Mozilla (*-moz-opacity:x*), x can be a value from 0.0 - 1.0. A lower value makes the element more transparent (The same things goes for the CSS3-valid syntax *opacity:x*).

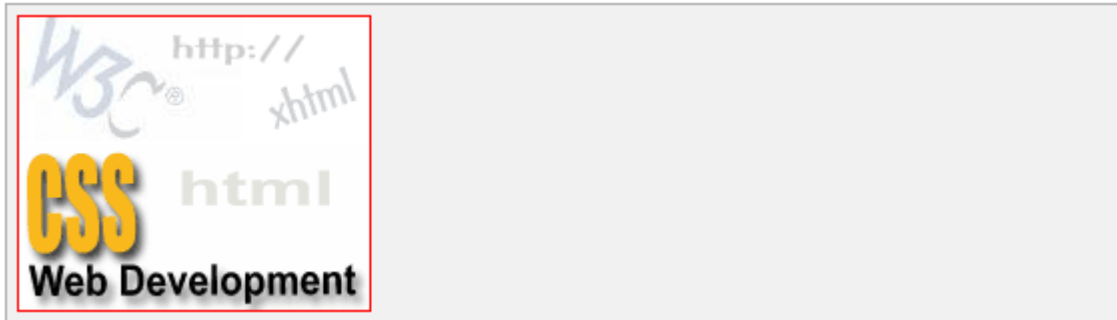
In IE (*filter:alpha(opacity=x)*), x can be a value from 0 - 100. A lower value makes the element more transparent.

Here is an example:

```

```

It will produce the following result:



10. LINKS

This chapter 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>
```

Now, we will see how to use these properties to give different effects to hyperlinks.

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>
```

It will produce the following black link:

[Black Link](#)

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>
```

It will produce the following link. Once you click this link, it will change its color to green.

[Click this link](#)

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>
```

It will produce the following link. Now, you bring your mouse over this link and you will see that it changes its color to yellow.

[Bring Mouse Here](#)

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>
```

It will produce the following link. It will change its color to pink when the user clicks it.

[Click This Link](#)

11. TABLES

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

- The **border-collapse** specifies whether the browser should control the appearance of the adjacent borders that touch each other or whether each cell should maintain its style.
- The **border-spacing** specifies the width that should appear between table cells.
- The **caption-side** captions are presented in the <caption> element. By default, these are rendered above the table in the document. You use the *caption-side* property to control the placement of the table caption.
- The **empty-cells** specifies whether the border should be shown if a cell is empty.
- 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.

Now, we will see how to use these properties with examples.

The border-collapse Property

This property can have two values *collapse* and *separate*. The following example uses both the values:

```
<style type="text/css">
table.one {border-collapse:collapse;}
table.two {border-collapse:separate;}
td.a {
    border-style:dotted;
    border-width:3px;
    border-color:#000000;
    padding: 10px;
}
```

```

td.b {border-style:solid;
      border-width:3px;
      border-color:#333333;
      padding:10px;
}
</style>
<table class="one">
<caption>Collapse Border Example</caption>
<tr><td class="a"> Cell A Collapse Example</td></tr>
<tr><td class="b"> Cell B Collapse Example</td></tr>
</table>
<br />
<table class="two">
<caption>Separate Border Example</caption>
<tr><td class="a"> Cell A Separate Example</td></tr>
<tr><td class="b"> Cell B Separate Example</td></tr>
</table>

```

It will produce the following result:

Collapse Border Example
Cell A Collapse Example
Cell B Collapse Example
Separate Border Example
Cell A Separate Example
Cell B Separate Example

The border-spacing Property

The border-spacing property specifies the distance that separates the adjacent cells' borders. It can take either one or two values; these should be units of length.

If you provide one value, it applies to both vertical and horizontal borders. Or you can specify two values, in which case, the first refers to the horizontal spacing and the second to the vertical spacing:

NOTE: Unfortunately, this property does not work in Netscape 7 or IE 6.

```
<style type="text/css">
/* If you provide one value */
table.example {border-spacing:10px;}
/* This is how you can provide two values */
table.example {border-spacing:10px; 15px;}
</style>
```

Now let's modify the previous example and see the effect:

```
<style type="text/css">
table.one {
    border-collapse:separate;
    width:400px;
    border-spacing:10px;
}
table.two {
    border-collapse:separate;
    width:400px;
    border-spacing:10px 50px;
}
</style>
<table class="one" border="1">
<caption>Separate Border Example with border-spacing</caption>
<tr><td> Cell A Collapse Example</td></tr>
<tr><td> Cell B Collapse Example</td></tr>
```



```

</table>
<br />
<table class="two" border="1">
<caption>Separate Border Example with border-spacing</caption>
<tr><td> Cell A Separate Example</td></tr>
<tr><td> Cell B Separate Example</td></tr>
</table>

```

It will produce the following result:

<table> <tr> <td>Separate Border Example with border-spacing</td></tr> <tr> <td>Cell A Collapse Example</td></tr> <tr> <td>Cell B Collapse Example</td></tr> </table>	Separate Border Example with border-spacing	Cell A Collapse Example	Cell B Collapse Example
Separate Border Example with border-spacing			
Cell A Collapse Example			
Cell B Collapse Example			
<table> <tr> <td>Separate Border Example with border-spacing</td></tr> <tr> <td>Cell A Separate Example</td></tr> <tr> <td>Cell B Separate Example</td></tr> </table>	Separate Border Example with border-spacing	Cell A Separate Example	Cell B Separate Example
Separate Border Example with border-spacing			
Cell A Separate Example			
Cell B Separate Example			

The caption-side Property

The `caption-side` property allows you to specify where the content of a `<caption>` element should be placed in relationship to the table. The table that follows lists the possible values.

This property can have one of the four values *top*, *bottom*, *left*, or *right*. The following example uses each value.

NOTE: These properties may not work with your IE Browser.

```

<style type="text/css">
caption.top {caption-side:top}
caption.bottom {caption-side:bottom}
caption.left {caption-side:left}

```

```
caption.right {caption-side:right}
</style>

<table style="width:400px; border:1px solid black;">
<caption class="top">
This caption will appear at the top
</caption>
<tr><td > Cell A</td></tr>
<tr><td > Cell B</td></tr>
</table>
<br />

<table style="width:400px; border:1px solid black;">
<caption class="bottom">
This caption will appear at the bottom
</caption>
<tr><td > Cell A</td></tr>
<tr><td > Cell B</td></tr>
</table>
<br />

<table style="width:400px; border:1px solid black;">
<caption class="left">
This caption will appear at the left
</caption>
<tr><td > Cell A</td></tr>
<tr><td > Cell B</td></tr>
</table>
<br />

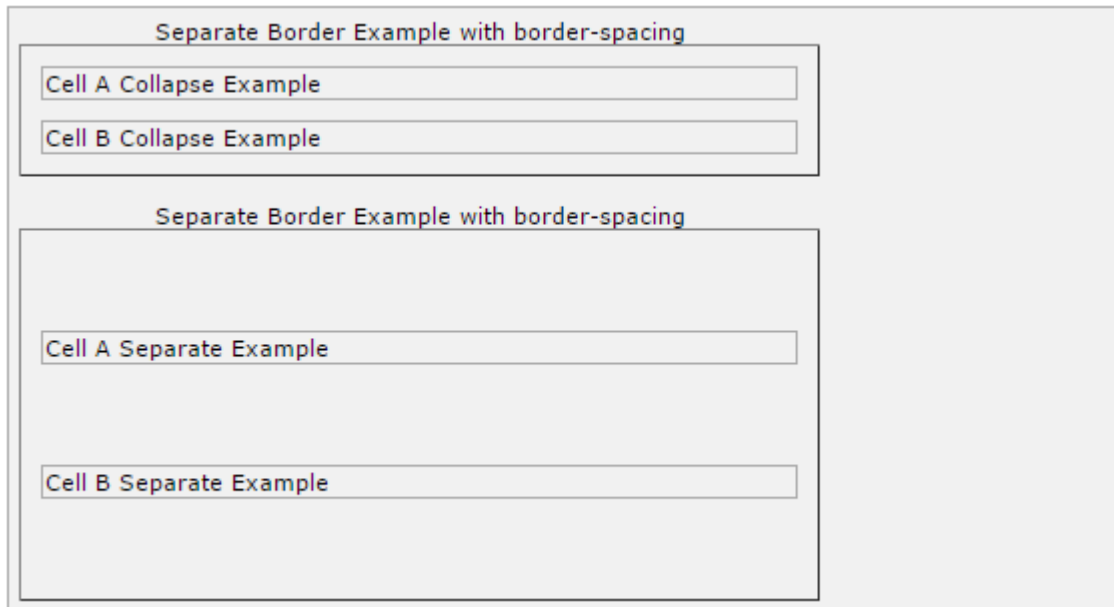
<table style="width:400px; border:1px solid black;">
```

```

<caption class="right">
This caption will appear at the right
</caption>
<tr><td > Cell A</td></tr>
<tr><td > Cell B</td></tr>
</table>

```

It will produce the following result:



The empty-cells Property

The empty-cells property indicates whether a cell without any content should have a border displayed.

This property can have one of the three values - *show*, *hide*, or *inherit*.

Here is the empty-cells property used to hide borders of empty cells in the `<table>` element.

```

<style type="text/css">
table.empty{
    width:350px;
    border-collapse:separate;

```

```

        empty-cells:hide;
    }
    td.empty{
        padding:5px;
        border-style:solid;
        border-width:1px;
        border-color:#999999;
    }
</style>
<table class="empty">
<tr>
<th></th>
<th>Title one</th>
<th>Title two</th>
</tr>
<tr>
<th>Row Title</th>
<td class="empty">value</td>
<td class="empty">value</td>
</tr>
<tr>
<th>Row Title</th>
<td class="empty">value</td>
<td class="empty"></td>
</tr>
</table>

```

It will produce the following result:

	Title one	Title two
Row Title	value	value
Row Title	value	

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.

```
<style type="text/css">
table.auto
{
table-layout: auto
}
table.fixed
{
table-layout: fixed
}
</style>
<table class="auto" border="1" width="100%">
<tr>
<td width="20%">100000000000000000000000000000</td>
<td width="40%">10000000</td>
<td width="40%">100</td>
</tr>
</table>
<br />
```

```
<table class="fixed" border="1" width="100%">
<tr>
<td width="20%">100000000000000000000000000000000</td>
<td width="40%">100000000</td>
<td width="40%">100</td>
</tr>
</table>
```

It will produce the following result:

100000000000000000000000000000000	100000000	100
100000000000000000000000000000000	10000000000	100

12. BORDERS

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:

- The **border-color** specifies the color of a border.
- The **border-style** specifies whether a border should be solid, dashed line, double line, or one of the other possible values.
- The **border-width** specifies the width of a border.

Now, we will see how to use these properties with examples.

The border-color Property

The `border-color` property allows you to change the color of the border surrounding an element. You can individually change the color of the bottom, left, top and right sides of an element's border using the properties:

- **border-bottom-color** changes the color of bottom border.
- **border-top-color** changes the color of top border.
- **border-left-color** changes the color of left border.
- **border-right-color** changes the color of right border.

The following example shows the effect of all these properties:

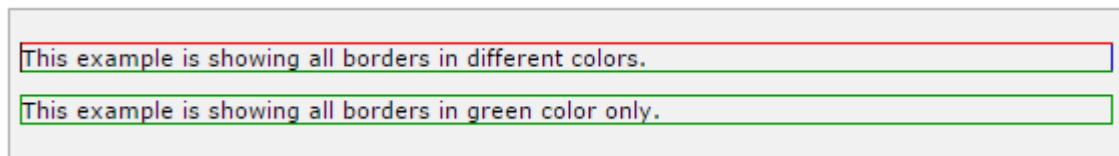
```
<style type="text/css">
p.example1{
    border:1px solid;
    border-bottom-color:#009900; /* Green */
    border-top-color:#FF0000;    /* Red */
    border-left-color:#330000;   /* Black */
    border-right-color:#0000CC;  /* Blue */
}
p.example2{
    border:1px solid;
```

```

    border-color:#009900;      /* Green */
}
</style>
<p class="example1">
This example is showing all borders in different colors.
</p>
<p class="example2">
This example is showing all borders in green color only.
</p>

```

It will produce the following result:



The border-style Property

The border-style property allows you to select one of the following styles of border:

- **none:** No border. (Equivalent of border-width:0;)
- **solid:** Border is a single solid line.
- **dotted:** Border is a series of dots.
- **dashed:** Border is a series of short lines.
- **double:** Border is two solid lines.
- **groove:** Border looks as though it is carved into the page.
- **ridge:** Border looks the opposite of groove.
- **inset:** Border makes the box look like it is embedded in the page.
- **outset:** Border makes the box look like it is coming out of the canvas.
- **hidden:** Same as none, except in terms of border-conflict resolution for table elements.

You can individually change the style of the bottom, left, top, and right borders of an element using the following properties:

- **border-bottom-style** changes the style of bottom border.
- **border-top-style** changes the style of top border.
- **border-left-style** changes the style of left border.
- **border-right-style** changes the style of right border.

The following example shows all these border styles:

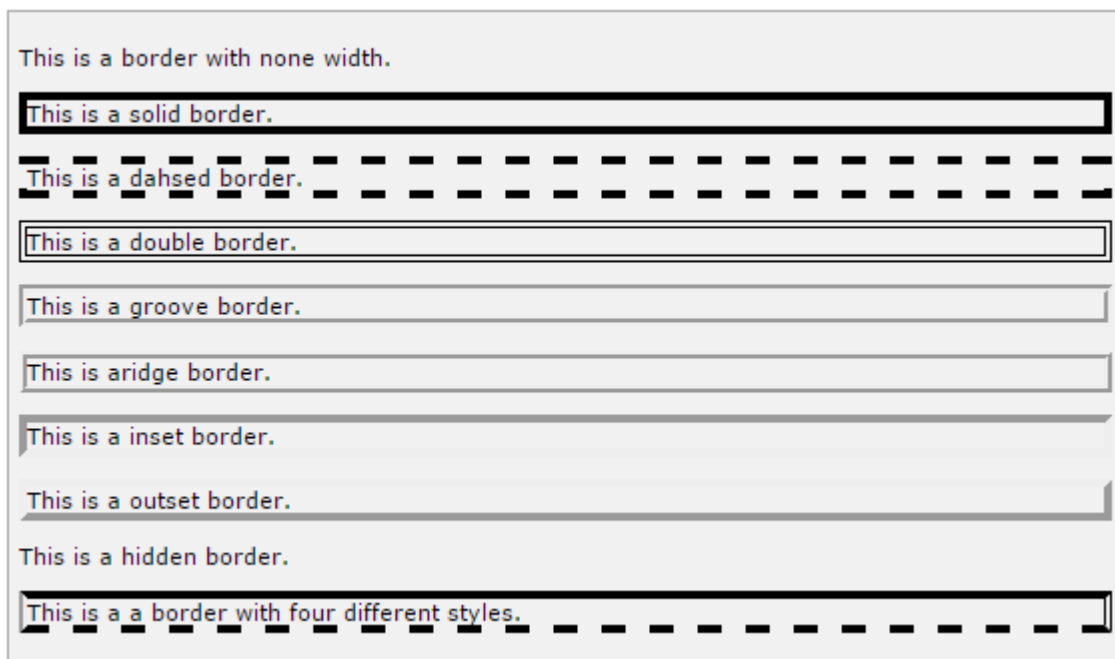
```
<p style="border-width:4px; border-style:none;">
This is a border with none width.
</p>
<p style="border-width:4px; border-style:solid;">
This is a solid border.
</p>
<p style="border-width:4px; border-style:dashed;">
This is a dahsed border.
</p>
<p style="border-width:4px; border-style:double;">
This is a double border.
</p>
<p style="border-width:4px; border-style:groove;">
This is a groove border.
</p>
<p style="border-width:4px; border-style:ridge">
This is aridge border.
</p>
<p style="border-width:4px; border-style:inset;">
This is a inset border.
</p>
<p style="border-width:4px; border-style:outset;">
This is a outset border.
</p>
<p style="border-width:4px; border-style:hidden;">
```

