

# HTML Advanced Concepts & CSS

## 5 CHAPTER

Tables and forms are integral part of any website or webpage. In this chapter we are going to discuss these advanced concepts of HTML and their attributes. We will also discuss the basic concepts of CSS, a simple design language intended to simplify the process of making web pages presentable.

### HTML Tables

The HTML tables allow web authors to arrange data like text, images, links, other tables, etc. into rows and columns of cells.

The HTML tables are created using the `<table>` tag in which the `<tr>` tag is used to create table rows and `<td>` tag is used to create data cells. The elements under `<td>` are regular and left aligned by default. Table cells which act as column headers or row headers should use the `<th>` (table header) element which also comes inside the `<td>` tag. A simple example is given below.

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Tables</title>
</head>
<body>
<table border = "1">
<tr>
<th>Firstname</th>
<th>Ages</th>
</tr>
```

Handwritten notes:   
 create data cells   
 fd →   
 fd →   
 fd →

```
<td>Alice</td>
<td>bob </td>
</tr>
```

```
<tr>
<td>20</td>
<td>21</td>
</tr>
</table>
</body>
</html>
```

This will produce the following result.

Firstname	Age
Alice	20
bob	21

We can specify border for HTML tables either using border attribute of table in HTML, or using border property in CSS. There are two attributes called cellpadding and cellspacing which you will use to adjust the white space in your table cells. The cellpadding attribute defines space between table cells, while cellpadding represents the distance between cell borders and the content within a cell. An example is given below.

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Tables</title>
</head>
<body>
```

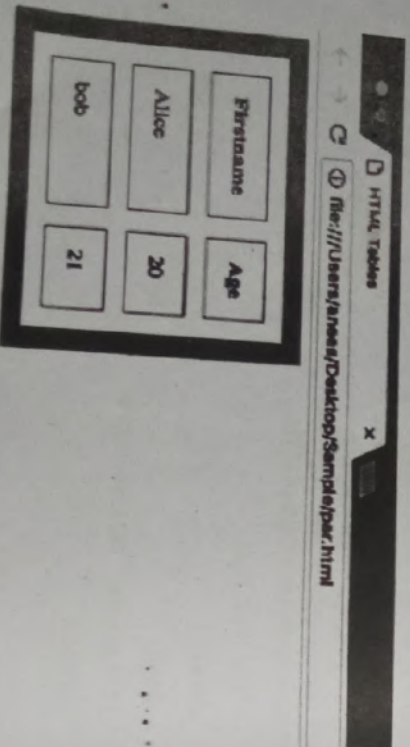


<table border = "15" cellpadding = "15" cellspacing = "15",

```
<tr>
<th>Firstname</th>
<th>Age</th>
</tr>
<tr>
<td>Alice</td>
<td>bob </td>
</tr>
```

```
<tr>
<td>20</td>
<td>21</td>
</tr>
</table>
</body>
</html>
```

The following table will be obtained.



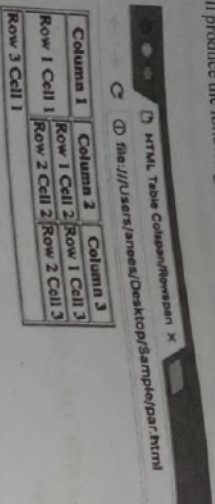
Firstname	Age
Alice	20
bob	21

Table cells can be merged using the colspan and rowspan attributes. You can use colspan attribute if you want to merge two or more columns into a single column. Similar way you can use rowspan if you want to merge two or more rows.

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Table Colspan/Rowspan</title>
</head>
<body>
<table border = "1">
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
</tr>
<tr>
<td rowspan = "2">Row 1 Cell 1</td>
<td>Row 1 Cell 2</td>
<td>Row 1 Cell 3</td>
</tr>
<tr>
<td>Row 2 Cell 2</td>
<td>Row 2 Cell 3</td>
</tr>
<tr>
<td colspan = "3">Row 3 Cell 1</td>
</tr>
</table>
</body>
</html>
```



This will produce the following result.



Column 1	Column 2	Column 3
Row 1 Cell 1	Row 2 Cell 2	Row 3 Cell 1
Row 2 Cell 2	Row 3 Cell 1	
Row 3 Cell 1		

Table background can be set using one of the following two ways

- bgcolor attribute " You can set background color for whole table or just for one cell.
- background attribute " You can set background image for whole table or just for one cell.

You can also set border color also using bordercolor attribute. The syntax is given below.

