

CASCADING STYLE SHEETS (CSS)

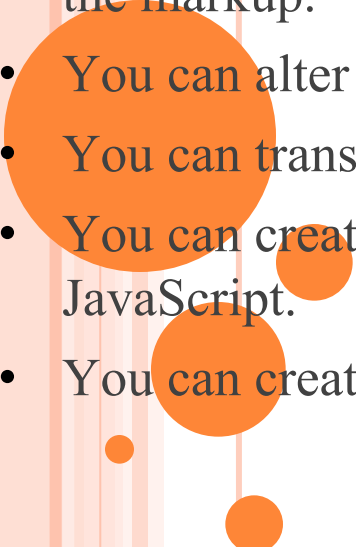


1

- CSS stands for Cascading Style Sheets. CSS is a standard style sheet language used for describing the presentation (i.e. the layout and formatting) of the web pages.
- Prior to CSS, nearly all of the presentational attributes of HTML documents were contained within the HTML markup (specifically inside the HTML tags); all the font colors, background styles, element alignments, borders and sizes had to be explicitly described within the HTML.
- As a result, development of the large websites became a long and expensive process, since the style information were repeatedly added to every single page of the website.
- To solve this problem CSS was introduced in 1996 by the World Wide Web Consortium (W3C), which also maintains its standard.
- **CSS was designed to enable the separation of presentation and content.**
- **Now web designers can move the formatting information of the web pages to a separate style sheet which results in considerably simpler HTML markup, and better maintainability.**
- **CSS3 is the latest version of the CSS specification.**
- **CSS3 adds several new styling features and improvements to enhance the web presentation capabilities.**

WHAT IS CSS?

- ❑ Separate presentation and content.
- ❑ Simple mechanism for adding style to web document.
- ❑ Style is simply set of formatting instructions that can be applied to a piece of text.
- ❑ External style sheets are store with **.css** extension.
- ❑ It primary concern is to separate document content from document presentation.

- **Use of CSS:**
 - You can easily apply same style rules on multiple elements.
 - You can control the presentation of multiple pages of a website with a single style sheet.
 - You can present the same page differently on different devices.
 - You can style dynamic states of elements such as hover, focus, etc. that isn't possible otherwise.
 - You can change the position of an element on a web page without changing the markup.
 - You can alter the display of existing HTML elements.
 - You can transform elements like scale, rotate, skew, etc. in 2D or 3D space.
 - You can create animations and transitions effects without using any JavaScript.
 - You can create print friendly version of your web pages.
- 

- **Advantages of Using CSS**

- The biggest advantage of CSS is that it allows the separation of style and layout from the content of the document.
- CSS Save Lots of Time —
 - CSS gives lots of flexibility to set the style properties of an element.
 - You can write CSS once; and then the same code can be applied to the groups of HTML elements, and can also be reused in multiple HTML pages.
- Easy Maintenance —
 - CSS provides an easy means to update the formatting of the documents, and to maintain the consistency across multiple documents.
 - Because the content of the entire set of web pages can be easily controlled using one or more style sheets.
- Pages Load Faster —
 - CSS enables multiple pages to share the formatting information, which reduces complexity and repetition in the structural contents of the documents.
 - It significantly reduces the file transfer size, which results in a faster page loading.



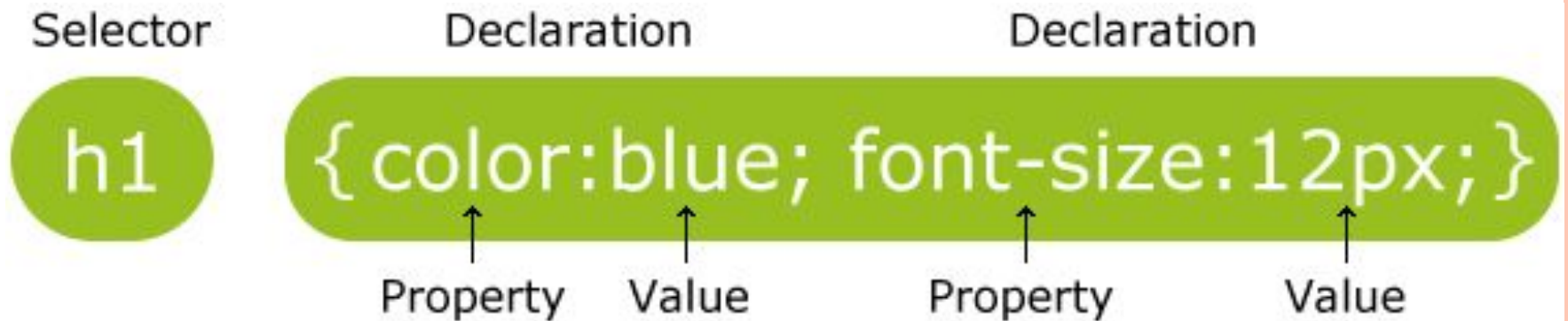
- Superior Styles to HTML —
 - CSS has much wider presentation capabilities than HTML and provide much better control over the layout of your web pages.
 - So you can give far better look to your web pages in comparison to the HTML presentational elements and attributes.
- Multiple Device Compatibility —
 - CSS also allows web pages to be optimized for more than one type of device or media.
 - Using CSS the same HTML document can be presented in different viewing styles for different rendering devices such as desktop, cell phones, etc.



CSS SAVES A LOT OF WORK!

- ❑ CSS defines **HOW** HTML elements are to be displayed.
- ❑ Styles are normally saved in external **.css** files.
- ❑ External style sheets enable you to change the appearance and layout of all the pages in a Web site, just by editing one single file.

CSS SYNTAX



Selector :-is used to create link between the rule and the HTML tag.

BASIC SYNTAX STRUCTURE

```
selector {property: value;.....}  
body {color: yellow }
```

- Values that have multiple words must be enclosed in ("") quotes.

```
P {font-family: "sans serif" }
```

- More than one property to be set then separated by ;

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

- Many selectors are also possible

```
h1,h2,h3,h4,h5 { color: red; }
```

EXAMPLE

<h1>The main heading</h1>
add color – red,
underline,
font – italic

- **If you want to change every h1 tag to have these attributes the old way in html, you would have to go through your code and do it individually.**
- **CSS allows particular style to be attached to a selector which is usually an HTML element.**
- **With css it's possible to do this quit simply by selector, which in this case is h1, followed by list of properties and values:**

**h1 { color : red; text-decoration : underline;
font : italic 1cm Times; }**

CSS EXAMPLE

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

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

- [Css_example1.html](#)

- [Output](#)

CSS COMMENTS

- ▣ **with `"/**`", and ends with `*/`, like this:**
- ▣ **`/*This is a comment*/`
`p`
`{`
`text-align:center;`
`/*This is another comment*/`
`color:black;`
`font-family:arial;`
`}`**

Including CSS code in HTML

- **There are three ways of inserting a style sheet:**
 - 1. External style sheet**
 - 2. Internal style sheet**
 - 3. Inline style**

EXTERNAL STYLE SHEET

- ❑ An external style sheet is ideal when the style is applied to many pages.
- ❑ With an external style sheet, you can change the look of an entire Web site by changing one file.
- ❑ Each page must link to the style sheet using the `<link>` tag. The `<link>` tag goes inside the head section:

`<head>`

`<link rel="stylesheet"`

`type="text/css"href="mystyle.css" />`

`</head>`

EXTERNAL STYLE SHEET

- ❑ **Rel-** tells browser type of link you are using.
- ❑ **An external style sheet can be written in any text editor.**
- ❑ **The file should not contain any html tags.**
- ❑ **Your style sheet should be saved with a .css extension.**

- The type attribute specifies the MIME type of the linked document/resource.
- This attribute is only used if the href attribute is set.
- A very common MIME type for <link> is "text/css", which specifies a style sheet.

```
hr {color:sienna;}
```

```
p {margin-left:20px;}
```

```
body
```

```
{background-image:url("images/back  
0.gif");}
```



EXTERNAL STYLE SHEET

style1.css

External_css.html □ output

Internal Style sheet/Embedded Style sheet

The `<style>` tag is used to define style information for an HTML document.

Inside the `<style>` element you specify how HTML elements should render in a browser.

Each HTML document can contain multiple `<style>` tags.