```

This is a hidden border.
</p>
<p style="border-width:4px;
        border-top-style:solid;
        border-bottom-style:dashed;
        border-left-style:groove;
        border-right-style:double;">
This is a a border with four different styles.
</p>

```

It will produce the following result:



The border-width Property

The border-width property allows you to set the width of an element borders. The value of this property could be either a length in px, pt, or cm, or it should be set to *thin*, *medium*, or *thick*.

You can individually change the width of the bottom, top, left, and right borders of an element using the following properties:

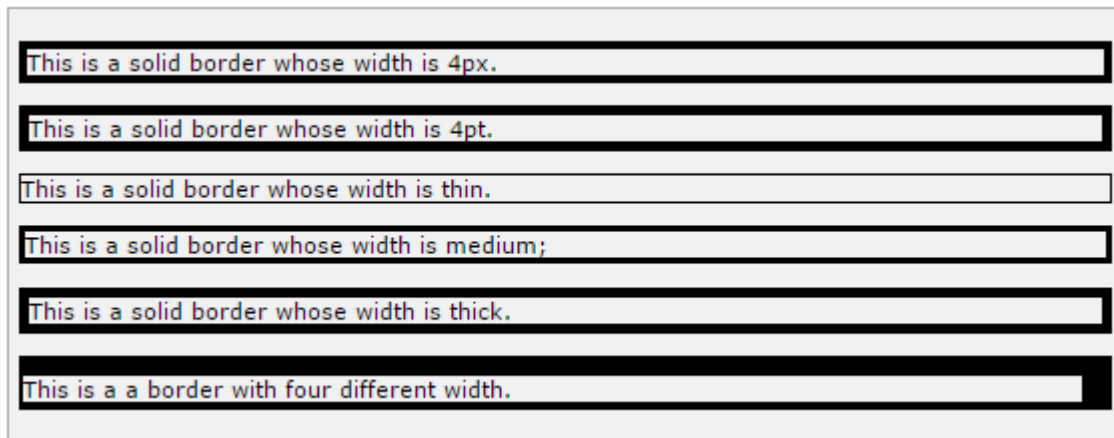
- **border-bottom-width** changes the width of bottom border.
- **border-top-width** changes the width of top border.

- **border-left-width** changes the width of left border.
- **border-right-width** changes the width of right border.

The following example shows all these border width:

```
<p style="border-width:4px; border-style:solid;">
This is a solid border whose width is 4px.
</p>
<p style="border-width:4pt; border-style:solid;">
This is a solid border whose width is 4pt.
</p>
<p style="border-width:thin; border-style:solid;">
This is a solid border whose width is thin.
</p>
<p style="border-width:medium; border-style:solid;">
This is a solid border whose width is medium;
</p>
<p style="border-width:thick; border-style:solid;">
This is a solid border whose width is thick.
</p>
<p style="border-bottom-width:4px;
        border-top-width:10px;
        border-left-width: 2px;
        border-right-width:15px;
        border-style:solid;">
This is a a border with four different width.
</p>
```

It will produce the following result:



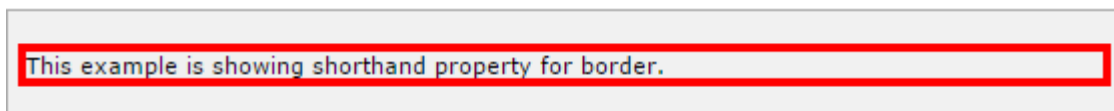
Border Properties Using Shorthand

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. 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.

Now, we will see how to use these properties with examples.

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:

```
<style type="text/css">
p {margin: 15px}
all four margins will be 15px

p {margin: 10px 2%}
top and bottom margin will be 10px, left and right margin will be 2% of
the total width of the document.
```

```
p {margin: 10px 2% -10px}
```

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 {margin: 10px 2% -10px auto}
```

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

```
</style>
```

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

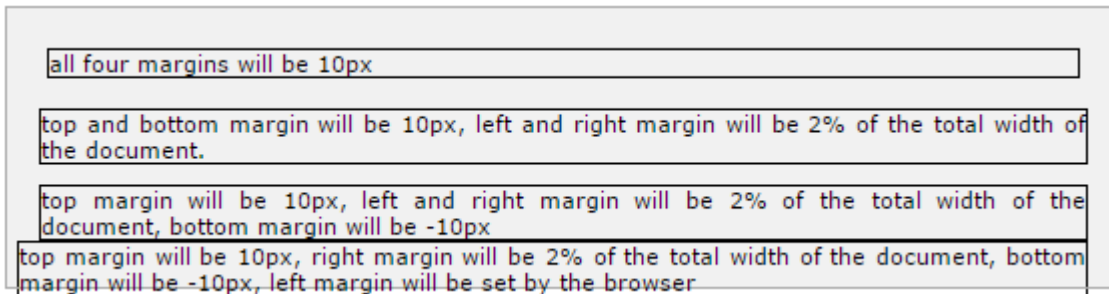
```
</p>
```

```
<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:



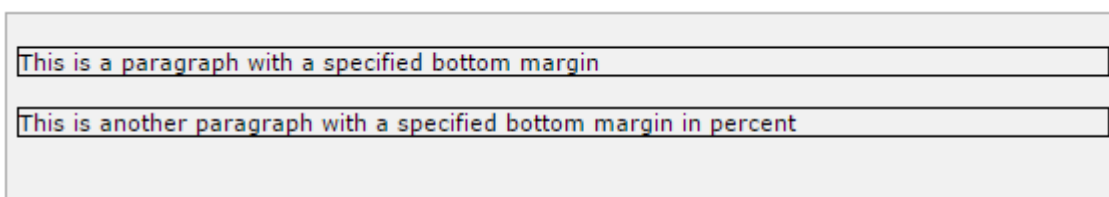
The margin-bottom Property

The margin-bottom property allows you to set the bottom margin of an element. It can have a value in length, %, or auto.

Here is an example:

```
<p style="margin-bottom: 15px; border:1px solid black;">
This is a paragraph with a specified bottom margin
</p>
<p style="margin-bottom: 5%; border:1px solid black;">
This is another paragraph with a specified bottom margin in percent
</p>
```

It will produce the following result:



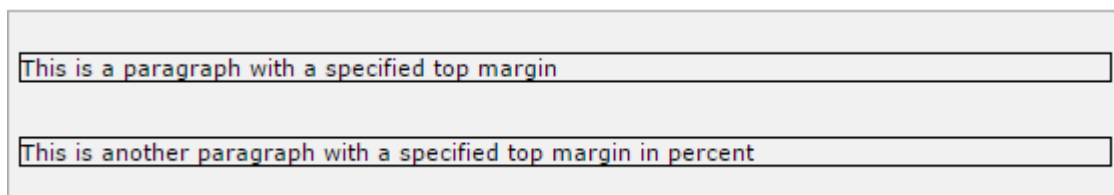
The margin-top Property

The margin-top property allows you to set the top margin of an element. It can have a value in length, %, or auto.

Here is an example:

```
<p style="margin-top: 15px; border:1px solid black;">  
This is a paragraph with a specified top margin  
</p>  
<p style="margin-top: 5%; border:1px solid black;">  
This is another paragraph with a specified top margin in percent  
</p>
```

It will produce the following result:



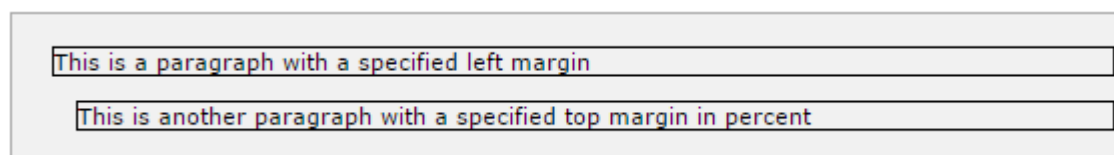
The margin-left Property

The margin-left property allows you to set the left margin of an element. It can have a value in length, %, or auto.

Here is an example:

```
<p style="margin-left: 15px; border:1px solid black;">  
This is a paragraph with a specified left margin  
</p>  
<p style="margin-left: 5%; border:1px solid black;">  
This is another paragraph with a specified top margin in percent  
</p>
```

It will produce the following result:



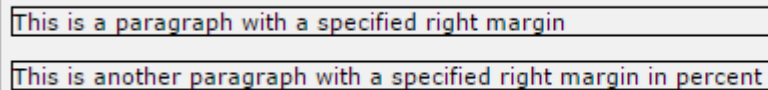
The margin-right Property

The margin-right property allows you to set the right margin of an element. It can have a value in length, %, or auto.

Here is an example:

```
<p style="margin-right: 15px; border:1px solid black;">  
This is a paragraph with a specified right margin  
</p>  
<p style="margin-right: 5%; border:1px solid black;">  
This is another paragraph with a specified right margin in percent  
</p>
```

It will produce the following result:



This is a paragraph with a specified right margin

This is another paragraph with a specified right margin in percent

14. LISTS

Lists are very helpful in conveying a set of either numbered or bulleted points. This chapter teaches you how to control list type, position, style, etc., using CSS.

We have the following five CSS properties, which can be used to control lists:

- The **list-style-type** allows you to control the shape or appearance of the marker.
- The **list-style-position** specifies whether a long point that wraps to a second line should align with the first line or start underneath the start of the marker.
- The **list-style-image** specifies an image for the marker rather than a bullet point or number.
- The **list-style** serves as shorthand for the preceding properties.
- The **marker-offset** specifies the distance between a marker and the text in the list.

Now we will see how to use these properties with examples.

The list-style-type Property

The *list-style-type* property allows you to control the shape or style of a bullet point (also known as a marker) in case of unordered lists and the style of numbering characters in ordered lists.

Here are the values, which can be used for an unordered list:

Value	Description
None	NA
disc (default)	A filled-in circle
Circle	An empty circle

Square	A filled-in square
--------	--------------------

Here are the values, which can be used for an ordered list:

Value	Description	Example
Decimal	Number	1,2,3,4,5
decimal-leading-zero	0 before the number	01, 02, 03, 04, 05
lower-alpha	Lowercase alphanumeric characters	a, b, c, d, e
upper-alpha	Uppercase alphanumeric characters	A, B, C, D, E
lower-roman	Lowercase Roman numerals	i, ii, iii, iv, v
upper-roman	Uppercase Roman numerals	I, II, III, IV, V
lower-greek	The marker is lower-greek	alpha, beta, gamma
lower-latin	The marker is lower-latin	a, b, c, d, e
upper-latin	The marker is upper-latin	A, B, C, D, E
hebrew	The marker is traditional Hebrew numbering	
armenian	The marker is traditional Armenian numbering	
georgian	The marker is traditional Georgian numbering	

cjk-ideographic	The marker is plain ideographic numbers	
hiragana	The marker is hiragana	a, i, u, e, o, ka, ki
katakana	The marker is katakana	A, I, U, E, O, KA, KI
hiragana-iroha	The marker is hiragana-iroha	i, ro, ha, ni, ho, he, to
katakana-iroha	The marker is katakana-iroha	I, RO, HA, NI, HO, HE, TO

Here is an example:

```
<ul style="list-style-type:circle;">
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ul>

<ul style="list-style-type:square;">
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ul>

<ol style="list-style-type:decimal;">
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
```

```

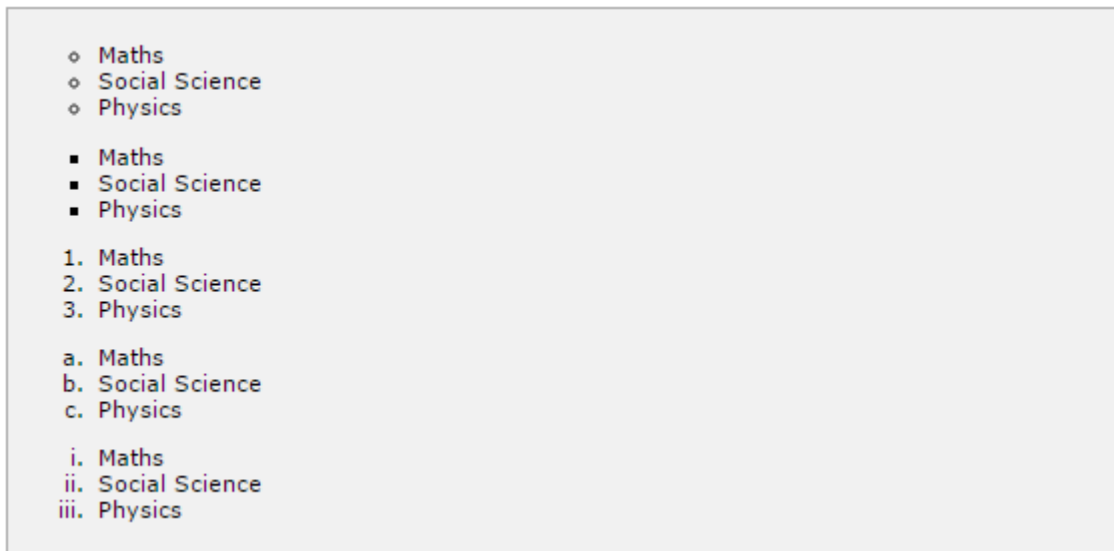
</ol>

<ol style="list-style-type:lower-alpha;">
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ol>

<ol style="list-style-type:lower-roman;">
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ol>

```

It will produce the following result:



The list-style-position Property

The *list-style-position* property indicates whether the marker should appear inside or outside of the box containing the bullet points. It can have one of the two values:

Value	Description
none	NA
inside	If the text goes onto a second line, the text will wrap underneath the marker. It will also appear indented to where the text would have started if the list had a value of outside.
outside	If the text goes onto a second line, the text will be aligned with the start of the first line (to the right of the bullet).

Here is an example:

```
<ul style="list-style-type:circle; list-stlye-position:outside;">
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ul>