```
<tableborder="1"bordercolor="green"bgcolor="yellow">
<tableborder="1"bordercolor="green"background="/images/
test.png">
```

You can set a table width and height using width and height attributes. Table width or height can be set in terms of pixels or in terms of percentage of available screen area.

Example

```
<tableborder="1"width="400"height="150">
```

A caption can be added to a table using the <caption> element.

```
<caption>This is the caption</caption>
```

Tables can be broken into sections using the following elements:

- <thead> — Table header
- <tbody> — Table body
- <tfoot> — Table footer

A table may contain several <tbody> elements to indicate different pages or groups of data. But it is notable that <thead> and <tfoot> tags should appear before <tbody>. An example describing table background, setting height and width, inserting caption and setting head and foot is given below.

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Table</title>
</head>
<body>
<table border = "1" width = "400" height = "150" bordercolor
= "green" bgcolor = "yellow">
<caption>Student Details</caption>
<thead>
<tr>
<td colspan = "4">This is the head of the table</td>
</tr>
</thead>
<tbody>
<tr>
<td colspan = "4">This is the foot of the table</td>
</tr>
<tr>
<td>alice</td>
<td>Bob</td>
<td>Tom</td>
```

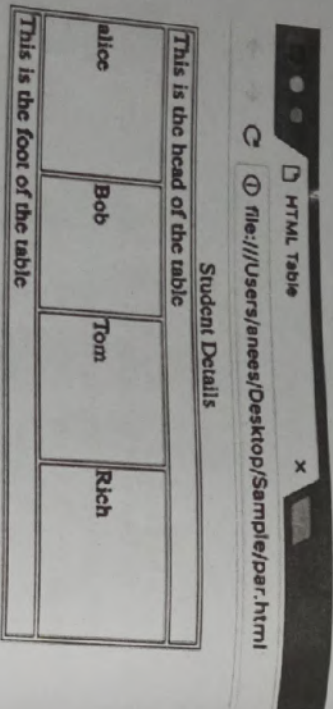


```

<td>Rich</td>
</tr>
</tbody>
</table>
</body>

```

This will produce the following result



You can use `<col>` and `<colgroup>` to define table columns for styling. However, there are a number of limitations with this practice. Column groups allow authors to create structural divisions within a table. Authors may highlight this structure through style sheets or HTML attributes (e.g., the `rules` attribute for the `TABLE` element). For an example of the visual presentation of column groups, please consult the sample table. A table may either contain a single implicit column group (no `COLGROUP` element delimits the columns) or any number of explicit column groups (each delimited by an instance of the `COLGROUP` element). The `COL` element allows authors to share attributes among several columns without implying any structural grouping. The `"span"` of the `COL` element is the number of columns that will share the element's attributes.

### HTML Forms

An HTML form is a section of document that contains interactive controls that enable a user to submit information to a web server. HTML Forms are required to collect different kinds of user inputs, such as contact details like name, email address, phone numbers, or details like credit card information, etc. Forms contain special elements called controls like inputbox, checkboxes, radio-buttons, submit buttons, etc. Users generally complete

a form by modifying its controls e.g. entering text, selecting items, etc. and submitting this form to a web server for processing. The HTML `<form>` tag is used to create an HTML form and it has following syntax.

```

<form action = "Script URL" method = "GET|POST">
  form elements like input, textarea etc.
</form>

```

Most frequently used form attributes are:

Sr.No	Attribute & Description
1	<b>action</b> Backend script ready to process your passed data.
2	<b>method</b> Method to be used to upload data. The most frequently used are GET and POST methods.
3	<b>target</b> Specify the target window or frame where the result of the script will be displayed. It takes values like <code>_blank</code> , <code>_self</code> , <code>_parent</code> etc.
4	<b>enctype</b> You can use the <code>enctype</code> attribute to specify how the browser encodes the data before it sends it to the server. Possible values are – application/x-www-form-urlencoded – This is the standard method most forms use in simple scenarios. multipart/form-data – This is used when you want to upload binary data in the form of files like image, word file etc.