- ❑ An internal style sheet should be used when a single document has a unique style.
- ❑ Define internal styles in the head section of an HTML page, by using the <style> tag.

<head>

<style type="text/css">

hr {color:sienna;}

p {margin-left:20px;}

body {background-image:url("images/back40.gif");}

</style>

</head>

INTERNAL STYLE SHEET

- ▣ Internal_css.html
- ▣ Output

INLINE STYLE SHEET

An inline style can be used if a unique style is to be applied to one single occurrence of an element.

To use inline styles, use the style attribute in the relevant tag.

The style attribute can contain any CSS property.

eg.

```
<p style="color:blue;margin-left:20px;">This is a  
paragraph.</p>
```

```
<html>
```

```
<body style="background-color:yellow;">
```

```
<h2 style="background-color:red;">This is a heading</h2>
```

```
<p style="background-color:green;">This is a paragraph.</p>
```

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

Inline Style Sheets:

- Inline_css.html
- output

Multiple Style Sheets

- ❑ If some properties have been set for the same selector in different style sheets, the values will be inherited from the more specific style sheet.
- ❑ For example, an external style sheet has these properties for the h3 selector:
- ❑ **h3**
{
color:red;
text-align:left;
font-size:8pt;
}

Multiple Style Sheets

- And an internal style sheet has these properties for the h3 selector:

```
h3  
{  
text-align:right;  
font-size:20pt;  
}
```

- If the page with the internal style sheet also links to the external style sheet the properties for h3 will be:
- color:red;**
text-align:right;
font-size:20pt;
- The color is inherited from the external style sheet and the text-alignment and the font-size is replaced by the internal style sheet.

Multiple Styles Will Cascade into One Styles can be specified:


- ▣ **inside an HTML element**
- ▣ **inside the head section of an HTML page**
- ▣ **in an external CSS file**

Cascading order

- **What style will be used when there is more than one style specified for an HTML element?**
 - **All the styles will "cascade" into a new "virtual" style sheet by the following rules, where number **four has the highest priority:****
- 1. Browser default**
 - 2. External style sheet**
 - 3. Internal style sheet (in the head section)**
 - 4. Inline style (inside an HTML element)**

Cascading order

- ❑ **So, an inline style (inside an HTML element) has the highest priority, which means that it will override a style defined inside the <head> tag, or in an external style sheet, or in a browser (a default value).**
- ❑ **Note: If the link to the external style sheet is placed after the internal style sheet in HTML <head>, the external style sheet will override the internal style sheet!**

- CSS Selectors:
 - A CSS selector selects the HTML element(s) you want to style.
 - **CSS Selectors:**
 - CSS selectors are used to "find" (or select) the HTML elements you want to style.
 - We can divide CSS selectors into five categories:
 - Simple selectors (select elements based on name, id, class)
 - Combinator selectors (select elements based on a specific relationship between them)
 - Pseudo-class selectors (select elements based on a certain state)
 - Pseudo-elements selectors (select and style a part of an element)
 - Attribute selectors (select elements based on an attribute or attribute value)
- 

- **The CSS element Selector**
- The element selector selects HTML elements based on the element name.
- **Example**
- Here, all <p> elements on the page will be center-aligned, with a red text color:

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



- **The CSS id Selector**
- The id selector uses the id attribute of an HTML element to select a specific element.
- The id of an element is unique within a page, so the id selector is used to select one unique element!
- To select an element with a specific id, write a hash (#) character, followed by the id of the element.
- **Example**
- The CSS rule below will be applied to the HTML element with id="para1":

```
#para1 {  
  text-align: center;  
  color: red;  
}
```



- **The CSS class Selector**

- The class selector selects HTML elements with a specific class attribute.
- To select elements with a specific class, write a period (.) character, followed by the class name.

- **Example**

- In this example all HTML elements with `class="center"` will be red and center-aligned:

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



- `p.center {
 text-align: center;
 color: red;
}`
- `<p class="center">`This paragraph refers to two classes.`</p>`



- **The CSS Universal Selector**

- The universal selector (*) selects all HTML elements on the page.

- **Example**

- The CSS rule below will affect every HTML element on the page:

```
* {  
  text-align: center;  
  color: blue;  
}
```



- **The CSS Grouping Selector**

- The grouping selector selects all the HTML elements with the same style definitions.
- Look at the following CSS code (the h1, h2, and p elements have the same style definitions):

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

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

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



All CSS Simple Selectors

Selector	Example	Example description
<u>#id</u>	#firstname	Selects the element with id="firstname"
<u>.class</u>	.intro	Selects all elements with class="intro"
<u>element.class</u>	p.intro	Selects only <p> elements with class="intro"
<u>*</u>	*	Selects all elements
<u>element</u>	p	Selects all <p> elements
<u>element,element,..</u>	div, p	Selects all <div> elements and all <p> elements



Grouping Selectors

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

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

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

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

you can group the selectors,

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



- **CSS selector:**
- A CSS selector selects the HTML element(s) you want to style.
- CSS selectors are used to "find" (or select) the HTML elements you want to style.
- A CSS selector is a pattern to match the elements on a web page.
- The style rules associated with that selector will be applied to the elements that match the selector pattern.
- **Selectors determines elements on which rules are to be applied.**
- **In CSS, selectors are used to declare which part of the markup a style applies to, a kind of match expression.**



The CSS element Type Selector:

The element selector selects HTML elements based on the element name.

An element type selector matches all instance of the element in the document with the corresponding element type name.

Example

Here, all `<p>` elements on the page will be center-aligned, with a red text color:

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

- [Example](#)



```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Example of CSS element type selector</title>
  <style>
    h1 {
      color: red;
    }
    p {
      color: blue;
    }
  </style>
</head>
<body>
  <h1>This is heading</h1>
  <p>This is a paragraph.</p>
  <p>This is a paragraph.</p>
</body>
</html>
```



THE ID SELECTOR

- ❑ The id selector is used to specify a style for a single, unique element.
- ❑ The id selector uses the id attribute of the HTML element, and is defined with a "#".
- ❑ The style rule below will be applied to the element with id="para1":
- ❑ The CSS element Selector
- ❑ The element selector selects HTML elements based on the element name.
- ❑ [Id_selector1.html](#)


```
<html>
```

```
<head>
```

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

```
#para1
```

```
{
```

```
text-align:center;
```

```
color:red;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<p id="para1">Hello World!</p>
```

```
<p>This paragraph is not affected by the style.</p>
```

```
</body>
```

```
</html>
```



THE CLASS SELECTOR

- ❑ The class selector is used to specify a style for a group of elements.
- ❑ The class selectors can be used to select any HTML element that has a class attribute.
- ❑ All the elements having that class will be formatted according to the defined rule.
- ❑ The class selector is defined with a period sign (.) immediately followed by the class value.
- ❑ Unlike the id selector, the class selector is most often used on several elements.
- ❑ This allows you to set a particular style for many HTML elements with the same class.
- ❑ In the example below, all HTML elements with `class="center"` will be center-aligned:
- ❑ [Class_selector.html](#) ❑ [output](#)

- **The CSS class Selector**

- The class selector selects HTML elements with a specific class attribute.
- To select elements with a specific class, write a period (.) character, followed by the class name.

```
.center
```

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

```
<p class="center">This is Paragraph</P>
```

- **<h1 class="Center">**



- In this example only `<p>` elements with `class="center"` will be red and center-aligned:

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

```
<p class="center">This paragraph refers to two  
  classes.</p>
```



THE CLASS SELECTOR

- ❑ You can also specify that only specific HTML elements should be affected by a class.
- ❑ In the example below, all p elements with `class="center"` will be center-aligned:
- ❑ [Class_selector1.html](#)

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Example of CSS class selector</title>
  <style>
    .blue {
      color: #0000ff;
    }
  </style>
</head>
<body>
  <h1 class="blue">This is a heading</h1>
  <p class="blue">This is a paragraph.</p>
  <p>This is another paragraph.</p>
</body>
</html>
```



- ❑ **Universal selector(The CSS Universal Selector) - ***
- ❑ **This selector apples the style to any element.**
- ❑ The universal selector (*) selects all HTML elements on the page.
- ❑ The universal selector, denoted by an asterisk (*), matches every single element on the page.
- ❑ The universal selector may be omitted if other conditions exist on the element.
- ❑ This selector is often used to remove the default margins and paddings from the elements for quick testing purpose.

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

❑ *** { color: red; } </style>**

❑ **Example**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Example of CSS universal selector</title>
  <style>
    * {
      margin: 0;
      padding: 0;
      color:red;
    }
  </style>
</head>
<body>
  <h1>This is heading</h1>
  <p>This is a paragraph.</p>
</body>
</html>
```



- ❑ **Element class selector - *h1.myStyle***
- ❑ The class selector can be applied to elements so that any element of a certain type that has the appropriate class attribute can have certain styles applied.
- ❑ **p.boldText { font-weight: bold; }**
- ❑ Example output

CLASSES (using external css)

- It possible that sometime you will need to make several styles for the same HTML elements.

e.g. : paragraph

[Class_css.html](#) □ [output](#)
[style2.txt](#)

ID(using external css)

- ▣ **There is the id selector which will apply a specific style to an identified element.**

[Id_css.html](#)

[style3.css](#)

Nesting:

If the CSS is structured well, there shouldn't be a need to use many class or ID selectors.