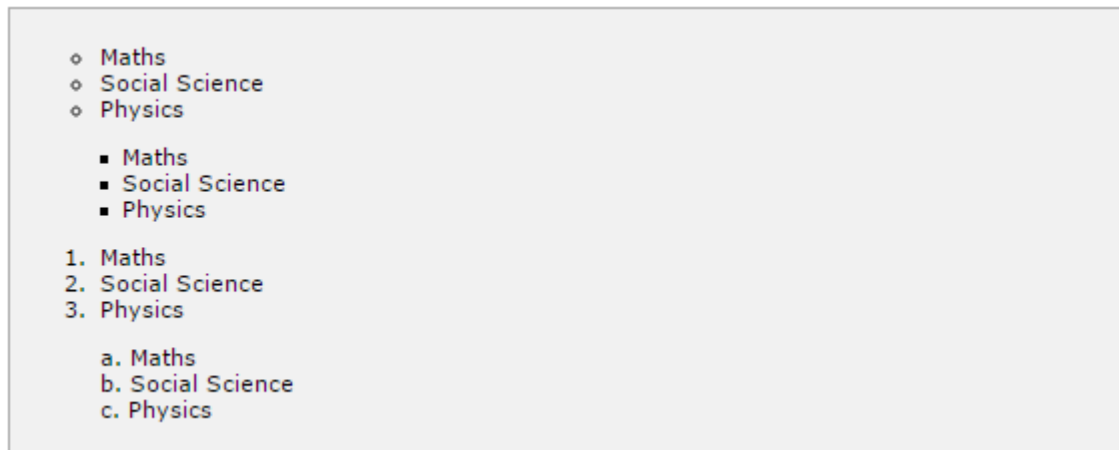
<ul style="list-style-type:square;list-style-position:inside;">
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ul>

<ol style="list-style-type:decimal;list-stlye-position:outside;">
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ol>

<ol style="list-style-type:lower-alpha;list-style-position:inside;">
```

```
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ol>
```

It will produce the following result:



The list-style-image Property

The *list-style-image* allows you to specify an image so that you can use your own bullet style. The syntax is similar to the background-image property with the letters url starting the value of the property followed by the URL in brackets. If it does not find the given image then default bullets are used.

Here is an example:

```
<ul>
<li style="list-style-image: url(/images/bullet.gif);">Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ul>

<ol>
<li style="list-style-image: url(/images/bullet.gif);">Maths</li>
<li>Social Science</li>
<li>Physics</li>
```

```
</ol>
```

It will produce the following result:

- Maths
 - Social Science
 - Physics
-
- Maths
 2. Social Science
 3. Physics

The list-style Property

The *list-style* allows you to specify all the list properties into a single expression. These properties can appear in any order.

Here is an example:

```
<ul style="list-style: inside square;">
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ul>

<ol style="list-style: outside upper-alpha;">
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ol>
```

It will produce the following result:

- Maths
 - Social Science
 - Physics
- A. Maths
 - B. Social Science
 - C. Physics

The marker-offset Property

The *marker-offset* property allows you to specify the distance between the marker and the text relating to that marker. Its value should be a length as shown in the following example:

Unfortunately, this property is not supported in IE 6 or Netscape 7.

Here is an example:

```
<ul style="list-style: inside square; marker-offset:2em;">
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ul>

<ol style="list-style: outside upper-alpha; marker-offset:2cm;">
<li>Maths</li>
<li>Social Science</li>
<li>Physics</li>
</ol>
```

It will produce the following result:

- Maths
 - Social Science
 - Physics
- A. Maths
 - B. Social Science
 - C. Physics

15. 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.

Now, we will see how to use these properties with examples.

The padding-bottom Property

The *padding-bottom* property sets the bottom padding (space) of an element. This can take a value in terms of length or %.

Here is an example:

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

This is a paragraph with a specified bottom padding

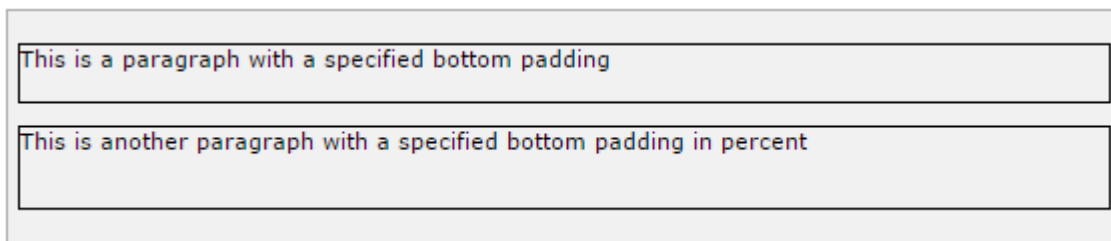
```
</p>
```

```
<p style="padding-bottom: 5%; border:1px solid black;">
```

This is another paragraph with a specified bottom padding in percent

```
</p>
```

It will produce the following result:



The padding-top Property

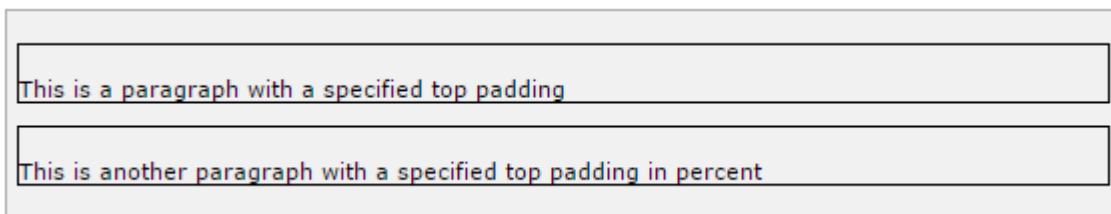
The *padding-top* property sets the top padding (space) of an element. This can take a value in terms of length of %.

Here is an example:

```
<p style="padding-top: 15px; border:1px solid black;">
This is a paragraph with a specified top padding
</p>

<p style="padding-top: 5%; border:1px solid black;">
This is another paragraph with a specified top padding in percent
</p>
```

It will produce the following result:



The padding-left Property

The *padding-left* property sets the left padding (space) of an element. This can take a value in terms of length of %.

Here is an example:

```
<p style="padding-left: 15px; border:1px solid black;">
This is a paragraph with a specified left padding
```

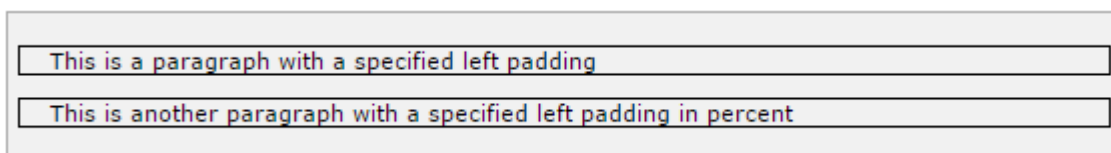
```
</p>
```

```
<p style="padding-left: 15%; border:1px solid black;">
```

```
This is another paragraph with a specified left padding in percent
```

```
</p>
```

It will produce the following result:



The padding-right Property

The *padding-right* property sets the right padding (space) of an element. This can take a value in terms of length of %.

Here is an example:

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

```
This is a paragraph with a specified right padding
```

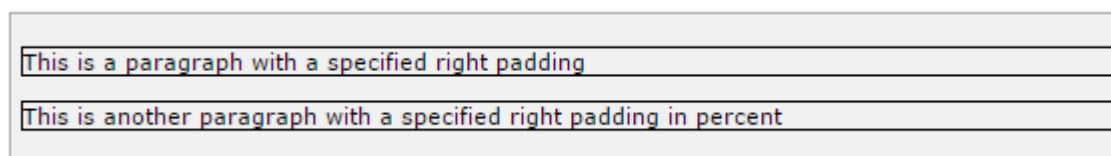
```
</p>
```

```
<p style="padding-right: 5%; border:1px solid black;">
```

```
This is another paragraph with a specified right padding in percent
```

```
</p>
```

It will produce the following result:



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>

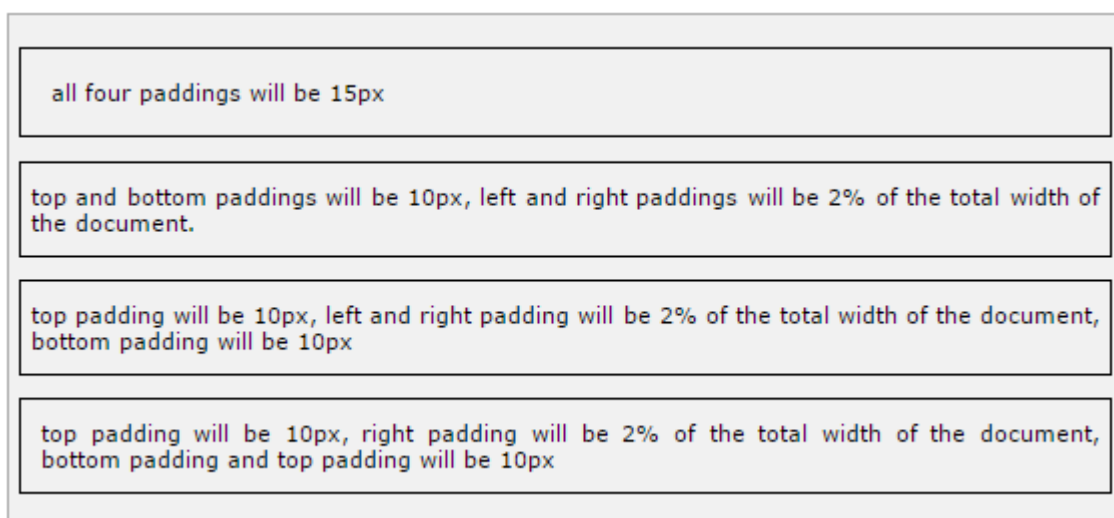
<p style="padding: 10px 2% 10px; border:1px solid black;">
top padding will be 10px, left and right padding will be 2% of the total
width of the document, bottom padding will be 10px

</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:



16. 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-resize	The cursor indicates that an edge of a box is to be moved up and left (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-resize	The cursor indicates that an edge of a box is to be moved down and left (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, and in 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>
```

It will produce the following result:

Move the mouse over the words to see the cursor change:

Auto
Crosshair
Default
Pointer
Move
e-resize
ne-resize
nw-resize
n-resize
se-resize
sw-resize
s-resize
w-resize
text
wait
help

17. OUTLINES

Outlines are very similar to borders, but there are few major differences as well:

- An outline does not take up space.
- Outlines do not have to be rectangular.
- Outline is always the same on all sides; you cannot specify different values for different sides of an element.

NOTE: The outline properties are not supported by IE 6 or Netscape 7.

You can set the following outline properties using CSS.

- The **outline-width** property is used to set the width of the outline.
- The **outline-style** property is used to set the line style for the outline.
- The **outline-color** property is used to set the color of the outline.
- The **outline** property is used to set all the above three properties in a single statement.

The outline-width Property

The *outline-width* property specifies the width of the outline to be added to the box. Its value should be a length or one of the values *thin*, *medium*, or *thick*, just like the border-width attribute.

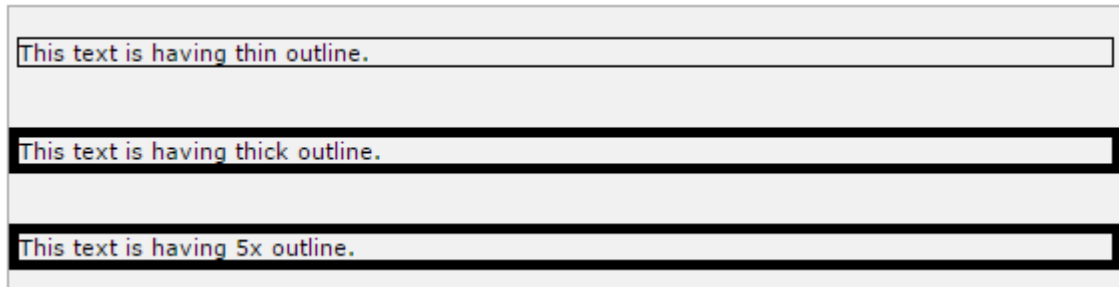
A width of zero pixels means no outline.

Here is an example:

```
<p style="outline-width:thin; outline-style:solid;">
This text is having thin outline.
</p>
<br />
<p style="outline-width:thick; outline-style:solid;">
This text is having thick outline.
</p>
<br />
```

```
<p style="outline-width:5px; outline-style:solid;">
This text is having 5x outline.
</p>
```

It will produce the following result:



The outline-style Property

The *outline-style* property specifies the style for the line (solid, dotted, or dashed) that goes around an element. It can take one of the following values:

- **none:** No border. (Equivalent of `outline-width:0;`)
- **solid:** Outline is a single solid line.
- **dotted:** Outline is a series of dots.
- **dashed:** Outline is a series of short lines.
- **double:** Outline is two solid lines.
- **groove:** Outline looks as though it is carved into the page.
- **ridge:** Outline looks the opposite of groove.
- **inset:** Outline makes the box look like it is embedded in the page.
- **outset:** Outline makes the box look like it is coming out of the canvas.
- **hidden:** Same as none.

Here is an example:

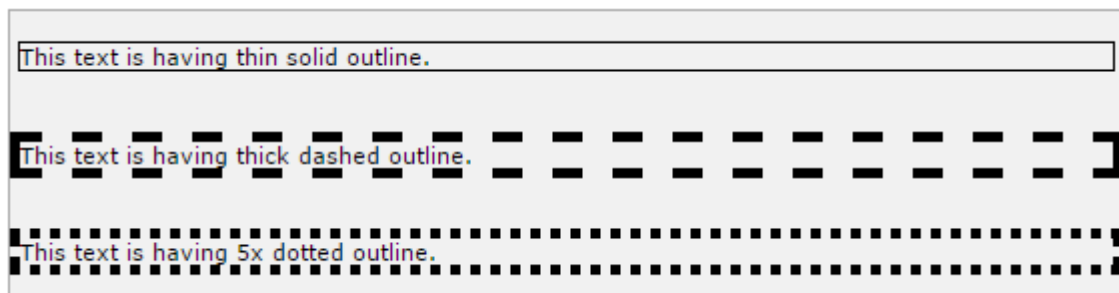
```
<p style="outline-width:thin; outline-style:solid;">
This text is having thin solid outline.
</p>
<br />
```

```

<p style="outline-width:thick; outline-style:dashed;">
This text is having thick dashed outline.
</p>
<br />
<p style="outline-width:5px;outline-style:dotted;">
This text is having 5x dotted outline.
</p>

```

It will produce the following result:



The outline-color Property

The *outline-color* property allows you to specify the color of the outline. Its value should either be a color name, a hex color, or an RGB value, as with the color and border-color properties.

Here is an example:

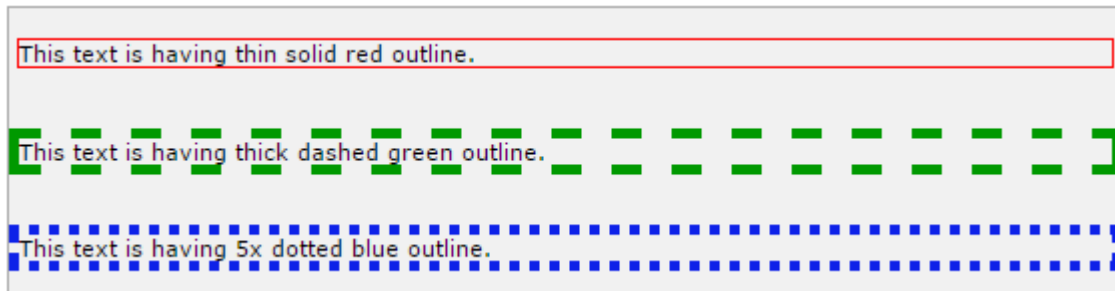
```

<p style="outline-width:thin; outline-style:solid;
        outline-color:red">
This text is having thin solid red  outline.
</p>
<br />
<p style="outline-width:thick; outline-style:dashed;
        outline-color:#009900">
This text is having thick dashed green outline.
</p>
<br />

```

```
<p style="outline-width:5px;outline-style:dotted;
        outline-color:rgb(13,33,232)">
This text is having 5x dotted blue outline.
</p>
```

It will produce the following result:



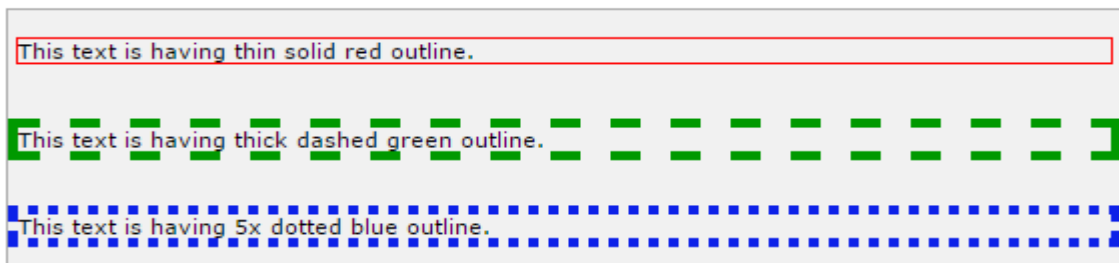
The Outline Property

The *outline* property is a shorthand property that allows you to specify values for any of the three properties discussed previously in any order but in a single statement.