HTTP POST requests supply additional data from the client (browser) to the server in the message body. In contrast, GET requests include all required data in the URL. Forms in HTML can use either method by specifying `method="POST"` or `method="GET"` (default) in the `<form>` element. The method specified determines how form data is submitted to the server. When the method is GET, all form data is encoded into the URL, appended to the action URL as query string parameters. With POST, form data appears within the message body of the HTTP request.



Difference between GET and POST is given below.

GET	POST
Parameters remain in browser history because they are part of the URL.	Parameters are not saved in browser history.
Can be bookmarked.	Cannot be bookmarked.
GET requests are re-secured but may not be re-submitted to server if the HTML is stored in the browser cache.	The browser usually alerts the user that data will need to be re-submitted.
application/x-www-form-urlencoded	multipart/form-data or application/x-www-form-urlencoded. Use multipart encoding for binary data.
can send but the parameter data is limited to what we can stuff into the request line (URL). Safest to use less than 2K of parameters, some servers handle up to 64K.	Can send parameters, including uploading files, to the server.
Easier to hack for script kiddies.	More difficult to hack.
only ASCII characters allowed.	No restrictions on form of data type. Binary data is also allowed.
GET is less secure compared to POST because data sent is part of the URL. So it's saved in browser history and server logs in plaintext.	POST is a little safer than GET because the parameters are not stored in browser history or in web server logs.
Since form data is in the URL and URL length is restricted.	No restrictions.
GET method should not be used when sending passwords or other sensitive information.	POST method used when sending passwords or other sensitive information.
GET method is visible to everyone and has limits on the amount of information to send.	POST method variables are not displayed in the URL.
Can be cached.	Not cached.

### HTML Form Controls

There are different types of form controls that you can use to collect data using HTML form. They are

- Text Input Controls
- Checkboxes Controls
- Radio Box Controls
- Select Box Controls
- File Select boxes
- Hidden Controls
- Clickable Buttons
- Submit and Reset Button

#### Text Input Controls

There are three types of text input used on forms

1. Single-line text input controls " This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML <input> tag.
2. Password input controls " This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTML <input> tag.
3. Multi-line text input controls " This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML <textarea> tag.

#### Single-line text input controls

Single-line text input controls are created using an <input> element, whose type attribute has a value of text. Here's an example of a single-line text input used to take username.

```
<!DOCTYPE html>
<html>
<head>
<title>Text Input Control</title>
</head>
<body>
<form >
```



```
User name: <input type = "text" name = "username" />
```

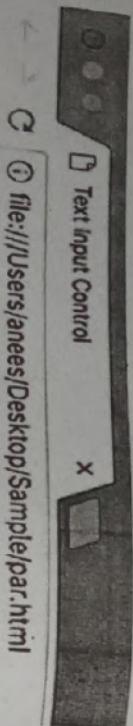
&lt;br&gt;

&lt;/form&gt;

&lt;/body&gt;

&lt;/html&gt;

This will give you the following result.

User name: 

Following is the list of attributes for <input> tag for creating text field.

Attribute	Description
type	Indicates the type of input control and for text input control it will be set to text.
name	Used to give a name to the control which is sent to the server to be recognized and get the value.
value	This can be used to provide an initial value inside the control.
size	Allows to specify the width of the text-input control in terms of characters.
maxlength	Allows to specify the maximum number of characters a user can enter into the text box.