This is because you can specify properties to selectors **within** other selectors.

For example:

```
#top { background-color: #ccc; padding: 1em }
```

```
#top h1 { color: #ff0; }
```

```
#top p { color: red; font-weight: bold; }
```

```
<div id="top"> <h1>
```

```
Chocolate curry</h1>
```

```
<p>This is my recipe for making curry purely with chocolate</p>
```

```
<p>Mmm mm mmmmm</p>
```

```
</div>
```



Attribute Selector:

```
input[type="text"] {  
  width: 150px;  
  display: block;  
  margin-bottom: 10px;  
  background-color: yellow;  
}
```

```
input[type="button"] {  
  width: 120px;  
  margin-left: 35px;  
  display: block;
```



- **CSS Combinators**

- A CSS selector can contain more than one simple selector.
- Between the simple selectors, we can include a combinator.
- There are four different combinators in CSS:
 - descendant selector (space)
 - child selector (>)
 - adjacent sibling selector (+)
 - general sibling selector (~)



- **Descendant Selector**

- The descendant selector matches all elements that are descendants of a specified element.
- The following example selects all `<p>` elements inside `<div>` elements:

- Example

```
div p {  
    background-color: yellow;  
}
```



- **Child Selector (>)**

- The child selector selects all elements that are the children of a specified element.
- The following example selects all <p> elements that are children of a <div> element:

- Example

```
div > p {  
    background-color: yellow;  
}
```



- **Adjacent Sibling Selector (+)**
 - The adjacent sibling selector is used to select an element that is directly after another specific element.
 - Sibling elements must have the same parent element, and "adjacent" means "immediately following".
 - The following example selects the first <p> element that are placed immediately after <div> elements:
 - Example
- ```
div + p {
 background-color: yellow;
}
```



- **General Sibling Selector (~)**

- The general sibling selector selects all elements that are next siblings of a specified element.
- The following example selects all <p> elements that are next siblings of <div> elements:

- Example

```
div ~ p {
 background-color: yellow;
}
```



```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
div p {
```

```
 background-color: yellow;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<h2>Descendant Selector</h2>
```

```
<p>The descendant selector matches all elements that are
descendants of a specified element.</p>
```

```
<div>
```

```
 <p>Paragraph 1 in the div.</p>
```

```
 <p>Paragraph 2 in the div.</p>
```

```
 <section><p>Paragraph 3 in the div.</p></section>
```

```
</div>
```

```
<p>Paragraph 4. Not in a div.</p>
```

```
<p>Paragraph 5. Not in a div.</p>
```



```
<html>
<head>
<style>
div > p {
 background-color: yellow;
}
</style>
</head>
<body>
<h2>Child Selector</h2>
<p>The child selector (>) selects all elements that are the children
of a specified element.</p>
<div>
 <p>Paragraph 1 in the div.</p>
 <p>Paragraph 2 in the div.</p>
 <section>
 <!-- not Child but Descendant -->
 <p>Paragraph 3 in the div (inside a section element).</p>
 </section>
 <p>Paragraph 4 in the div </p>
```



## Child Selector

The child selector (`>`) selects all elements that are the children of a specified element.

Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3 in the div (inside a section element).

Paragraph 4 in the div.

Paragraph 5. Not in a div.

Paragraph 6. Not in a div.



- **CSS Attribute Selectors**

- The [attribute] selector is used to select elements with a specified attribute.
- The following example selects all <a> elements with a target attribute:
- CSS [attribute="value"] Selector
- The [attribute="value"] selector is used to select elements with a specified attribute and value.
- The following example selects all <a> elements with a target="\_blank" attribute:
- Example

```
a[target="_blank"] {
 background-color: yellow;
}
```



- **CSS [attribute~="value"] Selector**

- The [attribute~="value"] selector is used to select elements with an attribute value containing a specified word.

- The following example selects all elements with a title attribute that contains a space-separated list of words, one of which is "flower":

- Example

```
[title~="flower"] {
 border: 5px solid yellow;
}
```



- **CSS [attribute]="value" Selector**

- The [attribute]="value" selector is used to select elements with the specified attribute, whose value can be exactly the specified value, or the specified value followed by a hyphen (-).
- Note: The value has to be a whole word, either alone, like class="top", or followed by a hyphen( - ), like class="top-text".

- Example

```
[class]="top" {
 background: yellow;
}
```





```
<!DOCTYPE html>
<html>
<head>
<style>
[class|=top] {
 background: yellow;
}
</style>
</head>
<body>
<h2>CSS [attribute|="value"] Selector</h2>
<h1 class="top-header">Welcome</h1>
<p class="top-text">Hello world!</p>
<p class="topcontent">Are you learning CSS?</p>
</body>
</html>
```



## CSS [attribute="value"] Selector

**Welcome**

Hello world!

Are you learning CSS?



- CSS [attribute^="value"] Selector
- The [attribute^="value"] selector is used to select elements with the specified attribute, whose value starts with the specified value.
- The following example selects all elements with a class attribute value that starts with "top":
- Note: The value does not have to be a whole word!

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
[class^="top"] {
 background: yellow;
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<h2>CSS [attribute^="value"] Selector</h2>
```

```
<h1 class="top-header">Welcome</h1>
```

```
<p class="top-text">Hello world!</p>
```

```
<p class="topcontent">Are you learning CSS?</p>
```



## CSS [attribute^="value"] Selector

**Welcome**

Hello world!

Are you learning CSS?



- CSS [attribute\$="value"] Selector
- The [attribute\$="value"] selector is used to select elements whose attribute value ends with a specified value.
- The following example selects all elements with a class attribute value that ends with "test":
- Note: The value does not have to be a whole word!

### Example

```
[class$="test"] {
 background: yellow;
}
```



- CSS [attribute\*="value"] Selector
- The [attribute\*="value"] selector is used to select elements whose attribute value contains a specified value.
- The following example selects all elements with a class attribute value that contains "te":
- Note: The value does not have to be a whole word!

### Example

```
[class*="te"] {
 background: yellow;
}
```



# All CSS Attribute Selectors

Selector	Example	Example description
<code>[<u>attribute</u>]</code>	<code>[target]</code>	Selects all elements with a target attribute
<code>[<u>attribute=value</u>]</code>	<code>[target=_blank]</code>	Selects all elements with target="_blank"
<code>[<u>attribute~=value</u>]</code>	<code>[title~=flower]</code>	Selects all elements with a title attribute containing the word "flower"
<code>[<u>attribute =value</u>]</code>	<code>[lang =en]</code>	Selects all elements with a lang attribute value starting with "en"
<code>[<u>attribute^=value</u>]</code>	<code>a[href^="https"]</code>	Selects every <a> element whose href attribute value begins with "https"
<code>[<u>attribute\$=value</u>]</code>	<code>a[href\$=".pdf"]</code>	Selects every <a> element whose href attribute value ends with ".pdf"
<code>[<u>attribute*=value</u>]</code>	<code>a[href*="w3schools"]</code>	Selects every <a> element whose href attribute value contains the substring "w3schools"



- **The CSS background properties are used to add background effects for elements.**
- background-color
- background-image
- background-repeat
- background-attachment
- background-position
- background





- **CSS background-color**
- The background-color property specifies the background color of an element.
- **Example**
- The background color of a page is set like this:

```
body
{
 background-color: lightblue;
}
```



- `div {  
 background-color: green;  
 opacity: 0.3;  
}`
- `div {  
 background: rgba(0, 128, 0, 0.3) /* Green  
background with 30% opacity */  
}`
- `body {  
 background-image: url("paper.gif");  
}`
- `body {  
 background-image: url("gradient_bg.png");  
 background-repeat: repeat-x;  
}`



- **CSS background-position**

- The background-position property is used to specify the position of the background image.

- **Example**

- Position the background image in the top-right corner:

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



- **CSS background-attachment**
- The background-attachment property specifies whether the background image should scroll or be fixed (will not scroll with the rest of the page):
- **Example**
- **Specify that the background image should be fixed:**

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



# CSS Borders

- The CSS border properties allow you to specify the style, width, and color of an element's border.
- CSS Border Style
- The border-style property specifies what kind of border to display.
- **The following values are allowed:**
  - dotted - Defines a dotted border
  - dashed - Defines a dashed border
  - solid - Defines a solid border
  - double - Defines a double border
  - groove - Defines a 3D grooved border.



- The effect depends on the border-color value
- **ridge** - Defines a 3D ridged border.
- The effect depends on the border-color value
- **inset** - Defines a 3D inset border.
- The effect depends on the
- border-color value
- **outset** - Defines a 3D outset border. The effect depends on the border-color value
- **none** - Defines no border
- **hidden** - Defines a hidden border



```
p.dotted {border-style: dotted;}
p.dashed {border-style: dashed;}
p.solid {border-style: solid;}
p.double {border-style: double;}
p.groove {border-style: groove;}
p.ridge {border-style: ridge;}
p.inset {border-style: inset;}
p.outset {border-style: outset;}
p.none {border-style: none;}
p.hidden {border-style: hidden;}
p.mix {border-style: dotted dashed solid
double;}
```



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





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

-



- CSS Shorthand Border Property

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

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

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



- **Left Border**
- **p {**
- **border-left: 6px solid red;**
- **}**
- **Bottom Border**
- **p {**
- **border-bottom: 6px solid red;**
- **}**



- **CSS Rounded Borders**
- **The border-radius property is used to add rounded borders to an element:**

```
p {  
  border: 2px solid red;  
  border-radius: 5px;  
}
```



- **CSS Margins**

- The CSS margin properties are used to create space around elements, outside of any defined borders.
- With CSS, you have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).
- CSS has properties for specifying the margin for each side of an element:
 - margin-top
 - margin-right
 - margin-bottom
 - margin-left



- **All the margin properties can have the following values:**
- auto - the browser calculates the margin
- length - specifies a margin in px, pt, cm, etc.
- % - specifies a margin in % of the width of the containing element
- inherit - specifies that the margin should be inherited from the parent element



- Set different margins for all four sides of a `<p>` element:
- `p {`
 - `margin-top: 100px;`
 - `margin-bottom: 100px;`
 - `margin-right: 150px;`
 - `margin-left: 80px;``}`



- Margin Collapse
- Top and bottom margins of elements are sometimes collapsed into a single margin that is equal to the largest of the two margins.

All CSS Margin Properties

| Property | Description |
|--------------------------------------|---|
| <u>margin</u> | A shorthand property for setting all the margin properties in one declaration |
| <u>margin-bottom</u> | Sets the bottom margin of an element |
| <u>margin-left</u> | Sets the left margin of an element |
| <u>margin-right</u> | Sets the right margin of an element |
| <u>margin-top</u> | Sets the top margin of an element |

- CSS Padding
- Padding is used to create space around an element's content, inside of any defined borders.
- **The CSS padding properties are used to generate space around an element's content, inside of any defined borders.**
- **With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).**



- **CSS Height, Width and Max-width**
- The CSS height and width properties are used to set the height and width of an element.
- The CSS max-width property is used to set the maximum width of an element.
- **CSS height and width Values**
- The height and width properties may have the following values:
 - **auto** - This is default. The browser calculates the height and width
 - **length** - Defines the height/width in px, cm etc.
 - **%** - Defines the height/width in percent of the containing block
 - **initial** - Sets the height/width to its default value
 - **inherit** - The height/width will be inherited from its parent value



- **CSS height and width Examples**
- **This element has a height of 200 pixels and a width of 50%**
- ```
div {
 height: 200px;
 width: 50%;
 background-color: powderblue;
}
```



- CSS Layout – The position Property
- **The position property specifies the type of positioning method used for an element (static, relative, fixed, absolute or sticky).**
- **The position property specifies the type of positioning method used for an element (static, relative, fixed, absolute or sticky).**
- **The position Property**
- **The position property specifies the type of positioning method used for an element.**



- **position: static;**
- HTML elements are positioned static by default.
- Static positioned elements are not affected by the top, bottom, left, and right properties.
- An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page:



- `div.static {  
 position: static;  
 border: 3px solid #73AD21;  
}`



- **position: relative;**
- **An element with position: relative; is positioned relative to its normal position.**
- **Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.**
- 



```
div.relative {
 position: relative;
 left: 30px;
 border: 3px solid #73AD21;
}
```





- **position: fixed;**
- **An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled.**
- **The top, right, bottom, and left properties are used to position the element.**
- **A fixed element does not leave a gap in the page where it would normally have been located.**



```
div.fixed {
 position: fixed;
 bottom: 0;
 right: 0;
 width: 300px;
 border: 3px solid #73AD21;
}
```



- **position: absolute;**
- **An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).**
- **However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.**



Here is a simple example:

This <div> element has position: relative;

This <div> element has  
position: absolute;



```
div.relative {
 position: relative;
 width: 400px;
 height: 200px;
 border: 3px solid #73AD21;
}
```

```
div.absolute {
 position: absolute;
 top: 80px;
 right: 0;
 width: 200px;
 height: 100px;
 border: 3px solid #73AD21;
}
```



- **position: sticky;**
- An element with `position: sticky;` is positioned based on the user's scroll position.
- A sticky element toggles between relative and fixed, depending on the scroll position.
- It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like `position: fixed`).



- Note: Internet Explorer does not support sticky positioning. Safari requires a -webkit- prefix (see example below).
- You must also specify at least one of top, right, bottom or left for sticky positioning to work.