Here is an example:

```
<p style="outline:thin solid red;">
This text is having thin solid red outline.
</p>
<br />
<p style="outline:thick dashed #009900;">
This text is having thick dashed green outline.
</p>
<br />
<p style="outline:5px dotted rgb(13,33,232);">
This text is having 5x dotted blue outline.
</p>
```

It will produce the following result:



18. 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 chapter, 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.

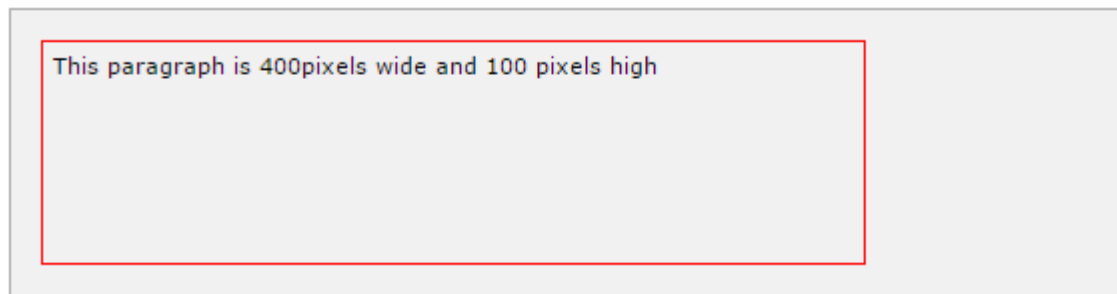
The Height and Width Properties

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:



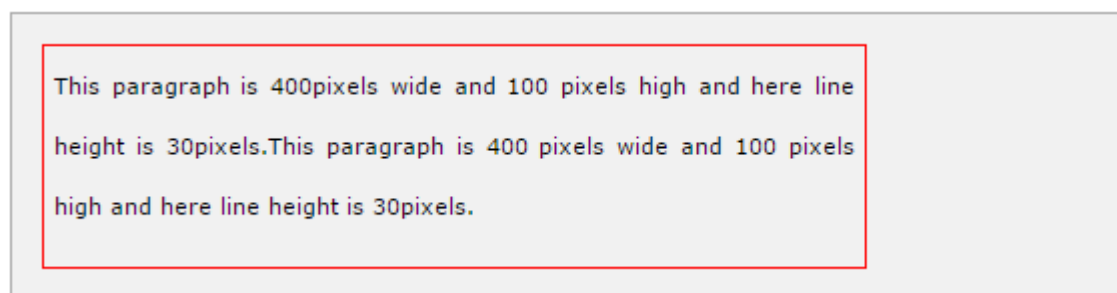
The line-height Property

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.

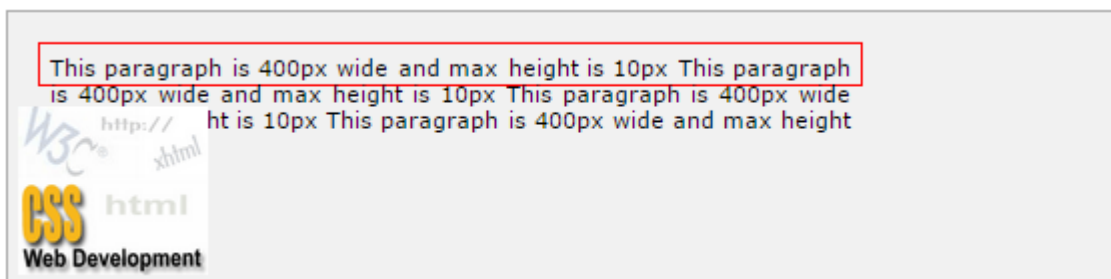
NOTE: This property does not work in either Netscape 7 or IE 6.

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>

```

It will produce the following result:



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
```

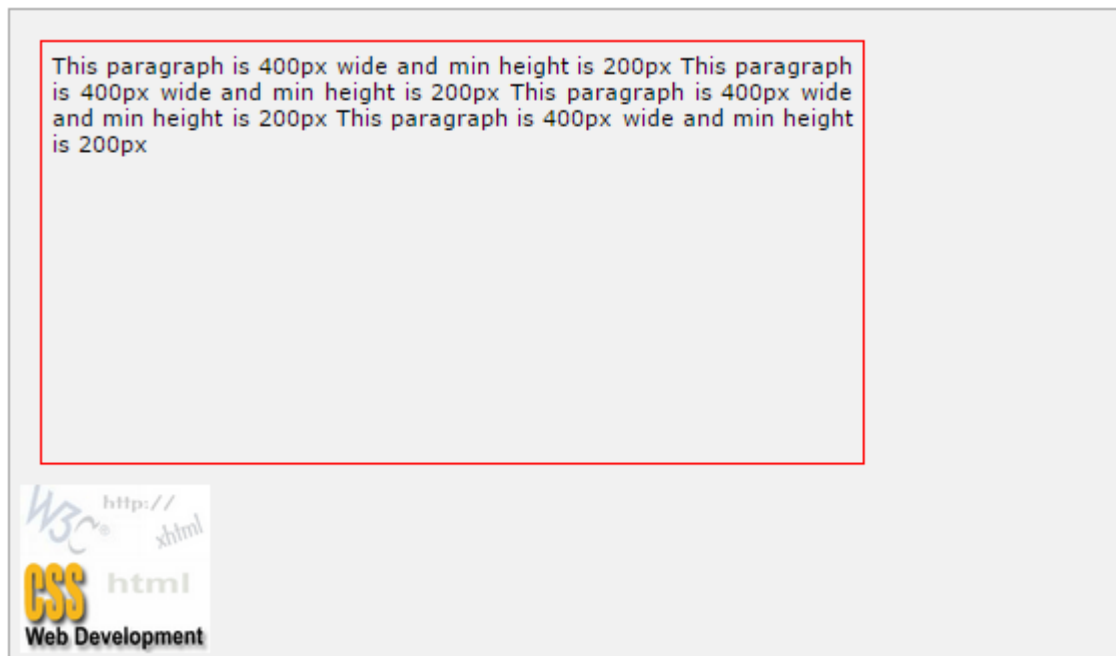


```

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>


```

It will produce the following result:



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

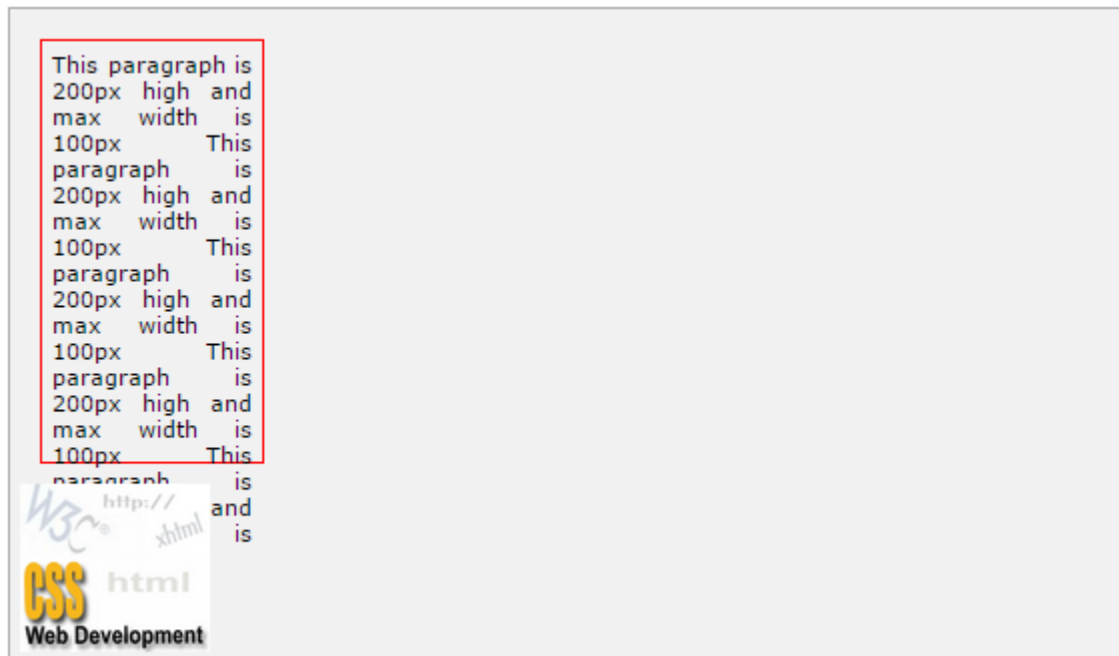
```

```

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


```

It will produce the following result:



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:

```

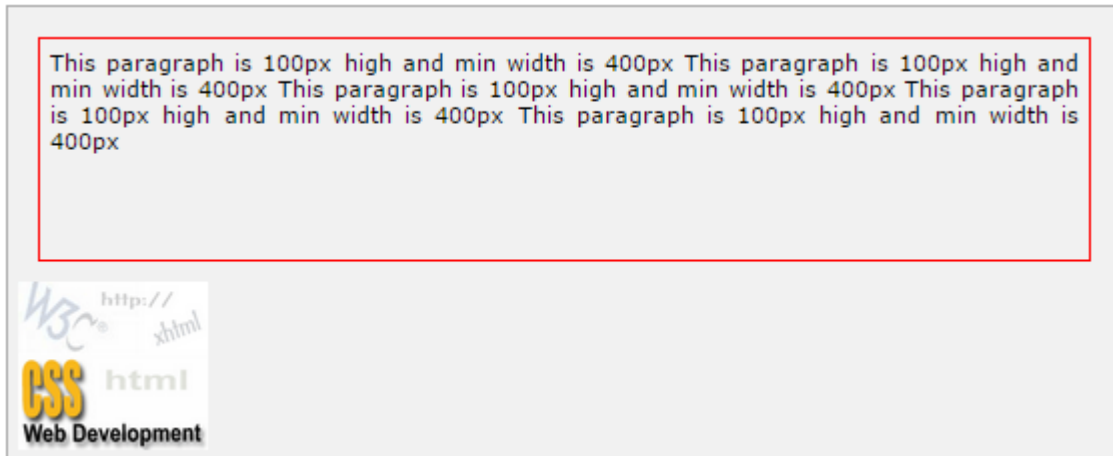
<p style="min-width:400px; height:100px;border:1px solid red;
padding:5px; margin:10px;">
This paragraph is 100px high and min width is 400px
This paragraph is 100px high and min width is 400px
This paragraph is 100px high and min width is 400px
This paragraph is 100px high and min width is 400px

```

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

```

It will produce the following result:



19. 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:



20. VISIBILITY

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 invisible paragraph, 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:

This paragraph should be visible in normal way.

21. POSITIONING

CSS helps you to position your HTML element. You can put any HTML element at whatever location you like. You can specify whether you want the element positioned relative to its natural position in the page or absolute based on its parent element.

Now, we will see all the CSS positioning related properties with examples.

Relative Positioning

Relative positioning changes the position of the HTML element relative to where it normally appears. So "left:20" adds 20 pixels to the element's LEFT position.

You can use two values *top* and *left* along with the *position* property to move an HTML element anywhere in an HTML document.

- Move Left - Use a negative value for *left*.
- Move Right - Use a positive value for *left*.
- Move Up - Use a negative value for *top*.
- Move Down - Use a positive value for *top*.

NOTE: You can use the *bottom* or *right* values as well in the same way as *top* and *left*.

Here is an example:

```
<div style="position:relative;left:80px;top:2px;  
        background-color:yellow;">  
This div has relative positioning.  
</div>
```

It will produce the following result:

This div has relative positioning.

Absolute Positioning

An element with **position: absolute** is positioned at the specified coordinates relative to your screen top-left corner.

You can use two values *top* and *left* along with the *position* property to move an HTML element anywhere in HTML document.

- Move Left - Use a negative value for *left*.
- Move Right - Use a positive value for *left*.
- Move Up - Use a negative value for *top*.
- Move Down - Use a positive value for *top*.

NOTE: You can use *bottom* or *right* values as well in the same way as *top* and *left*.

Here is an example:

```
<div style="position:absolute;left:80px;top:20px;  
        background-color:yellow;">  
This div has absolute positioning.  
</div>
```

Fixed Positioning

Fixed positioning allows you to fix the position of an element to a particular spot on the page, regardless of scrolling. Specified coordinates will be relative to the browser window.

You can use two values *top* and *left* along with the *position* property to move an HTML element anywhere in the HTML document.

- Move Left - Use a negative value for *left*.
- Move Right - Use a positive value for *left*.
- Move Up - Use a negative value for *top*.
- Move Down - Use a positive value for *top*.

NOTE: You can use *bottom* or *right* values as well in the same way as *top* and *left*.

Here is an example:

```
<div style="position:fixed;left:80px;top:20px;  
        background-color:yellow;">
```

This div has fixed positioning.

```
</div>
```

22. LAYERS

CSS gives you an opportunity to create layers of various divisions. The CSS layers refer to applying the *z-index* property to elements that overlap with each other.

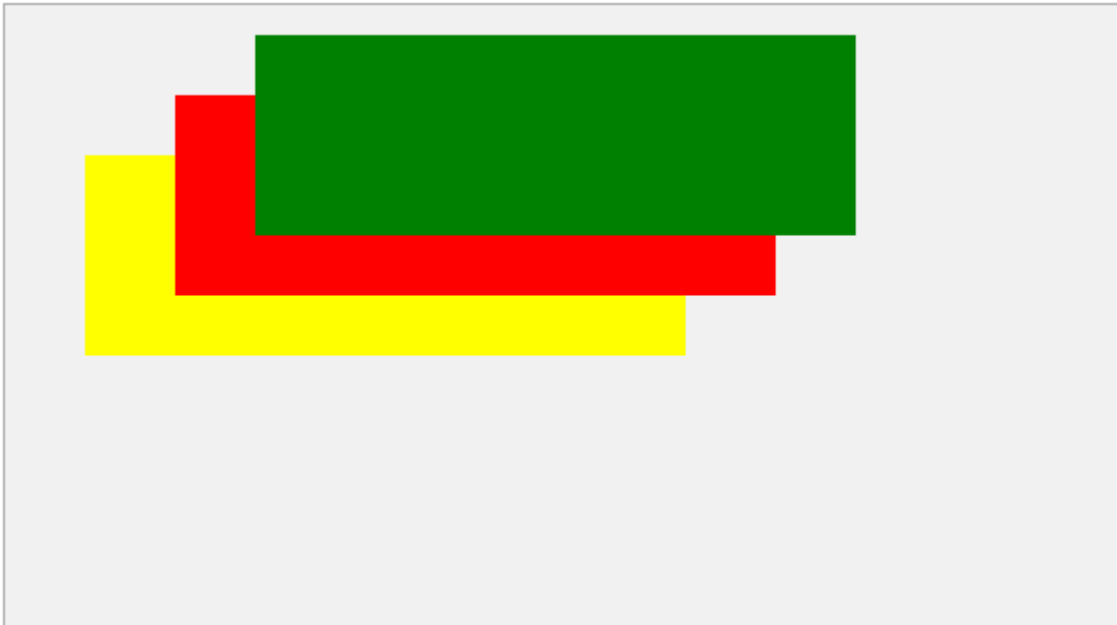
The z-index property is used along with the *position* property to create an effect of layers. You can specify which element should come on top and which element should come at bottom.