### Password Field

Password fields are similar to text fields. The only difference is; characters in a password field are masked i.e. shown as asterisks or dots. This is to prevent others from reading the password on the screen. This is also a single-line text input controls created using an <input> element whose type attribute has a value of password. Here's an example of a single-line password input used to take user password.

&lt;!DOCTYPE html&gt;

&lt;html&gt;

&lt;head&gt;

&lt;title&gt;Password Input Control&lt;/title&gt;

&lt;/head&gt;

&lt;body&gt;

&lt;form &gt;

Password: &lt;input type = "password" name = "password" /&gt;

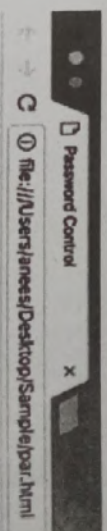
&lt;br&gt;

&lt;/form&gt;

&lt;/body&gt;

&lt;/html&gt;

This will produce the following result.

Password: 

The attributes for <input> tag for creating password field is same as for creating text field.

### Multiple-Line Text Input Controls

Textarea is a multiple-line text input control that allows a user to enter more than one line of text. Multi-line text input controls are created using an <textarea> element. Here is a basic example of a multi-line text input used to take item description.

&lt;!DOCTYPE html&gt;

&lt;html&gt;

&lt;head&gt;

input  
textarea  
password  
checkbox  
radio

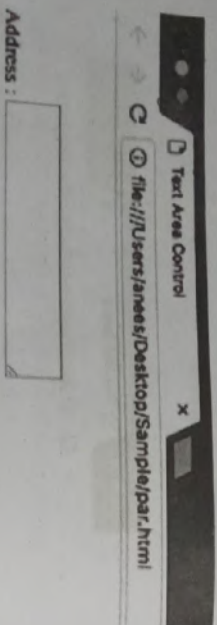


```

<title>Text area Control</title>
</head>
<body>
<form>
  Address: <textarea rows = "3" cols = "30" name = "address"
  id = "address" ></textarea>
<br>
</form>
</body>
</html>

```

This will produce the following result.



Following is the list of attributes for <textarea> tag.

Attribute	Description
name	Used to give a name to the control which is sent to the server to be recognized and get the value.
rows	Indicates the number of rows of text area box.
cols	Indicates the number of columns of text area box.

#### Checkboxes

Checkboxes allows the user to select one or more option from a pre-defined set of options. It is created using an <input> element whose type attribute has a value of checkbox.

```

<!DOCTYPE html>
<html>
<head>
<title>Checkbox Control</title>
</head>
<body>
<form>
  <input type="checkbox" name="sports" id="sports">
  <label for="sports">Sports</label>
  <input type="checkbox" name="cricket" id="cricket">
  <label for="cricket">Cricket</label>
  <input type="checkbox" name="baseball" id="baseball">
  <label for="baseball">Baseball</label>

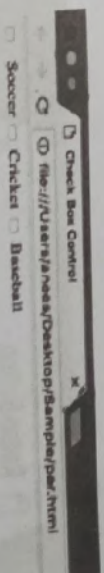
```

```

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

```

This will produce the following result.



Following is the list of attributes for <checkbox> tag.

Attribute	Description
type	Indicates the type of input control and for checkbox input control it will be set to checkbox..
name	Used to give a name to the control which is sent to the server to be recognized and get the value.
value	The value that will be used if the checkbox is selected.
checked	Set to checked if you want to select it by default.

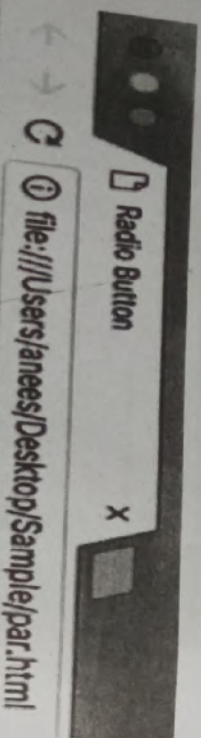


**Radio Buttons**

Radio buttons are used to let the user select exactly one option from a pre-defined set of options. It is created using an `<input>` element whose type attribute has a value of `radio`.

```
<!DOCTYPE html>
<html>
<head>
<title>Radio Button</title>
</head>
<body>
<form>
  <input type="radio" name="sex" id="male">
  <label for="male">Male</label>
  <input type="radio" name="sex" id="female">
  <label for="female">Female</label>
</form>
</body>
</html>
```

This will produce the following result.