```
div.sticky {
 position: -webkit-sticky; /* Safari */
 position: sticky;
 top: 0;
 background-color: green;
 border: 2px solid #4CAF50;
}
```



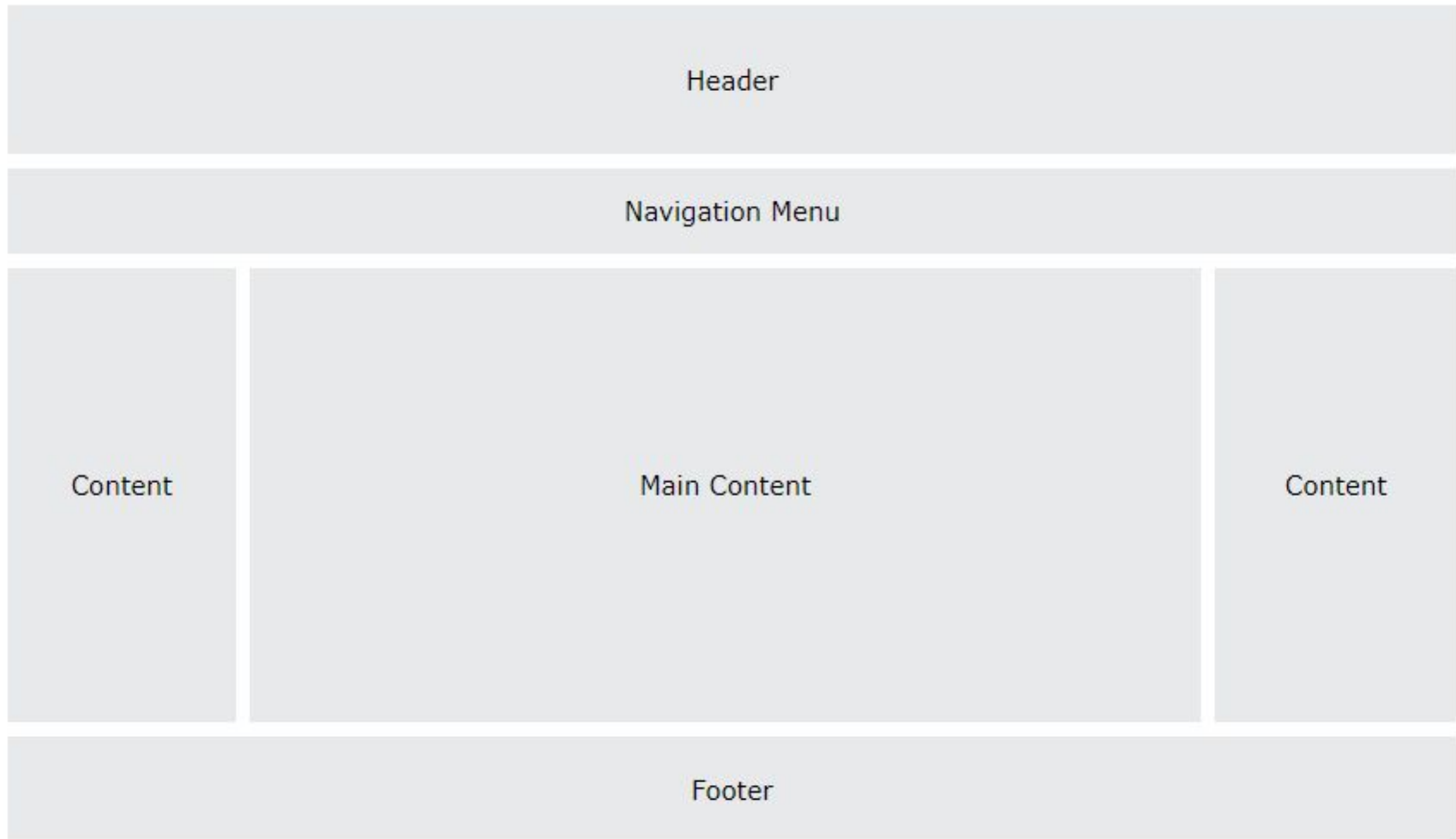
# All CSS Positioning Properties

Property	Description
<u>bottom</u>	Sets the bottom margin edge for a positioned box
<u>clip</u>	Clips an absolutely positioned element
<u>left</u>	Sets the left margin edge for a positioned box
<u>position</u>	Specifies the type of positioning for an element
<u>right</u>	Sets the right margin edge for a positioned box
<u>top</u>	Sets the top margin edge for a positioned box





- **CSS Website Layout**



- **Header**
- **A header is usually located at the top of the website (or right below a top navigation menu). It often contains a logo or the website name:**

- **Example**

```
.header {
 background-color: #F1F1F1;
 text-align: center;
 padding: 20px;
}
```



```
/* The navbar container */
```

```
.topnav {
 overflow: hidden;
 background-color: #333;
}
```

```
* Navbar links */
```

```
.topnav a {
 float: left;
 display: block;
 color: #f2f2f2;
 text-align: center;
 padding: 14px 16px;
 text-decoration: none;
}
```

```
/* Links - change color on hover */
```

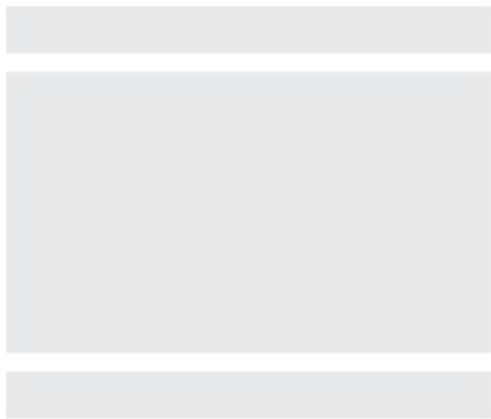
```
.topnav a:hover {
 background-color: #ddd;
 color: black;
```



- **Content**
- **The layout in this section, often depends on the target users. The most common layout is one (or combining them) of the following:**
  - **1-column (often used for mobile browsers)**
  - **2-column (often used for tablets and laptops)**
  - **3-column layout (only used for desktops)**



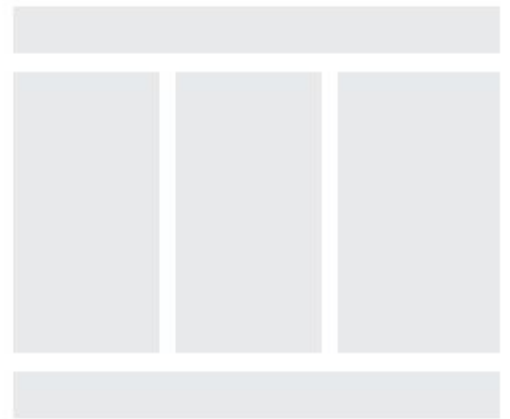
1-column:



2-column:



3-column:



- **CSS Combinators**
- **A CSS selector can contain more than one simple selector.**
- **Between the simple selectors, we can include a combinator.**
- **There are four different combinators in CSS:**
  - **descendant selector (space)**
  - **child selector (>)**
  - **adjacent sibling selector (+)**
  - **general sibling selector (~)**



- **Descendant Selector**
- **The descendant selector matches all elements that are descendants of a specified element.**
- **The following example selects all <p> elements inside <div> elements:**
- **Example**  
**div p {**  
    **background-color: yellow;**  
**}**



- **Child Selector (>)**
- **The child selector selects all elements that are the children of a specified element.**
- **The following example selects all <p> elements that are children of a <div> element:**
- **Example**
- **div > p {**
- **background-color: yellow;**
- **}**





- **Adjacent Sibling Selector (+)**
- **The adjacent sibling selector is used to select an element that is directly after another specific element.**
- **Sibling elements must have the same parent element, and "adjacent" means "immediately following".**
- **The following example selects the first <p> element that are placed immediately after <div> elements:**
- **Example**
- **div + p {**
- **background-color: yellow;**
- **}**



- **General Sibling Selector (~)**
- **The general sibling selector selects all elements that are next siblings of a specified element.**
- **The following example selects all <p> elements that are next siblings of <div> elements:**
- **Example**
- **div ~ p {**
- **background-color: yellow;**
- **}**



## **BACKGROUND IMAGES, COLORS AND PROPERTIES**

**Using css it is possible to control the color of an element's background, use an image in background , tile an image or place an image at specific place on a page.**

# BACKGROUND COLOR

- ❑ The background-color property specifies the background color of an element.
- ❑ The background color of a page is defined in the body selector

**body**

```
{
 background: #ffff20; /* set all background properties in
 one direction rgb */
 color: #000000; /* text color */
}
```

**With CSS, a color is most often specified by:**

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

<b>Background</b>	<b>Set all background in one declaration</b>	<b>RGB</b>
<b>Background-attachm ent</b>	<b>Sets whether image moves with page when scroll</b>	<b>Scroll or fixed</b>
<b>Background-color</b>	<b>Sets background color of an element</b>	<b>RGB,hex,name</b>
<b>Background-image</b>	<b>Set as image in background</b>	<b>url</b>
<b>Background-position</b>	<b>Set the starting position of an image in the background</b>	<b>Top left, Top center, Top right, Center left, Center center, center right, Bottom left, Bottom center, Bottom right, X-% y-% X-pos y-pos</b>
<b>Background-repeat</b>	<b>Sets the repetition of an image</b>	<b>Repeat, no-repeat, Repeat-x,repeat-y</b>