A z-index property can help you to create more complex webpage layouts. The following example shows how to create layers in CSS.

```
<div style="background-color:red;
    width:300px;
    height:100px;
    position:relative;
    top:10px;
    left:80px;
    z-index:2">
</div>
<div style="background-color:yellow;
    width:300px;
    height:100px;
    position:relative;
    top:-60px;
    left:35px;
    z-index:1;">
</div>
<div style="background-color:green;
width:300px;
    height:100px;
    position:relative;
    top:-220px;
```

```
left:120px;  
z-index:3;">  
</div>
```

It will produce the following result:



23. PSEUDO CLASSES

CSS pseudo-classes are used to add special effects to some selectors. You do not need to use JavaScript or any other script to use those effects. A simple syntax of pseudo-classes is as follows:

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

CSS classes can also be used with pseudo-classes:

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

The most commonly used pseudo-classes are as follows:

Value	Description
:link	Use this class to add special style to an unvisited link.
:visited	Use this class to add special style to a visited link.
:hover	Use this class to add special style to an element when you mouse over it.
:active	Use this class to add special style to an active element.
:focus	Use this class to add special style to an element while the element has focus.
:first-child	Use this class to add special style to an element that is the first child of some other element.
:lang	Use this class to specify a language to use in a specified element.

While defining pseudo-classes in a `<style>...</style>` block, the following points should be noted:


- `a:hover` MUST come after `a:link` and `a:visited` in the CSS definition in order to be effective.
- `a:active` MUST come after `a:hover` in the CSS definition in order to be effective.
- Pseudo-class names are not case-sensitive.
- Pseudo-classes are different from CSS classes, but they can be combined.

The `:link` pseudo-class

The following example demonstrates how to use `:link` class 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>
```

It will produce the following black link:




The `:visited` pseudo-class

The following example demonstrates how to use `:visited` class 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>
```

It will produce the following link. Once you click this link, it will change its color to green.



The :hover pseudo-class

The following example demonstrates how to use the *:hover* class 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>
```

It will produce the following link. Now you bring your mouse over this link and you will see that it changes its color to yellow.

[Bring Mouse Here](#)

The :active pseudo-class

The following example demonstrates how to use the *:active* class 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>
```

It will produce the following link. When a user clicks it, the color changes to pink.

[Click This Link](#)

The :focus pseudo-class

The following example demonstrates how to use the *:focus* class to change the color of the focused links. Possible values could be any color name in any valid format.

```
<style type="text/css">
a:focus {color: #0000FF}
</style>
```



```
</style>
<a href="/html/index.htm">Click this Link</a>
```

It will produce the following link. When this link gets focused, its color changes to orange. The color changes back when it loses focus.

[Click This Link](#)

The :first-child pseudo-class

The *:first-child* pseudo-class matches a specified element that is the first child of another element and adds special style to that element that is the first child of some other element.

To make *:first-child* work in IE <!DOCTYPE> must be declared at the top of document.

For example, to indent the first paragraph of all <div> elements, you could use this definition:

```
<style type="text/css">
div > p:first-child
{
text-indent: 25px;
}
</style>
<div>
<p>
First paragraph in div. This paragraph will be indented
</p>
<p>
Second paragraph in div. This paragraph will not be indented
</p>
</div>
<p>But it will not match the paragraph in this HTML:</p>
<div>
<h3>Heading</h3>
```

```
<p>
The first paragraph inside the div.
This paragraph will not be effected.
</p>
</div>
```

It will produce the following result:

```
First paragraph in div. This paragraph will be indented
Second paragraph in div. This paragraph will not be indented
But it will not match the paragraph in this HTML:

Heading

The first paragraph inside the div.
This paragraph will not be effected.
```

The :lang pseudo-class

The language pseudo-class *:lang*, allows constructing selectors based on the language setting for specific tags.

This class is useful in documents that must appeal to multiple languages that have different conventions for certain language constructs. For example, the French language typically uses angle brackets (< and >) for quoting purposes, while the English language uses quote marks (' and ').

In a document that needs to address this difference, you can use the *:lang* pseudo-class to change the quote marks appropriately. The following code changes the <blockquote> tag appropriately for the language being used:

```
<style type="text/css">
/* Two levels of quotes for two languages*/
:lang(en) { quotes: ' ' ' '; }
:lang(fr) { quotes: "<<" ">>" "<" ">"; }
</style>
<p>...<q lang="fr">A quote in a paragraph</q>...</p>
```

The :lang selectors will apply to all the elements in a document. However, not all elements make use of the quotes property, so the effect will be transparent for most elements.

```
...<<A quote in a paragraph>>...
```

24. PSEUDO ELEMENTS

CSS pseudo-elements are used to add special effects to some selectors. You do not need to use JavaScript or any other script to use those effects. A simple syntax of pseudo-element is as follows:

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

CSS classes can also be used with the pseudo-elements:

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

The most commonly used pseudo-elements are as follows:

Value	Description
:first-line	Use this element to add special styles to the first line of the text in a selector.
:first-letter	Use this element to add special style to the first letter of the text in a selector.
:before	Use this element to insert some content before an element.
:after	Use this element to insert some content after an element.

The :first-line pseudo-element

The following example demonstrates how to use the *:first-line* element to add special effects to the first line of elements in a document.

```
<style type="text/css">
p:first-line { text-decoration: underline; }
p.noline:first-line { text-decoration: none; }
</style>
<p class="noline"> This line would not have any underline
```

```
because this belongs to nline class.</p>
```

```
<p>The first line of this paragraph will be underlined
as defined in the CSS rule above. Rest of the lines in this
paragraph will remain normal. This example shows how to use
:first-line pseduo element to give effect to the first line
of any HTML element.</p>
```

It will produce the following link:

This line would not have any underline because this belongs to nline class.

The first line of this paragraph will be underlined as defined in the CSS rule above. Rest of the lines in this paragraph will remain normal. This example shows how to use :first-line pseduo element to give effect to the first lines of any HTML element.

The :first-letter pseudo-element

The following example demonstrates how to use the *:first-letter* element to add special effect to the first letter of elements in the document.

```
<style type="text/css">
p:first-letter { font-size: 5em; text-color:red; }
p.normal:first-letter { font-size: 10px; }
</style>
<p class="normal"> First character of this paragraph will
be normal and will have font size 10 px;</p>

<p>The first character of this paragraph will be 5em big
and in red color as defined in the CSS rule above. Rest of the
characters in this paragraph will remain normal. This example
shows how to use :first-letter pseduo element to give effect to
the first characters of any HTML element.</p>
```

It will produce the following black link:

First character of this paragraph will be normal and will have font size 10 px;

The first character of this paragraph will be 3em big and in red color as defined in the CSS rule above. Rest of the characters in this paragraph will remain normal. This example shows how to use `:first-letter` pseudo element to give effect to the first characters of any HTML element.

The `:before` pseudo-element

The following example demonstrates how to use `:before` element to add some content before any element.

```
<style type="text/css">
p:before
{
content: url(/images/bullet.gif)
}
</style>
<p> This line will be preceded by a bullet.</p>
<p> This line will be preceded by a bullet.</p>
<p> This line will be preceded by a bullet.</p>
```

It will produce the following black link:

■ This line will be preceded by a bullet.

■ This line will be preceded by a bullet.

■ This line will be preceded by a bullet.

The `:after` pseudo-element

The following example demonstrates how to use `:after` element to add some content after any element.

```
<style type="text/css">
```

```
p:after
{
content: url(/images/bullet.gif)
}
</style>
<p> This line will be succeeded by a bullet.</p>
<p> This line will be succeeded by a bullet.</p>
<p> This line will be succeeded by a bullet.</p>
```

It will produce the following black link:

```
This line will be succeeded by a bullet.■
This line will be succeeded by a bullet.■
This line will be succeeded by a bullet.■
```

25. @ RULES

This chapter will cover the following important @ rules:

- The **@import** rule imports another style sheet into the current style sheet.
- The **@charset** rule indicates the character set the style sheet uses.
- The **@font-face** rule is used to exhaustively describe a font face for use in a document.
- The **!important** rule indicates that a user-defined rule should take precedence over the author's style sheets.

NOTE: There are other @ rules which we will cover in subsequent chapters.

The @import Rule

The @import rule allows you to import styles from another style sheet. It should appear right at the start of the style sheet before any of the rules, and its value is a URL.

It can be written in one of the two following ways:

```
<style tyle="text/css">
<!--
@import "mystyle.css";

or

@import url("mystyle.css");
.....other CSS rules .....
-->
</style>
```

The significance of the @import rule is that it allows you to develop your style sheets with a modular approach. You can create various style sheets and then include them wherever you need them.

The @charset Rule

If you are writing your document using a character set other than ASCII or ISO-8859-1 you might want to set the @charset rule at the top of your style sheet to indicate what character set the style sheet is written in.

The @charset rule must be written right at the beginning of the style sheet without even a space before it. The value is held in quotes and should be one of the standard character-sets. For example:

```
<style tyle="text/css">
<!--
@charset "iso-8859-1"
.....other CSS rules .....
-->
</style>
```

The @font-face Rule

The @font-face rule is used to exhaustively describe a font face for use in a document. @font-face may also be used to define the location of a font for download, although this may run into implementation-specific limits.

In general, @font-face is extremely complicated, and its use is not recommended for any except those who are expert in font metrics.

Here is an example:

```
<style tyle="text/css">
<!--
@font-face {
    font-family: "Scarborough Light";
    src: url("http://www.font.site/s/scarbo-lt");
}
@font-face {
    font-family: Santiago;
    src: local ("Santiago"),
    url("http://www.font.site/s/santiago.tt")
```

```

format("truetype");
unicode-range: U+??,U+100-220;
font-size: all;
font-family: sans-serif;
}
-->
</style>

```

The !important Rule

Cascading Style Sheets cascade. It means that the styles are applied in the same order as they are read by the browser. The first style is applied and then the second and so on.

The !important rule provides a way to make your CSS cascade. It also includes the rules that are to be applied always. A rule having a !important property will always be applied, no matter where that rule appears in the CSS document.

For example, in the following style sheet, the paragraph text will be black, even though the first style property applied is red:

```

<style tyle="text/css">
<!--
p { color: #ff0000; }
p { color: #000000; }
-->
</style>

```

So, if you wanted to make sure that a property always applied, you would add the !important property to the tag. So, to make the paragraph text always red, you should write it as follows:

```

<style tyle="text/css">
<!--
p { color: #ff0000 !important; }
p { color: #000000; }
-->

```

```
</style>
```

Here you have made `p { color: #ff0000 !important; }` mandatory, now this rule will always apply even you have defined another rule `p { color: #000000; }`.

26. FILTERS

You can use CSS filters to add special effects to text, images and other aspects of a webpage without using images or other graphics. **Filters only work on Internet Explorer 4.0.** If you are developing your site for multiple browsers, then it may not be a good idea to use CSS filters because there is a possibility that it would not give any advantage.

In this chapter, we will see the details of each CSS filter. These filters may not work in your browser.

Alpha Channel

The Alpha Channel filter alters the opacity of the object, which makes it blend into the background. The following parameters can be used in this filter:

Parameter	Description
opacity	Level of the opacity. 0 is fully transparent, 100 is fully opaque.
finishopacity	Level of the opacity at the other end of the object.
Style	The shape of the opacity gradient. 0 = uniform 1 = linear 2 = radial 3 = rectangular
startX	X coordinate for opacity gradient to begin.
startY	Y coordinate for opacity gradient to begin.
finishX	X coordinate for opacity gradient to end.
finishY	Y coordinate for opacity gradient to end.

Example

```
<p>Image Example:</p>



<p>Text Example:</p>

<div style="width: 357; height: 50; font-size: 30pt; font-family: Arial
Black; color: blue; Filter: Alpha(Opacity=100, FinishOpacity=0, Style=1,
StartX=0, StartY=0, FinishX=580, FinishY=0)">CSS Tutorials</div>
```

It will produce the following result:



Motion Blur

Motion Blur is used to create blurred pictures or text with the direction and strength. The following parameters can be used in this filter:

Parameter	Description
add	True or false. If true, the image is added to the blurred image; and if false, the image is not added to the blurred image.

direction	<p>The direction of the blur, going clockwise, rounded to 45-degree increments. The default value is 270 (left).</p> <p>0 = Top</p> <p>45 = Top right</p> <p>90 = Right</p> <p>135 = Bottom right</p> <p>180 = Bottom</p> <p>225 = Bottom left</p> <p>270 = Left</p> <p>315 = Top left</p>
strength	<p>The number of pixels the blur will extend. The default is 5 pixels.</p>

Example

```
<p>Image Example:</p>
```

```

```

```
<p>Text Example:</p>
```

```
<div style="width: 357; height: 50; font-size: 30pt; font-family: Arial
Black; color: blue; Filter: Blur(Add = 1, Direction = 225, Strength =
10)">CSS Tutorials</div>
```

It will produce the following result:



Chroma Filter

Chroma Filter is used to make any particular color transparent and usually it is used with images. You can use it with scrollbars also. The following parameter can be used in this filter:

Parameter	Description
Color	The color that you'd like to be transparent.

Example

```
<p>Image Example:</p>



<p>Text Example:</p>
```

```
<div style="width: 580; height: 50; font-size: 30pt; font-family: Arial
Black; color: #3300FF; Filter: Chroma(Color = #3300FF)">CSS
Tutorials</div>
```

It will produce the following result:



Drop Shadow Effect

Drop Shadow is used to create a shadow of your object at the specified X (horizontal) and Y (vertical) offset and color.

The following parameters can be used in this filter:

Parameter	Description
Color	The color, in #RRGGBB format, of the dropshadow.
offX	Number of pixels the drop shadow is offset from the visual object, along the x-axis. Positive integers move the drop shadow to the right, negative integers move the drop shadow to the left.
offY	Number of pixels the drop shadow is offset from the visual object, along the y-axis. Positive integers move the drop shadow down, negative integers move the drop shadow up.

positive	If true, all opaque pixels of the object have a dropshadow. If false, all transparent pixels have a dropshadow. The default is true.
----------	--

Example

```
<p>Image Example:</p>



<p>Text Example:</p>

<div style="width: 357; height: 50; font-size: 30pt; font-family: Arial
Black; color: red; Filter: DropShadow(Color=#000000, OffX=2, OffY=2,
Positive=1)">CSS Tutorials</div>
```

It will produce the following result:



Flip Effect

Flip effect is used to create a mirror image of the object. The following parameters can be used in this filter:

Parameter	Description
FlipH	Creates a horizontal mirror image.
FlipV	Creates a vertical mirror image.