☒ Male ☐ Female

Following is the list of attributes for `<radio>` tag.

Attribute	Description
type	Indicates the type of input control and for checkbox input control it will be set to <code>radio</code> .
name	Used to give a name to the control which is sent to the server to be recognized and get the value.
value	The value that will be used if the checkbox is selected.
checked	Set to <i>checked</i> if you want to select it by default.

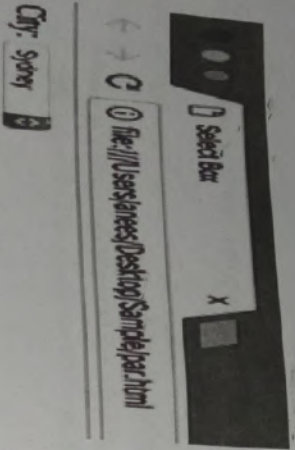
**Select Boxes**

A select box is a dropdown list of options that allows user to select one or more option from a pull-down list of options. Select box is created using the `<select>` element and `<option>` element. The option elements within the `<select>` element define each list item.

```
<!DOCTYPE html>
<html>
<head>
<title>Select Box</title>
</head>
<body>
<form>
  <label for="city">City:</label>
  <select name="city" id="city">
    <option value="sydney">Sydney</option>
    <option value="melbourne">Melbourne</option>
    <option value="cromwell">Cromwell</option>
  </select>
</form>
</body>
</html>
```

This will produce the following result.





Following is the list of important attributes of <select> tag

Attribute	Description
<b>name</b>	Used to give a name to the control which is sent to the server to be recognized and get the value.
<b>Size</b>	This can be used to present a scrolling list box.
<b>multiple</b>	If set to "multiple" then allows a user to select multiple items from the menu.

Following is the list of important attributes of <option> tag

Attribute	Description
<b>value</b>	The value that will be used if an option in the select box box is selected.
<b>selected</b>	Specifies that this option should be the initially selected value when the page loads.

**label** An alternative way of labeling options

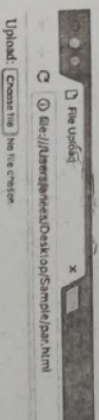
### File Upload box

The file fields allow a user to browse for a local file and send it as an attachment to the form data. It normally rendered as a text box with a button that enables the user to browse for a file. However, the user can also type the path and name of the file in the

text box. This is also created using an <input> element, whose type attribute value is set to file.

```
<!DOCTYPE html>
<html>
<head>
<title>File Upload</title>
</head>
<body>
<form >
  <label for="file-select">Upload:</label>
  <input type="file" name="upload" id="file-select">
</form>
</body>
</html>
```

This will produce the following result.



Following is the list of important attributes of file upload box

Attribute	Description
<b>name</b>	Used to give a name to the control which is sent to the server to be recognized and get the value.
<b>accept</b>	Specifies the types of files that the server accepts.



Button Controls

There are various ways in HTML to create clickable buttons. You can also create a clickable button using <input>tag by setting its type attribute to **button**. The type attribute can take the following values