```
body {
 background-image: url("images/tile.png");
}
```

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

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



# BACKGROUND STYLING EXAMPLES

- Background □ output
- **Background\_attachment** □ output
- Background output
- Background-color □ output
- **Background-repeat**
  1. no-repeat □ result
  2. repeat □ result
  3. repeat-x □ result
  4. repeat-y □ result
- Background\_image □ output



# BACKGROUND STYLING EXAMPLES

## ▣ **Background\_position:**

- ▣ [Left\\_top.html](#)      ▣ [output](#)
- ▣ [Left\\_cenetr.html](#)      ▣ [output](#)
- ▣ [Left\\_bottom.html](#)      ▣ [output](#)
- ▣ [Right\\_top.html](#)      ▣ [output](#)
- ▣ [Right\\_cenetr.html](#)      ▣ [output](#)
- ▣ [Right\\_bottom.html](#)      ▣ [output](#)
- ▣ [Center\\_top.html](#)      ▣ [output](#)
- ▣ [Center\\_cenetr.html](#)      ▣ [output](#)
- ▣ [Center\\_bottom.html](#)      ▣ [output](#)
- ▣ [Percentage.html](#)      ▣ [output](#)
- ▣ [Pixels.html](#)      ▣ [output](#)





- Here are the most commonly-used CSS properties related to text.
- [direction](#) [letter-spacing](#) [line-height](#) [text-align](#) [text-decoration](#) [text-indent](#) [text-transform](#) [word-spacing](#) direction
- The direction property specifies the text direction. Possible values are 'ltr' and 'rtl'.
- For example, with a CSS declaration of,
- ```
p {  
  direction:ltr;  
}
```

- `<p>LTR Direction</p>`
- renders
- LTR Direction
- With a CSS declaration of,
- ```
p {
 direction:rtl;
}
```
- The following HTML,
- `<p>RTL Direction</p>`
- renders

## RTL Direction □

Example

output

## ❑ **letter-spacing**

- ❑ The letter-spacing property specifies the amount of space between characters. For example, with a CSS declaration of,
- ❑ 

```
p {
 letter-spacing:8px;
}
```
- ❑ The following HTML,
- ❑ `<p>8px between letters</p>`
- ❑ renders
- ❑ 8px between letters

## □ **line-height**

- The line-height property specifies the amount of space between lines. For example, with a CSS declaration of,
- ```
p {  
  line-height:30px;  
}
```
- The following HTML,
- `<p>30px between line 1
and line 2.</p>`
- renders
- 30px between line 1
and line 2.

□ **text-align**

- The text-align property specifies how text is justified. Possible values are:
- left: left-justified
- right: right-justified
- center: text is centered
- justified: text is both right- and left-justified .

| CSS Declaration | Output |
|--|---|
| <pre>p {
 text-align:left;
}</pre> | This sentence illustrates what it looks like to be left-justified. |
| <pre>p {
 text-align:right;
}</pre> | This sentence illustrates what it looks like to be right-justified. |
| <pre>p {
 text-align:center;
}</pre> | This sentence illustrates what it looks like to be centered. |
| <pre>p {
 text-align:justify;
}</pre> | This sentence illustrates what it looks like to be fully-justified. |

text-decoration

The text-decoration property specifies how text is decorated.

Possible values are:

- underline: adds an underline to the text overline: adds a line on top of the text
- line-through: adds a line through the middle of the text. blink: causes the text to blink.

| CSS Declaration | Output |
|--|---|
| <pre>p {
 text-decoration:underline;
}</pre> | <u>An underline example</u> |
| <pre>p {
 text-decoration:overline;
}</pre> | <u>An overline example</u> |
| <pre>p {
 text-decoration:line-through;
}</pre> | A strikethrough (line-through) example |

□ **text-indent**

- The text-indent property specifies how much space to indent before the first line of the text in a block. Both length and percentage can be used. For example, with a CSS declaration of,

- ```
p {
 text-indent: 15px;
}
```

# MANIPULATING TEXT

Color	Set color to the text	RGB, hex, name
Background-color	Sets background color of an element	RGB, hex, name
text-align	Text get align	Left , right ,center
Letter-spacing	Spacing of the letters is manipulated on specific text	-3px to dec by 3px + 0.6 to inc by 0.6
Text-decoration	Decorate a text while display	Overline Line-through Underline blink none
Text-indent	Text get indented	fixed indentation in px, pt, cm, em, etc. % of the width of the parent element
Text-transform	Case of a text get control	Uppercase, lowercase, capitalize



# EXAMPLE OF TEXT

□ style4.txt

□ text.html □ OUTPUT

- **Styling Links with CSS**

- A link has four different states — link, visited, active and hover. These four states of a link can be styled differently through using the following anchor pseudo-class selectors.
- `a:link` — define styles for normal or unvisited links.
- `a:visited` — define styles for links that the user has already visited.
- `a:hover` — define styles for a link when the user place the mouse pointer over it.
- `a:active` — define styles for links when they are being clicked.