Example

```
<p>Image Example:</p>





<p>Text Example:</p>

<div style="width: 300; height: 50; font-size: 30pt; font-family: Arial
Black; color: red; Filter: FlipV">CSS Tutorials</div>
```

It will produce the following result:



Glow Effect

Glow effect is used to create a glow around the object. If it is a transparent image, then glow is created around the opaque pixels of it. The following parameters can be used in this filter:

Parameter	Description
color	The color you want the glow to be.
strength	The intensity of the glow (from 1 to 255).

Example

```
<p>Image Example:</p>



<p>Text Example:</p>

<div style="width: 357; height: 50; font-size: 30pt; font-family: Arial
Black; color: red; Filter: Glow(Color=#00FF00, Strength=20)">CSS
Tutorials</div>
```

It will produce the following result:



Grayscale Effect

Grayscale effect is used to convert the colors of the object to 256 shades of gray. The following parameter is used in this filter:

Parameter	Description
Gray	Converts the colors of the object to 256 shades of gray.

Example

```
<p>Image Example:</p>



<p>Text Example:</p>

<div style="width: 357; height: 50; font-size: 30pt; font-family: Arial
Black; color: red; Filter: Gray">CSS Tutorials</div>
```

It will produce the following result:



Invert Effect

Invert effect is used to map the colors of the object to their opposite values in the color spectrum, i.e., to create a negative image. The following parameter is used in this filter:

Parameter	Description
Invert	Maps the colors of the object to their opposite value in the color spectrum.

Example

```
<p>Image Example:</p>



<p>Text Example:</p>

<div style="width: 357; height: 50; font-size: 30pt; font-family: Arial
Black; color: red; Filter: invert">CSS Tutorials</div>
```

It will produce the following result:



Mask Effect

Mask effect is used to turn transparent pixels to a specified color and makes opaque pixels transparent. The following parameter is used in this filter:

Parameter	Description
Color	The color that the transparent areas will become.

Example

```
<p>Image Example:</p>



<p>Text Example:</p>

<div style="width: 357; height: 50; font-size: 30pt; font-family: Arial
Black; color: red; Filter: Mask(Color=#00FF00)">CSS Tutorials</div>
```

It will produce following result:



Shadow Filter

Shadow filter is used to create an attenuated shadow in the direction and color specified. This is a filter that lies in between Dropshadow and Glow. The following parameters can be used in this filter:

Parameter	Description
color	The color that you want the shadow to be.

direction	<p>The direction of the blur, going clockwise, rounded to 45-degree increments. The default value is 270 (left).</p> <p>0 = Top</p> <p>45 = Top right</p> <p>90 = Right</p> <p>135 = Bottom right</p> <p>180 = Bottom</p> <p>225 = Bottom left</p> <p>270 = Left</p> <p>315 = Top left</p>
-----------	--

Example

<p>Image Example:</p>

```

```

<p>Text Example:</p>

```
<div style="width: 357; height: 50; font-size: 30pt; font-family: Arial Black; color: red; Filter: Shadow(Color=#0000FF, Direction=225)">CSS Tutorials</div>
```


It will produce the following result:



Wave Effect

Wave effect is used to give the object a sine wave distortion to make it look wavy. The following parameters can be used in this filter:

Parameter	Description
Add	A value of 1 adds the original image to the waved image, 0 does not.
Freq	The number of waves.
Light	The strength of the light on the wave (from 0 to 100).
phase	At what degree the sine wave should start (from 0 to 100).
strength	The intensity of the wave effect.

Example

```
<p>Image Example:</p>



<p>Text Example:</p>

<div style="width: 357; height: 50; font-size: 30pt; font-family: Arial
Black; color: red; Filter: Wave(Add=0, Freq=1, LightStrength=10,
Phase=20, Strength=20)">CSS Tutorials</div>
```

It will produce the following result:



X-Ray Effect

X-Ray effect grayscales and flattens the color depth. The following parameter is used in this filter:

Parameter	Description
xray	Grayscales and flattens the color depth.

Example

```
<p>Image Example:</p>



<p>Text Example:</p>

<div style="width: 357; height: 50; font-size: 30pt; font-family: Arial
Black; color: red; style="Filter: Xray">CSS Tutorials</div>
```

It will produce the following result:



27. 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:

```
<style tyle="text/css">
<!--
<!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...
  </body>
</html>
-->
</style>
```

Recognized Media Types

The names chosen for CSS media types reflect target devices for which the relevant properties make sense. They give a sense of what device the media type is meant to refer to. Given below is a list of various media types:

Value	Description
all	Suitable for all devices.
aural	Intended for speech synthesizers.
braille	Intended for braille tactile feedback devices.

embossed	Intended for paged braille printers.
handheld	Intended for handheld devices (typically small screen, monochrome, limited bandwidth).
print	Intended for paged, opaque material and for documents viewed on screen in print preview mode. Please consult the section on paged media.
projection	Intended for projected presentations, for example projectors or print to transparencies. Please consult the section on paged media.
screen	Intended primarily for color computer screens.
tty	Intended for media using a fixed-pitch character grid, such as teletypes, terminals, or portable devices with limited display capabilities.
tv	Intended for television-type devices.

NOTE: Media type names are case-insensitive.

28. PAGED MEDIA

Paged media differ from continuous media in that the content of the document is split into one or more discrete pages. Paged media includes paper, transparencies, pages that are displayed on computer screens, etc.

The CSS2 standard introduces some basic pagination control features that let authors help the browser figure out how to best print their documents.

The CSS2 page model specifies how a document is formatted within a rectangular area -- the page box -- that has a finite width and height. These features fall into two groups:

- CSS2 features that define a particular page layout.
- CSS2 features that control the pagination of a document.

Defining Pages: The @page Rule

The CSS2 defines a "page box", a box of finite dimensions, in which content is rendered. The page box is a rectangular region that contains two areas:

- **The page area:** The page area includes the boxes laid out on that page. The edges of the page area act as the initial containing block for layout that occurs between page breaks.
- **The margin area:** It surrounds the page area.

You can specify the dimensions, orientation, margins, etc., of a page box within an @page rule. The dimensions of the page box are set with the 'size' property. The dimensions of the page area are the dimensions of the page box minus the margin area.

For example, the following @page rule sets the page box size to 8.5 x 11 inches and creates '2cm' margin on all sides between the page box edge and the page area:

```
<style type="text/css">
<!--
@page { size:8.5in 11in; margin: 2cm }
-->
</style>
```

You can use the *margin*, *margin-top*, *margin-bottom*, *margin-left*, and *margin-right* properties within the @page rule to set margins for your page.

Finally, the *marks* property is used within the @page rule to create crop and registration marks outside the page box on the target sheet. By default, no marks are printed. You may use one or both of the *crop* and *cross* keywords to create crop marks and registration marks, respectively, on the target print page.

Setting Page Size

The *size* property specifies the size and orientation of a page box. There are four values, which can be used for page size:

- **auto:** The page box will be set to the size and orientation of the target sheet.
- **landscape:** Overrides the target's orientation. The page box is of the same size as the target, and the longer sides are horizontal.
- **portrait:** Overrides the target's orientation. The page box is the same size as the target, and the shorter sides are horizontal.
- **length:** Length values for the 'size' property create an absolute page box. If only one length value is specified, it sets both the width and height of the page box. Percentage values are not allowed for the 'size' property.

In the following example, the outer edges of the page box will align with the target. The percentage value on the 'margin' property is relative to the target size so if the target sheet dimensions are 21.0cm x 29.7cm (i.e., A4), the margins are 2.10cm and 2.97cm.

```
<style type="text/css">
<!--
@page {
    size: auto;    /* auto is the initial value */
    margin: 10%;
}
-->
</style>
```

The following example sets the width of the page box to be 8.5 inches and the height to be 11 inches. The page box in this example requires a target sheet size of 8.5"x11" or larger.


```
<style type="text/css">
<!--
@page {
    size: 8.5in 11in; /* width height */
}
-->
</style>
```

Once you create a named page layout, you can use it in your document by adding the page property to a style that is later applied to an element in your document. For example, this style renders all the tables in your document on landscape pages:

```
<style type="text/css">
<!--
@page { size : portrait }
@page rotated { size : landscape }
table { page : rotated }
-->
</style>
```

Due to the above rule, while printing, if the browser encounters a `<table>` element in your document and the current page layout is the default portrait layout, it starts a new page and prints the table on a landscape page.

Left, Right, and First Pages

When printing double-sided documents, the page boxes on left and right pages should be different. It can be expressed through two CSS pseudo-classes as follows:

```
<style type="text/css">
<!--
@page :left {
    margin-left: 4cm;
    margin-right: 3cm;
}
-->
```

```
@page :right {
    margin-left: 3cm;
    margin-right: 4cm;
}
-->
</style>
```

You can specify the style for the first page of a document with the `:first` pseudo-class:

```
<style type="text/css">
<!--
@page { margin: 2cm } /* All margins set to 2cm */

@page :first {
    margin-top: 10cm    /* Top margin on first page 10cm */
}
-->
</style>
```

Controlling Pagination

Unless you specify otherwise, page breaks occur only when the page format changes or when the content overflows the current page box. To otherwise force or suppress page breaks, use the *page-break-before*, *page-break-after*, and *page-break-inside* properties.

Both *page-break-before* and *page-break-after* accept the *auto*, *always*, *avoid*, *left*, and *right* keywords.

The keyword *auto* is the default, it lets the browser generate page breaks as needed. The keyword *always* forces a page break before or after the element, while *avoid* suppresses a page break immediately before or after the element. The *left* and *right* keywords force one or two page breaks, so that the element is rendered on a left-hand or right-hand page.

Using pagination properties is quite straightforward. Suppose your document has level-1 headers to start new chapters with level-2 headers to denote sections.

You'd like each chapter to start on a new, right-hand page, but you don't want section headers to be split across a page break from the subsequent content. You can achieve this using the following rule:

```
<style type="text/css">
<!--
h1 { page-break-before : right }
h2 { page-break-after : avoid }
-->
</style>
```

Use only the *auto* and *avoid* values with the *page-break-inside* property. If you prefer that your tables are not be broken across pages if possible, you would write the rule as follows:

```
<style type="text/css">
<!--
table { page-break-inside : avoid }
-->
</style>
```

Controlling Widows and Orphans

In typographic lingo, orphans are those lines of a paragraph that are stranded at the bottom of a page due to a page break, while widows are those lines remaining at the top of a page following a page break. Generally, printed pages do not look attractive with single lines of text stranded at the top or bottom. Most printers try to leave at least two or more lines of text at the top or bottom of each page.

- The **orphans** property specifies the minimum number of lines of a paragraph that must be left at the bottom of a page.
- The **widows** property specifies the minimum number of lines of a paragraph that must be left at the top of a page.

Here is an example to create 4 lines at the bottom and 3 lines at the top of each page:

```
<style type="text/css">
```

```
<!--  
@page{orphans:4; widows:2;}  
-->  
</style>
```

29. AURAL MEDIA

A web document can be rendered by a speech synthesizer. CSS2 allows you to attach specific sound style features to specific document elements.

Aural rendering of documents is mainly used by the visually impaired. Some of the situations in which a document can be accessed by means of aural rendering rather than visual rendering are the following.

- Learning to read
- Training
- Web access in vehicles
- Home entertainment
- Industrial documentation
- Medical documentation

When using aural properties, the canvas consists of a three-dimensional physical space (sound surrounds) and a temporal space (one may specify sounds before, during, and after other sounds).

The CSS properties also allow you to vary the quality of synthesized speech (voice type, frequency, inflection, etc.).

Here is an example:

```
<style tyle="text/css">
<!--
h1, h2, h3, h4, h5, h6 {
    voice-family: paul;
    stress: 20;
    richness: 90;
    cue-before: url("ping.au")
}
p.heidi { azimuth: center-left }
p.peter { azimuth: right }
-->
```

</style>

It will direct the speech synthesizer to speak headers in a voice (a kind of audio font) called "paul", on a flat tone, but in a very rich voice. Before speaking the headers, a sound sample will be played from the given URL.

Paragraphs with class 'heidi' will appear to come from front left (if the sound system is capable of spatial audio), and paragraphs of class 'peter' from the right.

Now we will see the various properties related to aural media.

- The **azimuth** property sets, where the sound should come from horizontally.
- The **elevation** property sets, where the sound should come from vertically.
- The **cue-after** specifies a sound to be played after speaking an element's content to delimit it from other.
- The **cue-before** specifies a sound to be played before speaking an element's content to delimit it from other.
- The **cue** is a shorthand for setting cue-before and cue-after.
- The **pause-after** specifies a pause to be observed after speaking an element's content.
- The **pause-before** specifies a pause to be observed before speaking an element's content.
- The **pause** is a shorthand for setting pause-before and pause-after.
- The **pitch** specifies the average pitch (a frequency) of the speaking voice.
- The **pitch-range** specifies the variation in average pitch.
- The **play-during** specifies a sound to be played as a background, while an element's content is spoken.
- The **richness** specifies the richness, or brightness, of the speaking voice.
- The **speak** specifies whether the text will be rendered aurally and if so, in what manner.
- The **speak-numeral** controls how numerals are spoken.
- The **speak-punctuation** specifies how punctuation is spoken.
- The **speech-rate** specifies the speaking rate.

- The **stress** specifies the height of "local peaks" in the intonation contour of a voice.
- The **voice-family** specifies the prioritized list of voice family names.
- The **volume** refers to the median volume of the voice.

The azimuth Property

The azimuth property sets where the sound should come from horizontally. The possible values are listed below:

- **angle:** Position is described in terms of an angle within the range -*360deg* to *360deg*. The value *0deg* means directly ahead in the center of the sound stage. *90deg* is to the right, *180deg* behind, and *270deg* (or, equivalently and more conveniently, *-90deg*) to the left.
- **left-side:** Same as '*270deg*'. With 'behind', '*270deg*'.
- **far-left:** Same as '*300deg*'. With 'behind', '*240deg*'.
- **left:** Same as '*320deg*'. With 'behind', '*220deg*'.
- **center-left:** Same as '*340deg*'. With 'behind', '*200deg*'.
- **center:** Same as '*0deg*'. With 'behind', '*180deg*'.
- **center-right:** Same as '*20deg*'. With 'behind', '*160deg*'.
- **right:** Same as '*40deg*'. With 'behind', '*140deg*'.
- **far-right:** Same as '*60deg*'. With 'behind', '*120deg*'.
- **right-side:** Same as '*90deg*'. With 'behind', '*90deg*'.
- **leftwards:** Moves the sound to the left and relative to the current angle. More precisely, subtracts 20 degrees.
- **rightwards:** Moves the sound to the right, relative to the current angle. More precisely, adds 20 degrees.

Here is an example:

```
<style tyle="text/css">
<!--
h1 { azimuth: 30deg }
td.a { azimuth: far-right } /* 60deg */
#12 { azimuth: behind far-right } /* 120deg */
```

```
p.comment { azimuth: behind }      /* 180deg */
-->
</style>
```

The elevation Property

The elevation property sets where the sound should come from vertically. The possible values are as follows:

- **angle:** Specifies the elevation as an angle, between *-90deg* and *90deg*. *0deg* means on the forward horizon, which loosely means level with the listener. *90deg* means directly overhead and *-90deg* means directly below.
- **below:** Same as *'-90deg'*.
- **level:** Same as *'0deg'*.
- **above:** Same as *'90deg'*.
- **higher:** Adds 10 degrees to the current elevation.
- **lower:** Subtracts 10 degrees from the current elevation.