Type	Description
<b>submit</b>	This creates a button that automatically submits a form.
<b>reset</b>	This creates a button that automatically resets form controls to their initial values.
<b>button</b>	This creates a button that is used to trigger a client-side script when the user clicks that button.
<b>image</b>	This creates a clickable button but we can use an image as background of the button.

```

<!DOCTYPE html>
<html>
<head>
<title>Button Example</title>
</head>

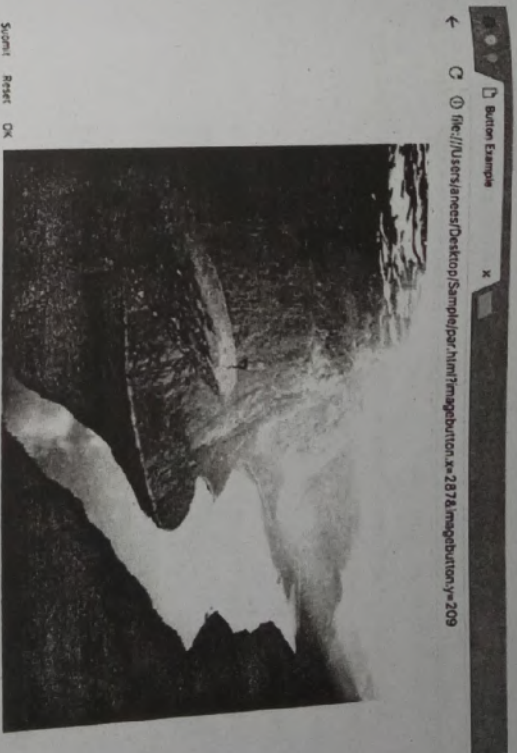
```

```

<body>
<form>
<input type = "submit" name = "submit" value = "Submit" />
<input type = "reset" name = "reset" value = "Reset" />
<input type = "button" name = "ok" value = "OK" />
<input type = "image" name = "imagebutton" src = "image.jpg" />
</form>
</body>
</html>

```

This will produce the following result

Hidden Form Controls

Hidden form controls are used to hide data inside the page which later on can be pushed to the server. This control hides inside the code and does not appear on the actual page. For example, following hidden form is being used to keep current page number. When a user will click next page then the value of hidden control will be sent to the web server and there it will decide which page will be displayed next based on the passed current page.

A Sample page which contain most of the above discussed fields is given below.

```

<!DOCTYPE html>
<html>
<head>
<title>Example</title>
</head>

```



```

<body>
<form>
  User name: <input type = "text" name = "username" />
  <br><br> Password: <input type = "password" name =
  "password" /> <br><br>
  Address: <textarea rows = "3" cols = "30" name = "address"
  id = "address" ></textarea>
<br><br>

```

Hobbies :

```

<input type="checkbox" name="sports" id="soccer">
<label for="soccer">Soccer</label>
<input type="checkbox" name="sports" id="cricket">
<label for="cricket">Cricket</label>
<input type="checkbox" name="sports" id="baseball">
<label for="baseball">Baseball</label>
<br><br>
Sex : <input type="radio" name="sex" id="male">
<label for="male">Male</label>
<input type="radio" name="sex" id="female">
<label for="female">Female</label>
<br><br>
<label for="city">City:</label>
<select name="city" id="city">
  <option value="sydney">Sydney</option>
  <option value="melbourne">Melbourne</option>
  <option value="cromwell">Cromwell</option>
</select>
<br><br>
<label for="file-select">Upload:</label>
<input type="file" name="upload" id="file-select">

```

```

<br><br>
<input type = "submit" name = "submit" value = "Submit" />
<input type = "reset" name = "reset" value = "Reset" />
<input type = "button" name = "Ok" value = "Ok" />
</form>
</body>
</html>

```

This will produce the following result.

### HTML Frames

HTML frames are used to divide your browser window into multiple sections where each section can load a separate HTML document. A collection of frames in the browser window is known as a frameset. The window is divided into frames in a similar way the tables are organized: into rows and columns.

To use frames on a page we use <frameset> tag instead of <body> tag. The <frameset> tag defines, how to divide the window into frames. The rows attribute of <frameset> tag defines horizontal frames and cols attribute defines vertical frames. Each frame is indicated



scrolling

This attribute controls the appearance of the scrollbars that appear on the frame. This takes values either "yes", "no" or "auto".

longdesc

This attribute allows you to provide a link to another page containing a long description of the contents of the frame.