```
a:link { /* unvisited link */
 color: #ff0000;
 text-decoration: none;
 border-bottom: 1px solid;
}
a:visited { /* visited link */
 color: #ff00ff;
}
a:hover { /* mouse over link */
 color: #00ff00;
 border-bottom: none;
}
a:active { /* active link */
 color: #00ffff;
}
```

- Example



- ❑ **CSS Box Model**
- ❑ **The CSS Box Model**
- ❑ **All HTML elements can be considered as boxes. In CSS, the term "box model" is used when talking about design and layout.**
- ❑ **The CSS box model is essentially a box that wraps around HTML elements, and it consists of: margins, borders, padding, and the actual content.**
- ❑ **The box model allows us to place a border around elements and space elements in relation to other elements.**
- ❑ **The image below illustrates the box model:**

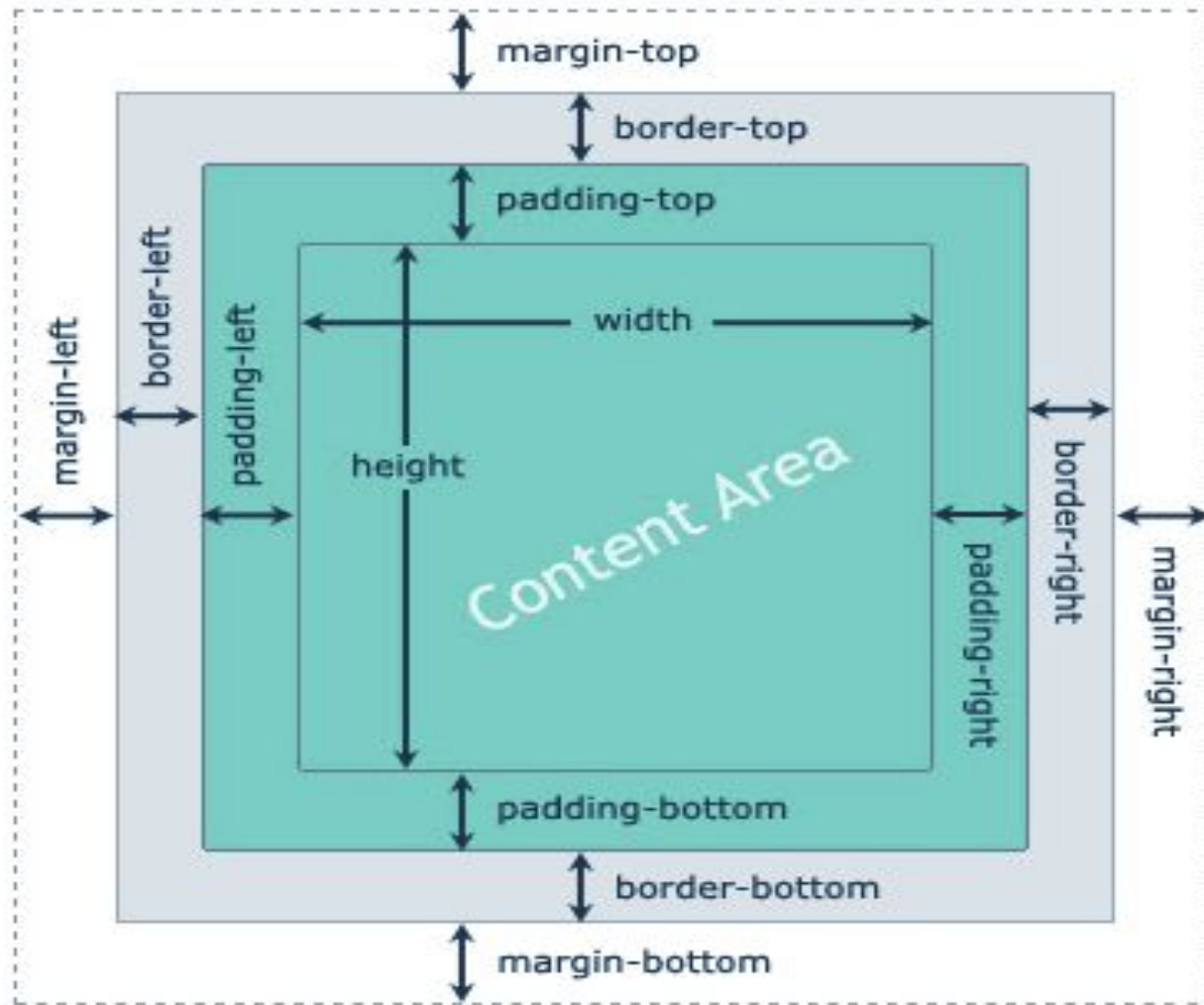


**MARGIN** - CLEARS AN AREA AROUND THE BORDER. THE MARGIN DOES NOT HAVE A BACKGROUND COLOR, IT IS COMPLETELY TRANSPARENT

**BORDER** - A BORDER THAT GOES AROUND THE PADDING AND CONTENT. THE BORDER IS AFFECTED BY THE BACKGROUND COLOR OF THE BOX

**PADDING** - CLEARS AN AREA AROUND THE CONTENT. THE PADDING IS AFFECTED BY THE BACKGROUND COLOR OF THE BOX

**CONTENT** - THE CONTENT OF THE BOX, WHERE TEXT AND IMAGES APPEAR



# FONT

- ❑ In CSS, there are two types of font family names:
- ❑ generic family - a group of font families with a similar look (like "Serif" or "Monospace")
- ❑ font family - a specific font family (like "Times New Roman" or "Arial")

Generic family	Font family	Description
Serif	Times New Roman Georgia	Serif fonts have small lines at the ends on some characters
Sans-serif	Arial Verdana	"Sans" means without - these fonts do not have the lines at the ends of characters
Monospace	Courier New Lucida Console	All monospace characters have the same width

# USING FONTS

font-family		Times, courier, sans-serif.
font-size	Set size of a text	100%,150% Or Small, medium, large
font-style	Set font to normal ,italic or oblique	Normal, italic, oblique

- Font-family.html   [output](#)
- Font-size.html   [output](#)
- Font-style.html   [output](#)





# STYLING LINKS

- Links can be styled with any CSS property (e.g. color, font-family, background, etc.).
- The four links states are:
- **a:link** - a normal, unvisited link
- **a:visited** - a link the user has visited
- **a:hover** - a link when the user keeps mouse over it
- **a:active** - a link the moment it is clicked

**When setting the style for several link states, there are some order rules:**

- a:hover MUST come after a:link and a:visited
- a:active MUST come after a:hover

# STYLING LINKS EXAMPLES

- ▣ **Css\_link.html** ▣ [output](#)
- ▣ **Css\_link1.html** [▣](#) [output](#)
- ▣ **Css\_link2.html** [▣](#) [output](#)

# CSS PADDING

- ❑ The padding clears an area around the content (inside the border) of an element.
- ❑ The padding is affected by the background color of the element.
- ❑ The top, right, bottom, and left padding can be changed independently using separate properties.
- ❑ A shorthand padding property can also be used, to change all paddings at once.

# PADDING - INDIVIDUAL SIDES

- ❑ **padding-top:25px;**
- ❑ **padding-bottom:25px;**
- ❑ **padding-right:50px;**
- ❑ **padding-left:50px;**
- ❑ [Css\\_padding1.html](#) ❑ [output](#)

# PADDING - SHORTHAND PROPERTY

- To shorten the code, it is possible to specify all the padding properties in one property. This is called a shorthand property.
- The padding property can have from one to four values.
- **padding:25px 50px 75px 100px;**
  - top, right , bottom , left
- **padding:25px 50px 75px;**
  - top , right and left , bottom
- **padding:25px 50px;**
  - top and bottom, right and left
- **padding:25px;**
  - all four paddings are 25px

[Css\\_padding2.html](#) □ [output](#)

# CSS MARGIN

- The margin clears an area around an element (outside the border).
- The margin does not have a background color, and is completely transparent.
- The top, right, bottom, and left margin can be changed independently using separate properties.
- A shorthand margin property can also be used, to change all margins at once.

# MARGIN - INDIVIDUAL SIDES

- ❑ **margin-top:100px;**
- ❑ **margin-bottom:100px;**
- ❑ **margin-right:50px;**
- ❑ **margin-left:50px;**
- ❑ Css\_margin1.html ❑ output

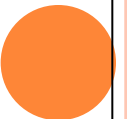
# MARGIN - SHORTHAND PROPERTY

▣ [Css\\_margin2.html](#) ▣ [output](#)



# LISTS

Property	Description
<a href="#"><u>list-style</u></a>	Sets all the properties for a list in one declaration
<a href="#"><u>list-style-image</u></a>	Specifies an image as the list-item marker
<a href="#"><u>list-style-position</u></a>	Specifies if the list-item markers should appear inside or outside the content flow
<a href="#"><u>list-style-type</u></a>	Specifies the type of list-item marker



# LISTS

- list-style □ output
- list-style-image □ output
- list-style-position □ output
- list-style-type □ output



# STYLING BORDERS

border-width	Sets width for all attributes	Thin ,medium ,thick
border-color	Sets color of four border	Name, hex, rgb
border-style	Sets style of four border	None, hidden, dotted, dashed, solid, double, groove, ridge, inset, outset
border-top-color	Set color for top border	Same as border-color
border-top-style	Set style for top border	Same as border-style
border-top-width	Set width for top border	Same as border-width

Last 3 attributes can be applied to

border-bottom- ....,

border-left-.....,

border-right-..



# EXAMPLE OF BORDER

- border-style □ output
- border-width □ output
- border-color □ output
- border-top-color □ output
- border-top-style □ **output**
- **border-top-width** □ **output**

## ***CSS TABLE PROPERTIES:***

THE CSS TABLE PROPERTIES ALLOW YOU TO SET THE LAYOUT OF A TABLE.

<b>Property</b>	<b>Description</b>	<b>Values</b>
border-collapse	Sets whether the table borders are collapsed into a single border or detached as in standard HTML	collapse separate
border-spacing	Sets the distance that separates cell borders (only for the "separated borders" model)	<i>length length</i>
caption-side	Sets the position of the table caption	top bottom left right
empty-cells	Sets whether or not to show empty cells in a table (only for the "separated borders" model)	show hide
table-layout	Sets the algorithm used to display the table cells, rows, and columns	auto fixed

## ❑ **Tableexample**                      **Output**

- ❑ The table in the example above has double borders. This is because both the table and the th/td elements have separate borders.
- ❑ Collapse Borders
- ❑ The border-collapse property sets whether the table borders are collapsed into a single border or separated:

## ❑ **Tableexample1**                      **Output**

## ❑ **Tableexample2**                      **Output**

## ❑ **Table Text Alignment**

- ❑ The text in a table is aligned with the text-align and vertical-align properties.
- ❑ The text-align property sets the horizontal alignment, like left, right, or center:

❑ <b>cssHalign</b>	<b>Output</b>
--------------------	---------------

❑ <b>CSSValign</b>	<b>Output</b>
--------------------	---------------

- ❑ **Table Padding**
- ❑ To control the space between the border and content in a table, use the padding property on td and th elements:
- ❑ **Csstablepadding**                      **output**
- ❑ **Table Color**
- ❑ The example below specifies the color of the borders, and the text and background color of th elements:
- ❑ **CSStablecolor**              **CSStablecolor**
- ❑ **CSSTableExample**              **CSSTableExampleOutput**



# Styling Tables

▣ **Table\_style.html** ▣ **output**



## ❑ **Floating elements (floats)**

- ❑ An element can be floated to the right or to left by using the property float.
- ❑ That is to say that the box with its contents either floats to the right or to the left in a document (or the containing box) The following figure illustrates the principle:



### A floating image

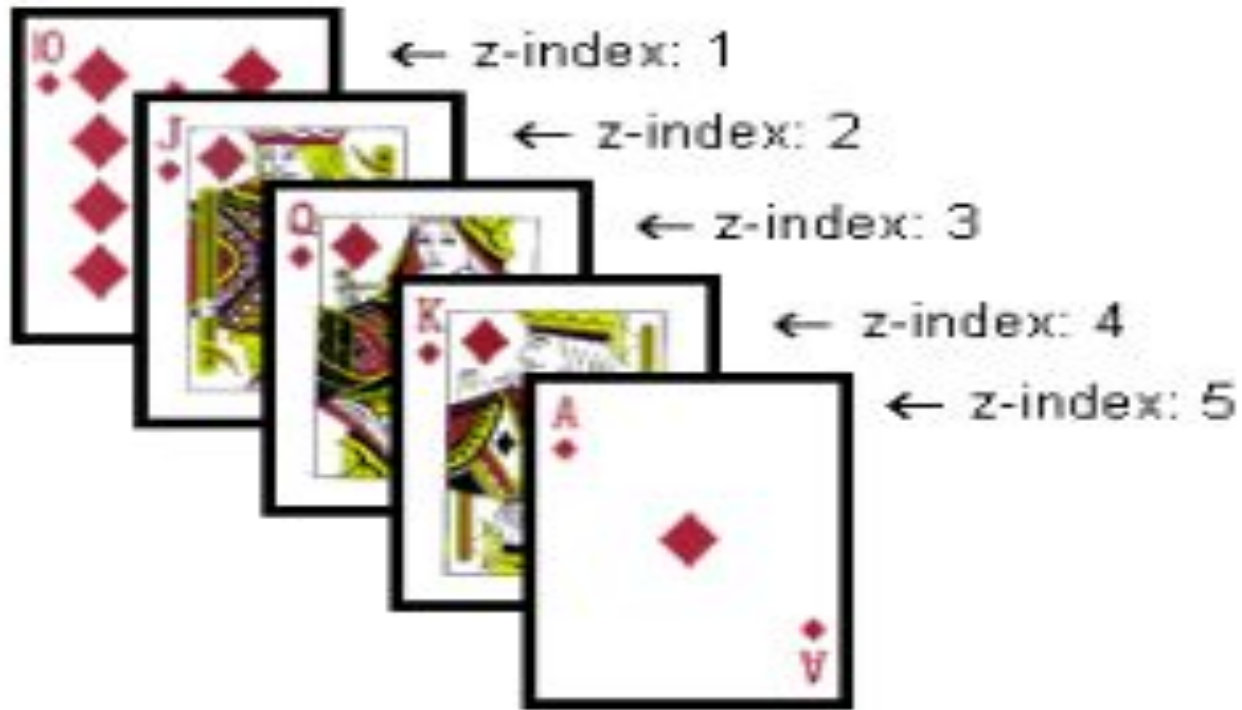
Iste quidem veteres inter ponetur honeste, qui  
vel mense brevi vel toto est iunior anno. Utor  
permisso, caudaeque pilos ut equinae paulatim  
vello unum, demo etiam unum, dum cadat elusus  
Interdum volgus rectum videt, est ubi peccat.  
Si veteres ita miratur laudatque poetas, ut  
nihil anteferat, nihil illis comparet, errat.  
Si quaedam nimis antique, si peraque dure

Interdum volgus rectum videt, est ubi peccat. Si veteres ita miratur laudatque  
poetas, ut nihil anteferat, nihil illis comparet, errat. Si quaedam nimis antique  
si peraque dure

FLOAT IMAGE      OUTPUT

IMAGEIMAGE      OUTPUT

- ❑ **CSS operates in three dimensions - height, width and depth.**
- ❑ **We have seen the first two dimensions in previous lessons.**
- ❑ **Layers of elements in short, this means the order of which the elements overlap one another.**
- ❑ **For that purpose, you can assign each element a number (z-index).**
- ❑ **The system is that an element with a higher number overlaps an element with a lower number.**
- ❑ **The z-index property specifies the stack order of an element.**
- ❑ **An element with greater stack order is always in front of an element with a lower stack order.**
- ❑ **z-index only works on positioned elements (position: absolute, position: relative, or position: fixed).**
- ❑ **Let us say we are playing poker and have a royal flush.**
- ❑ **Our hand can be presented in a way where each card has got a z-index:**



ZINDEXLAYOUT      OUTPUT

ZINDEXCSS.CSS

## □ Csslayoutexample**Csslayoutexample** output

- **CSS Comments**

- Comments are used to explain the code, and may help when you edit the source code at a later date.
- Comments are ignored by browsers.
- A CSS comment is placed inside the <style> element, and starts with /\* and ends with \*/:
- Example

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