Here is an example:

```
<style tyle="text/css">
<!--
h1 { elevation: above }
tr.a { elevation: 60deg }
tr.b { elevation: 30deg }
tr.c { elevation: level }
-->
</style>
```

The cue-after Property

The cue-after property specifies a sound to be played after speaking an element's content to delimit it from other. The possible values include:

- **url:** The URL of a sound file to be played.
- **none:** Nothing has to be played.

Here is an example:

```
<style tyle="text/css">
<!--
a {cue-after: url("dong.wav");}
h1 {cue-after: url("pop.au"); }
-->
</style>
```

The cue-before Property

This property specifies a sound to be played before speaking an element's content to delimit it from other. The possible values include:

- **url:** The URL of a sound file to be played.
- **none:** Nothing has to be played.

Here is an example:

```
<style tyle="text/css">
<!--
a {cue-before: url("bell.aiff");}
h1 {cue-before: url("pop.au"); }
-->
</style>
```

The cue Property

The cue property is a shorthand for setting *cue-before* and *cue-after*. If two values are given, the first value is *cue-before* and the second is *cue-after*. If only one value is given, it applies to both properties.

For example, the following two rules are equivalent:

```
<style tyle="text/css">
<!--
h1 {cue-before: url("pop.au"); cue-after: url("pop.au")}
h1 {cue: url("pop.au")}
-->
```

```
-->
</style>
```

The pause-after Property

This property specifies a pause to be observed after speaking an element's content. The possible values are:

- **time:** Expresses the pause in absolute time units (seconds and milliseconds).
- **percentage:** Refers to the inverse of the value of the *speech-rate* property. For example, if the speech-rate is 120 words per minute (i.e., a word takes half a second, or 500ms), then a *pause-after* of 100% means a pause of 500 ms and a *pause-after* of 20% means 100ms.

The pause-before Property

This property specifies a pause to be observed before speaking an element's content. The possible values are:

- **time:** Expresses the pause in absolute time units (seconds and milliseconds).
- **percentage:** Refers to the inverse of the value of the *speech-rate* property. For example, if the speech-rate is 120 words per minute (i.e., a word takes half a second, or 500ms), then a *pause-before* of 100% means a pause of 500 ms and a *pause-before* of 20% means 100ms.

The pause Property

This property is a shorthand for setting *pause-before* and *pause-after*. If two values are given, the first value is *pause-before* and the second is *pause-after*.

Here is an example:

```
<style tyle="text/css">
<!--
/* pause-before: 20ms; pause-after: 20ms */
h1 { pause : 20ms }
/* pause-before: 30ms; pause-after: 40ms */
h2{ pause : 30ms 40ms }
```

```
/* pause-before: ?; pause-after: 10ms */
h3 { pause-after : 10ms }
-->
</style>
```

The pitch Property

This property specifies the average pitch (a frequency) of the speaking voice. The average pitch of a voice depends on the voice family. For example, the average pitch for a standard male voice is around 120Hz, but for a female voice, it's around 210Hz. The possible values are:

- **frequency:** Specifies the average pitch of the speaking voice in hertz (Hz).
- **x-low, low, medium, high, x-high:** These values do not map to absolute frequencies, since these values depend on the voice family.

The pitch-range Property

This property specifies variation in average pitch. The possible values are:

- **number:** A value between '0' and '100'. A pitch range of '0' produces a flat, monotonic voice. A pitch range of 50 produces normal inflection. Pitch ranges greater than 50 produce animated voices.

The play-during Property

This property specifies a sound to be played as a background while an element's content is spoken. Possible values could be any of the followings:

- **URI:** The sound designated by this <uri> is played as a background while the element's content is spoken.
- **mix:** When present, this keyword means that the sound inherited from the parent element's *play-during* property continues to play and the sound designated by the *uri* is mixed with it. If *mix* is not specified, the element's background sound replaces the parent's.
- **repeat:** When present, this keyword means that the sound will repeat if it is too short to fill the entire duration of the element. Otherwise, the sound plays once and then stops.
- **auto:** The sound of the parent element continues to play.

- **none:** This keyword means that there is silence.

Here is an example:

```
<style tyle="text/css">
<!--
blockquote.sad { play-during: url("violins.aiff") }
blockquote q   { play-during: url("harp.wav") mix }
span.quiet     { play-during: none }
-->
</style>
```

The richness Property

This property specifies the richness or brightness of the speaking voice. The possible values are:

- **number:** A value between '0' and '100'. The higher the value, the more the voice will carry. A lower value will produce a soft, mellifluous voice.

The speak Property

This property specifies whether the text will be rendered aurally and if so, in what manner. The possible values are:

- **none:** Suppresses aural rendering so that the element requires no time to render.
- **normal:** Uses language-dependent pronunciation rules for rendering an element and its children.
- **spell-out:** Spells the text one letter at a time.

Note the difference between an element whose 'volume' property has a value of 'silent' and an element whose 'speak' property has the value 'none'. The former takes up the same time as if it had been spoken, including any pause before and after the element, but no sound is generated. The latter requires no time and is not rendered.

The speak-numeral Property

This property controls how numerals are spoken. The possible values are:

- **digits:** Speak the numeral as individual digits. Thus, "237" is spoken "Two Three Seven".
- **continuous:** Speak the numeral as a full number. Thus, "237" is spoken "Two hundred thirty seven". Word representations are language-dependent.

The speak-punctuation Property

This property specifies how punctuation is spoken. The possible values are:

- **code:** Punctuation such as semicolons, braces, and so on are to be spoken literally.
- **none:** Punctuation is not to be spoken, but instead rendered naturally as various pauses.

The speech-rate Property

This property specifies the speaking rate. Note that both absolute and relative keyword values are allowed. The possible values are:

- **number:** Specifies the speaking rate in words per minute.
- **x-slow:** Same as 80 words per minute.
- **slow:** Same as 120 words per minute .
- **medium:** Same as 180 - 200 words per minute.
- **fast:** Same as 300 words per minute.
- **x-fast:** Same as 500 words per minute.
- **faster:** Adds 40 words per minute to the current speech rate.
- **slower:** Subtracts 40 words per minutes from the current speech rate.

The stress Property

This property specifies the height of "local peaks" in the intonation contour of a voice. English is a stressed language, and different parts of a sentence are assigned primary, secondary, or tertiary stress. The possible values are:

- **number:** A value between '0' and '100'. The meaning of values depends on the language being spoken. For example, a level of '50' for a standard, English-speaking male voice (average pitch = 122Hz), speaking with normal intonation and emphasis would have a different meaning than '50' for an Italian voice.

The voice-family Property

The value is a comma-separated, prioritized list of voice family names. It can have the following values:

- **generic-voice:** Values are voice families. Possible values are 'male', 'female', and 'child'.
- **specific-voice:** Values are specific instances (e.g., comedian, trinoids, carlos, lani).

Here is an example:

```
<style tyle="text/css">
<!--
h1 { voice-family: announcer, male }
p.part.romeo { voice-family: romeo, male }
p.part.juliet { voice-family: juliet, female }
-->
</style>
```

The volume Property

Volume refers to the median volume of the voice. It can have the following values:

- **numbers:** Any number between '0' and '100'. '0' represents the minimum audible volume level and 100 corresponds to the maximum comfortable level.
- **percentage:** These values are calculated relative to the inherited value, and are then clipped to the range '0' to '100'.
- **silent:** No sound at all. The value '0' does not mean the same as 'silent'.
- **x-soft:** Same as '0'.
- **soft:** Same as '25'.

- **medium:** Same as '50'.
- **loud:** Same as '75'.
- **x-loud:** Same as '100'.

Here is an example:

```
<style tyle="text/css">
<!--
P.goat { volume: x-soft }
-->
</style>
```

Paragraphs with class **goat** will be very soft.

30. PRINTING

You can use CSS to change the appearance of your web page when it is printed on a paper. You can specify one font for the screen version and another for the print version.

You have seen @media rule in the previous chapters. This rule allows you to specify different styles for different media. So, you can define different rules for a screen and a printer.

The following example specifies different font families for screen and printer. The next CSS uses the same font size for both screen as well as printer.

```
<style tyle="text/css">
<!--
@media screen
{
p.bodyText {font-family:verdana, arial, sans-serif;}
}

@media print
{
p.bodyText {font-family:georgia, times, serif;}
}

@media screen, print
{
p.bodyText {font-size:10pt}
}
-->
</style>
```


If you are defining your style sheet in a separate file, then you can also use the media attribute when linking to an external style sheet:

```
<link rel="stylesheet" type="text/css"  
      media="print" href="mystyle.css">
```

31. LAYOUTS

Hope you are very comfortable with HTML tables and you are efficient in designing page layouts using HTML Tables. But you know CSS too provides plenty of controls for positioning elements in a document. Since CSS is *the wave of the future*, why not learn and use CSS instead of tables for page layout purposes?

The following list collects a few pros and cons of both the technologies:

- Most browsers support tables, while CSS support is being slowly adopted.
- Tables are more forgiving when the browser window size changes - morphing their content and wrapping to accommodate the changes accordingly. CSS positioning tends to be exact and fairly inflexible.
- Tables are much easier to learn and manipulate than CSS rules.

But each of these arguments can be reversed:

- CSS is pivotal to the future of Web documents and will be supported by most browsers.
- CSS is more exact than tables, allowing your document to be viewed as you intended, regardless of the browser window.
- Keeping track of nested tables can be a real pain. CSS rules tend to be well organized, easily read, and easily changed.

Finally, we would suggest you to use whichever technology makes sense to you and use what you know or what presents your documents in the best way.

CSS also provides *table-layout* property to make your tables load much faster. Following is an example:

```
<table style="table-layout:fixed;width:600px;">
  <tr height="30">
    <td width="150">CSS table layout cell 1</td>
    <td width="200">CSS table layout cell 2</td>
    <td width="250">CSS table layout cell 3</td>
  </tr>
</table>
```

You will notice the benefits more on large tables. With traditional HTML, the browser had to calculate every cell before finally rendering the table. When you set the table-layout algorithm to *fixed*, however, it only needs to look at the first row before rendering the whole table. It means your table will need to have fixed column widths and row heights.

Sample Column Layout

Here are the steps to create a simple Column Layout using CSS:

Set the margin and padding of the complete document as follows:

```
<style tyle="text/css">
<!--
body {
    margin:9px 9px 0 9px;
    padding:0;
    background:#FFF;
}
-->
</style>
```

Now, we will define a column with yellow color and later, we will attach this rule to a <div>:

```
<style tyle="text/css">
<!--
#level0 {
    background:#FC0;
}
-->
</style>
```

Up to this point, we will have a document with yellow body, so let us now define another division inside level0:

```
<style tyle="text/css">
<!--
```

```
#level1 {
    margin-left:143px;
    padding-left:9px;
    background:#FFF;
}
-->
</style>
```

Now, we will nest one more division inside level1, and we will change just the background color:

```
<style tyle="text/css">
<!--
#level2 {
    background:#FFF3AC;
}
-->
</style>
```

Finally, we will use the same technique, nest a level3 division inside level2 to get the visual layout for the right column:

```
<style tyle="text/css">
<!--
#level3 {
    margin-right:143px;
    padding-right:9px;
    background:#FFF;
}
#main {
    background:#CCC;
}
-->
</style>
```

Complete the source code as follows:

```
<style tyle="text/css">
<!--
body {
    margin:9px 9px 0 9px;
    padding:0;
    background:#FFF;}
#level0 {
    background:#FC0;}
#level1 {
    margin-left:143px;
    padding-left:9px;
    background:#FFF;}
#level2 {
    background:#FFF3AC;}
#level3 {
    margin-right:143px;
    padding-right:9px;
    background:#FFF;}
#main {
    background:#CCC;}
-->
</style>
<body>
    <div id="level0">
        <div id="level1">
            <div id="level2">
                <div id="level3">
                    <div id="main">
                        Final Content goes here...
                    </div>
```

```
        </div>  
    </div>  
</div>  
</div>  
</body>
```

Similarly, you can add a top navigation bar or an ad bar at the top of the page.

32. VALIDATIONS



Validation is the process of checking something against a rule. When you are a beginner, it is very common that you will commit many mistakes in writing your CSS rules. How you will make sure whatever you have written is 100% accurate and up to the W3 quality standards?

If you use CSS, your code needs to be correct. Improper code may cause unexpected results in how your page looks or functions.

But if you want to validate your CSS style sheet embedded in an (X)HTML document, you should first check that the (X)HTML you use is valid.

Tool to check the validity of (X)HTML document: **Validate (X)HTML document.**

You can use the following tools to check the validity of your CSS.

	W3C CSS Validator (World Wide Web Consortium) This validator checks your CSS by either file upload, direct input, or using URI - one page at a time. This validator helps you to locate all the errors in your CSS.
	The WDG CSS check validator, lets you validate your CSS by direct input, file upload, and using URI. Errors will be listed by line and column numbers if you have any. Errors usually come with links to explain the reason of error.

A CSS validator checks your Cascading Style Sheets to make sure they comply with the CSS standards set by the W3 Consortium. There are a few validators which will also tell you which CSS features are supported by which browsers (since not all browsers are equal in their CSS implementation).

Why Validate Your HTML Code?

There are a number of reasons why you should validate your code. But major ones are:

- It helps cross-browser, cross-platform, and future compatibility.
- A good quality website increases search engine visibility.

- Professionalism: As a web developer, your code should not raise errors while it is being used by visitors.

33. CSS2 REFERENCE GUIDE

This is a complete reference guide for web developers where we have listed all the CSS properties defined in the World Wide Web Consortium's Recommended Specification for Cascading Style Sheets, Level 2.

- [Aural](#)
- [Background](#)
- [Border](#)
- [Classification](#)
- [Dimension](#)
- [Font](#)
- [Generated Content](#)
- [List and Marker](#)
- [Margin](#)
- [Outlines](#)
- [Padding](#)
- [Positioning](#)
- [Table](#)
- [Text](#)
- [Print](#)
- [Pseudo-classes](#)
- [Pseudo-elements](#)

Click any property to see its description with examples:

Property	Description
azimuth	Describes the position of a sound source along the horizontal axis of the listener's environment.
background	Composite property for the following properties: background-attachment background-color background-image background-position background-repeat
background-	Determines if the background image is fixed in the window

<u>attachment</u>	or scrolls as the document scrolls
<u>background-color</u>	Sets the background color of an element
<u>background-image</u>	Sets the background image of an element
<u>background-position</u>	Sets the initial position of the element's background image, if specified; values normally are paired to provide x, y positions; default position is 0% 0%.
<u>background-repeat</u>	Determines how the background image is repeated (tiled) across an element
<u>border</u>	Sets all four of an element's borders; value is one or more of a color, a value for <i>border-width</i> , and a value for <i>border-style</i>
<u>border-bottom</u>	Sets an element's bottom border; value is one or more of a color, a value for <i>border-bottom-width</i> , and a value for <i>border-style</i>
<u>border-bottom-width</u>	Sets the thickness of an element's bottom border.
<u>border-collapse</u>	Sets the table border rendering algorithm
<u>border-color</u>	Sets the color of all four of an element's borders; default is the color of the element
<u>border-left-color</u>	Sets the color of an element's left borders; default is the color of the element
<u>border-right-color</u>	Sets the color of an element's right borders; default is the color of the element
<u>border-top-color</u>	Sets the color of an element's top borders; default is the color of the element
<u>border-bottom-color</u>	Sets the color of an element's bottom borders; default is the color of the element
<u>border-left</u>	Sets an element's left border; value is one or more of a

	color, a value for <i>border-left-width</i> , and a value for <i>border-style</i> .
<u>border-left-width</u>	Sets the thickness of an element's left border
<u>border-right</u>	Sets an element's right border; value is one or more of a color, a value for <i>border-right-width</i> , and a value for <i>border-style</i> .
<u>border-right-width</u>	Sets the thickness of an element's right border
<u>border-spacing</u>	With separate borders set the spacing between borders. One value sets vertical and horizontal spacing and two values sets horizontal and vertical spacing respectively.
<u>border-style</u>	Sets the style of all four of an element's borders
<u>border-top</u>	Sets an element's top border; value is one or more of a color, a value for <i>border-top-width</i> , and a value for <i>border-style</i>
<u>border-top-width</u>	Sets the thickness of an element's top border.
<u>border-width</u>	Sets the thickness of all four of an element's borders
<u>bottom</u>	Used with the <i>position</i> property to place the bottom edge of an element
<u>caption-side</u>	Sets the position for a table caption
<u>clear</u>	Sets which margins of an element must not be adjacent to a floating element; the element is moved down until that margin is clear
<u>clip</u>	Sets the clipping mask for an element
<u>color</u>	Sets the color of an element
<u>content</u>	Inserts generated content around an element.
<u>counter-increment</u>	Increments a counter by 1; value is a list of counter names, with each name optionally followed by a value by which it is incremented.

<u>counter-reset</u>	Resets a counter to zero; value is a list of counter names, with each name optionally followed by a value to which it is reset.
<u>cue-after</u>	Plays the designated sound after an element is spoken
<u>cue-before</u>	Plays the designated sound before an element is spoken
<u>cursor</u>	Defines the shape of the cursor
<u>direction</u>	Defines direction of the flow of an element content
<u>display</u>	Controls how an element is displayed
<u>elevation</u>	Sets the height at which a sound is played
<u>empty-cells</u>	With separate borders, hides empty cells in a table
<u>float</u>	Determines if an element floats to the left or right, allowing text to wrap around it or be displayed inline
<u>font</u>	Sets all the font attributes for an element. Value is any of the values for: font-style font-variant font-weight font-size line-height font-family
<u>font-family</u>	Defines the font for an element, either as a specific font or as one of the generic serif, sans-serif, cursive, fantasy, and monospace.
<u>font-size</u>	Defines the font size
<u>font-size-adjust</u>	Adjusts the current font's aspect ratio
<u>font-stretch</u>	Determines the amount to stretch the current font
<u>font-style</u>	Defines the style of the face, either normal or some type of

	slanted style
<u>font-variant</u>	Defines a font to be in small caps
<u>font-weight</u>	Defines the font weight. if a number is used, it must be a multiple of 100 between 100 and 900; 400 is normal, 700 is the same as the keyword bold
<u>height</u>	Defines the height of an element
<u>left</u>	Used with the <i>position</i> property to place the left edge of an element
<u>letter-spacing</u>	Inserts additional space between text characters
<u>line-height</u>	Sets the distance between adjacent text baselines
<u>list-style</u>	Defines list-related styles using any of the values for: list-style-image liststyle-position list-style-type
<u>list-style-image</u>	Defines an image to be used as a list item's marker, in lieu of the value for: list-style-type
<u>list-style-position</u>	Indents or extends (default) a list item's marker with respect to the item's content
<u>list-style-type</u>	Defines a list item's marker either for unordered lists (circle, disc, or square) or for ordered lists (decimal, lower-alpha, lower-roman, none, upper-alpha, or upper-roman)
<u>margin</u>	Defines all four of an element's margins
<u>margin-bottom</u>	Defines the bottom margin of an element. Default value is 0.
<u>margin-left</u>	Defines the left margin of an element. Default value is 0.
<u>margin-right</u>	Defines the right margin of an element. Default value is 0.

<u>margin-top</u>	Defines the top margin of an element. Default value is 0.
<u>marker-offset</u>	The marker-offset property can be used in bulleted lists for specifying the distance between the nearest border edges of a marker box (or bullet) and its associated principal box.
<u>marks</u>	The marks property is used to set crop marks and cross marks on paged media. This is used with the @page rule.
<u>max-height</u>	max-height property is used to constrain the height of an element.
<u>max-width</u>	max-width property is used to set the maximum width of an element.
<u>min-height</u>	min-height property is used to constrain the height of an element.
<u>min-width</u>	min-width property is used to constrain the width of an element.
<u>orphans</u>	Sets the minimum number of lines allowed in an orphaned paragraph fragment
<u>outline</u>	The outline property is a shorthand property to specify all outline properties.
<u>outline-color</u>	The outline-color property is used to specify the color of the outline. Note that, unlike the border property, outline does not take up extra space and it can be non-rectangular.
<u>outline-color-style</u>	The outline-style property is used to specify the style of the outline. Note that, unlike the border property, outline does not take up extra space and it can be non-rectangular.
<u>outline-width</u>	The outline-color property is used to specify the color of the outline. Note that, unlike the border property, outline does not take up extra space and it can be non-rectangular.
<u>overflow</u>	Determines how overflow content is rendered
<u>padding</u>	Defines all four padding amounts around an element
<u>padding-bottom</u>	Defines the bottom padding of an element. Default value is

	0
<u>padding-left</u>	Defines the left padding of an element. Default value is 0
<u>padding-right</u>	Defines the right padding of an element. Default value is 0
<u>padding-top</u>	Defines the top padding of an element. Default value is 0
<u>page</u>	Associates a named page layout with an element
<u>page-break-after</u>	Forces or suppresses page breaks after an element.
<u>page-break-before</u>	Forces or suppresses page breaks before an element.
<u>page-break-inside</u>	Suppresses page breaks within an element
<u>pause</u>	The <code>pause</code> property is CSS shorthand for specifying shorthand property for specifying pauses in aural media.
<u>pause-after</u>	Pauses a media after speaking an element
<u>pause-before</u>	Pauses a media before speaking an element
<u>pitch</u>	Sets the average pitch of an element's spoken content
<u>pitch-range</u>	Sets the range of the pitch, from 0 (flat) to 100 (broad); default is 50
<u>play-during</u>	If a URL is provided, it is played during an element's spoken content, specifying <i>repeat</i> loops the audio; <i>mix</i> causes it to mix with, rather than replace, other background audio.
<u>position</u>	Sets the positioning model for an element
<u>quotes</u>	Sets the quote symbols used to quote text
<u>richness</u>	Sets the richness of the voice, from 0 (flat) to 100 (mellifluous); default is 50
<u>right</u>	Used with the <i>position</i> property to place the right edge of an element.

<u>size</u>	The size property is used in paged media to specify the size of the page.
<u>speak</u>	Determines how an element's content is spoken.
<u>speak-header</u>	Determines if table headers are spoken once for each row or column or each time a cell is spoken.
<u>speak-numeral</u>	Determines how numerals are spoken
<u>speak-punctuation</u>	Determines if punctuation is spoken or used for inflection
<u>speech-rate</u>	Sets the rate of speech; a number sets the rate in words per minute
<u>stress</u>	Sets the stress of the voice, from 0 (catatonic) to 100 (hyperactive); default is 50.
<u>table-layout</u>	Determines the table-rendering algorithm
<u>text-align</u>	Sets the text alignment style for an element
<u>text-decoration</u>	Defines any decoration for the text; values may be combined
<u>text-indent</u>	Defines the indentation of the first line of text in an element; default is 0
<u>text-shadow</u>	Creates text drop shadows of varying colors and offsets
<u>text-transform</u>	Transforms the text in the element accordingly
<u>top</u>	Used with the <i>position</i> property to place the top edge of an element.
<u>vertical-align</u>	Sets the vertical positioning of an element
<u>visibility</u>	Determines if an element is visible in the document or table
<u>voice-family</u>	Selects a named voice family to speak an element's content
<u>volume</u>	Sets the volume of spoken content; numeric values range from 0 to 100


<u>white-space</u>	Defines how whitespace within an element is handled
<u>widows</u>	Sets the minimum number of lines allowed in a widowed paragraph fragment
<u>width</u>	Defines the width of an element
<u>word-spacing</u>	Inserts additional space between words
<u>z-index</u>	Sets the rendering layer for the current element.

Pseudo-classes & Pseudo-elements





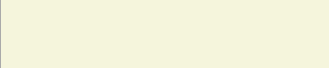










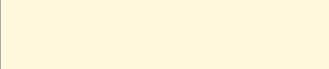




Property	Description
<u>:active</u>	Use this class to add special effect to an activated element
<u>:focus</u>	Use this class to add special effect to an element while the element has focus
<u>:hover</u>	Use this class to add special effect to an element when you mouse over it
<u>:link</u>	Use this class to add special effect to an unvisited link
<u>:visited</u>	Use this class to add special effect to a visited link
<u>:first-child</u>	Use this class to add special effect to an element that is the first child of some other element.
<u>:lang</u>	Use this class to specify a language to use in a specified element
<u>:first-letter</u>	Use this element to add special effect to the first letter of a text
<u>:first-line</u>	Use this element to add special effect to the first line of a text
<u>:before</u>	Use this element to insert some content before an element
<u>:after</u>	Use this element to insert some content after an element


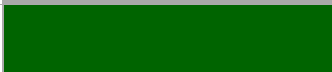


34. COLOR REFERENCES

The following table shows the 16 color names that were introduced in HTML 3.2, to support the 16 colors that 8-bit graphics cards offered. Same set of color can be used in CSS:




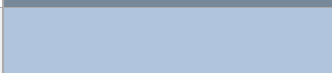

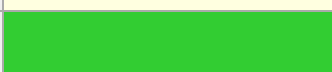
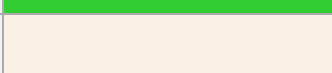
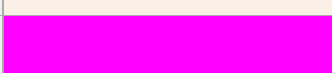





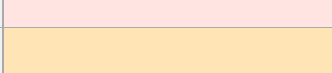









Color Name	Hex Value	Color	Show
aqua	#00ffff		Demo
black	#000000		Demo
blue	#0000ff		Demo
fuchsia	#ff00ff		Demo
green	#008000		Demo
gray	#808080		Demo
lime	#00ff00		Demo
maroon	#800000		Demo
navy	#000080		Demo
olive	#808000		Demo
purple	#800080		Demo
red	#ff0000		Demo
silver	#c0c0c0		Demo
teal	#008080		Demo
white	#ffffff		Demo
yellow	#ffff00		Demo

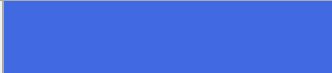











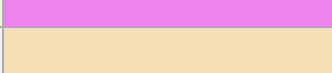
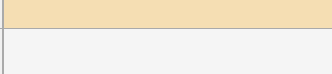


There are other colors, which are not part of HTML or XHTML but they are supported by most of the versions of IE or Netscape. These color names can be used in CSS as well.

Color Name	Hex Value	Color	Show
aliceblue	#f0f8ff		Demo
antiquewhite	#faebd7		Demo
aquamarine	#7fffd4		Demo
azure	#f0ffff		Demo
beige	#f5f5dc		Demo
bisque	#ffe4c4		Demo
blanchedalmond	#ffebcd		Demo
blueviolet	#8a2be2		Demo
brown	#a52a2a		Demo
burlywood	#deb887		Demo
cadetblue	#5f9ea0		Demo
chartreuse	#7fff00		Demo
chocolate	#d2691e		Demo
coral	#ff7f50		Demo
cornflowerblue	#6495ed		Demo
cornsilk	#fff8dc		Demo
crimson	#dc143c		Demo
cyan	#00ffff		Demo
darkblue	#00008b		Demo
darkcyan	#008b8b		Demo

darkgoldenrod	#b8860b		Demo
darkgray	#a9a9a9		Demo
darkgreen	#006400		Demo
darkkhaki	#bdb76b		Demo
darkmagenta	#8b008b		Demo
darkolivegreen	#556b2f		Demo
darkorange	#ff8c00		Demo
darkorchid	#9932cc		Demo
darkred	#8b0000		Demo
darksalmon	#e9967a		Demo
darkseagreen	#8fbc8f		Demo
darkslateblue	#483d8b		Demo
darkslategray	#2f4f4f		Demo
darkturquoise	#00ced1		Demo
darkviolet	#9400d3		Demo
deeppink	#ff1493		Demo
deepskyblue	#00bfff		Demo
dimgray	#696969		Demo
dodgerblue	#1e90ff		Demo
firebrick	#b22222		Demo
floralwhite	#fffaf0		Demo
forestgreen	#228b22		Demo
gainsboro	#dcdcdc		Demo

ghostwhite	#f8f8ff		Demo
gold	#ffd700		Demo
goldenrod	#daa520		Demo
gray	#808080		Demo
greenyellow	#adff2f		Demo
honeydew	#f0fff0		Demo
hotpink	#ff69b4		Demo
indianred	#cd5c5c		Demo
indigo	#4b0082		Demo
ivory	#fffff0		Demo
khaki	#f0e68c		Demo
lavender	#e6e6fa		Demo
lavenderblush	#fff0f5		Demo
lawngreen	#7cfc00		Demo
lemonchiffon	#fffacd		Demo
lightblue	#add8e6		Demo
lightcoral	#f08080		Demo
lightcyan	#e0ffff		Demo
lightgoldenrodyellow	#fafad2		Demo
lightgreen	#90ee90		Demo
lightgrey	#d3d3d3		Demo
lightpink	#ffb6c1		Demo
lightsalmon	#ffa07a		Demo

lightseagreen	#20b2aa		Demo
lightskyblue	#87cefa		Demo
lightslategray	#778899		Demo
lightsteelblue	#b0c4de		Demo
lightyellow	#ffffe0		Demo
limegreen	#32cd32		Demo
linen	#faf0e6		Demo
magenta	#ff00ff		Demo
mediumblue	#0000cd		Demo
mediumorchid	#ba55d3		Demo
mediumpurple	#9370db		Demo
midnightblue	#191970		Demo
mistyrose	#ffe4e1		Demo
moccasin	#ffe4b5		Demo
oldlace	#fdf5e6		Demo
orange	#ffa500		Demo
orchid	#da70d6		Demo
peachpuff	#ffdab9		Demo
peru	#cd853f		Demo
pink	#ffc0cb		Demo
plum	#dda0dd		Demo
purple	#800080		Demo
rosybrown	#bc8f8f		Demo

royalblue	#4169e1		Demo
salmon	#fa8072		Demo
sandybrown	#f4a460		Demo
seagreen	#2e8b57		Demo
sienna	#a0522d		Demo
skyblue	#87ceeb		Demo
slateblue	#6a5acd		Demo
steelblue	#4682b4		Demo
tan	#d2b48c		Demo
thistle	#d8bfd8		Demo
tomato	#ff6347		Demo
violet	#ee82ee		Demo
wheat	#f5deb3		Demo
whitesmoke	#f5f5f5		Demo
yellow	#ffff00		Demo
yellowgreen	#9acd32		Demo