```
p {
 color: red;
}
```



- `<h1 style="background-color:DodgerBlue;">Hello World</h1>`
- `<p style="background-color:Tomato;">Lorem ipsum...</p>`
- `<h1 style="background-color:rgb(255, 99, 71);">...</h1>`
- `<h1 style="background-color:#ff6347;">...</h1>`
- `<h1 style="background-color:hsl(9, 100%, 64%;">...</h1>`
- `<h1 style="background-color:rgba(255, 99, 71, 0.5);">...</h1>`
- `<h1 style="background-color:hsla(9, 100%, 64%, 0.5);">...</h1>`





- **rgb(red, green, blue)**
- **HEX Value**
- **#rrggbb**



## CSS Backgrounds:

- The CSS background properties are used to add background effects for elements.

```
h1 {
 background-color: green;
}
div {
 background-color: lightblue;
}
p {
 background-color: yellow;
}
```



- **CSS Backgrounds:**

- background-color
- background-image
- background-repeat
- background-attachment
- background-position
- background (shorthand property)

- **Opacity / Transparency:**

- The opacity property specifies the opacity/transparency of an element.
- It can take a value from 0.0 - 1.0.
- The lower value, the more transparent:



```
div {
 background-color: green;
 opacity: 0.3;
}
```



- **CSS background-image**
- The background-image property specifies an image to use as the background of an element.
- By default, the image is repeated so it covers the entire element.

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



**<!DOCTYPE html>**

**<html>**

**<head>**

**<style>**

**a:link {**

**text-decoration: none;**

**}**

**a:visited {**

**text-decoration: none;**

**}**

**a:hover {**

**text-decoration: underline;**

**}**

**a:active {**

**text-decoration: underline;**

**}**

**</style>**



```
</head>
```

```
<body>
```

```
<h2>CSS Links</h2>
```

```
<p>This is a
link</p>
```

```
<p>Note: a: hover MUST come after a: link and a: visited in
the CSS definition in order to be effective.</p>
```

```
<p>Note: a: active MUST come after a: hover in the CSS
definition in order to be effective.</p>
```

```
</body>
```

```
</html>
```



- **CSS3 features:**

- Using CSS3 Borders
- The CSS3 provides two new properties for styling the borders of an element in a more elegant way
- border-image property: for adding the images to borders
- border-radius property: for making the rounded corners without using any images.





- **Creating CSS3 Rounded Corners**
- **The border-radius property can be used to create rounded corners.**
- **This property typically defines the shape of the corner of the outer border edge.**
- **Prior to CSS3, sliced images are used for creating the rounded corners that was rather bothersome.**

- **Example**

```
.box {
 width: 300px;
 height: 150px;
 background: #ffb6c1;
 border: 2px solid #f08080;
 border-radius: 20px;
}
```




```
<!Doctype html> <!--Rounded Border -->
<head>
<style>
.box {
 width: 300px;
 height: 150px;
 background: #ffb6c1;
 border: 2px solid #f08080;
 border-radius: 20px;
}
</style>
</head>
<body>
<div class=box>
<p> Rounded boarder </p>
</div>
</body>
</html>
```



- **Adding CSS3 Border Images**

- The border-image property allows you to specify an image to act as an element's border.
- The design of the border is created from the sides and corners of the image specified in border-image source URL.
- The border image may be sliced, repeated, scaled and stretched in various ways to fit the size of the border image area.

```
.box {
 width: 300px;
 height: 150px;
 border: 15px solid transparent;
 -webkit-border-image: url("border.png") 30 30 round; /* Safari
3.1-5 */
 -o-border-image: url("border.png") 30 30 round; /* Opera 11-12.1
*/
 border-image: url("border.png") 30 30 round;
}
```



```
<!Doctype html>
<head>
<style>
.box {
 width: 300px;
 height: 150px;
 background: #ffb6c1;
 border: 2px solid #f08080;
 border-radius: 20px;
 border: 15px solid transparent;
 border-image: url("border.png") 30 30 round;
}
</style>
</head>
<body>
<div class=box>
<p> Rounded boarder </p>
</div>
</body>
</html>
```



- **Online Reference:**
- **<https://www.tutorialrepublic.com/css-examples.php>**