## Cascading Style Sheets CSS

Cascading Style Sheets (CSS) is a style language that defines layout of HTML documents. For example, CSS covers fonts, colours, margins, lines, height, width, background images, advanced positions and many other things. HTML can be used to add layout to websites, but CSS offers more options and is more accurate and sophisticated. CSS is supported by all browsers today. CSS was invented by Hakon Wium Lie on October 10, 1994 and maintained through a group of people within the W3C called the CSS Working Group (CSS1) was 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 was became a W3C recommendation in May 1998 and builds on CSS1. This version adds support for media-specific style sheets. CSS3 was became a W3C recommendation in June 1999 and builds on older versions CSS. It has divided into documentations is called as Modules and here each module having new extension features defined in CSS2.

The main difference between CSS and HTML is that HTML is used to structure content, but CSS is used for formatting structured content. The main benefits of using CSS include:

- **CSS saves time** " You can write CSS once and then reuse 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, 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 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 cell phones 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 to start using CSS in all the HTML pages to make them compatible to future browsers.
- **Offline Browsing** " CSS can store web applications locally with the help of an offline cache.
- **Platform Independence** " The Script offer consistent platform independence and can support latest browsers as well.

## CSS - Syntax

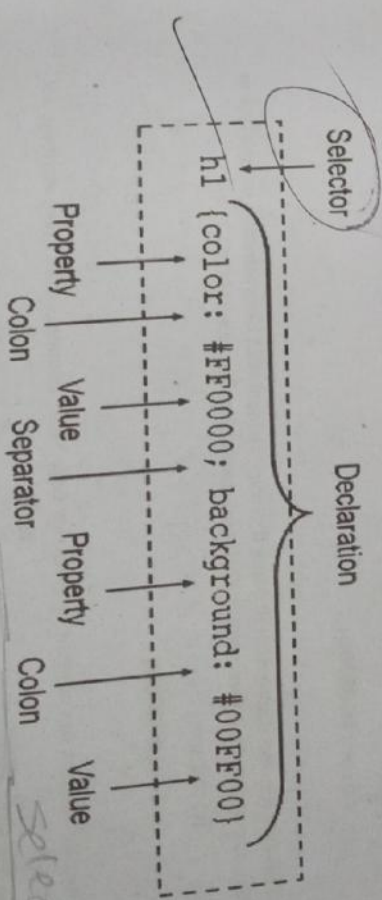
CSS consists of two building blocks:

- **Properties:** Human-readable identifiers that indicate which stylistic features (e.g. font, width, background color) you want to change.
- **Values:** Each specified property is given a value, which indicates how you want to change those stylistic features (e.g. what you want to change the font, width or background color to.)

A property paired with a value is called a CSS declaration. CSS declarations are put within CSS Declaration Blocks. And finally, CSS declaration blocks are paired with selectors to produce CSS Ruleset.

You can put CSS Style Rule Syntax as follows

selector {property: value }



You can define a heading h1 as follows

which are the elements that have the CSS property values.

1. e.g. a pattern of elements, e.g. colors, text, that render browser to have the CSS property values.



```
h1{color: #FF0000; background: #00FF00 }
```

Here h1 is a selector and color is the property and given value #FF0000, in the same way, background is the property and given value #00FF00

### The Type Selectors

This is the same selector we have seen above. The CSS type selector matches elements by node name. In other words, it selects all elements of the given type within a document.

```
table{ border: 1px solid #C00; }
```

This is another example

```
table{ border: 1px solid #C00; }
span {
background-color: skyblue;
}
```

### The Class selector

The CSS class selector matches elements based on the contents of their class attribute. 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.

The syntax for defining class selector is

```
.class_name{ style properties }
```

An example is given below

```
.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.

### The ID Selectors

In an HTML document, the CSS ID selector matches an element based on the value of its id attribute. The selected element's ID attribute must match exactly the value given in the selector. The syntax of id selector is given below.

```
#id_value{ style properties }
```

The following rule renders the content in black for every element with id attribute set to black in our document.

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

### 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 Attribute Selector

The CSS attribute selector matches elements based on the presence or value of a given attribute. 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.

Some example are given below.

```
/* <a> elements with a title attribute */
a[title] {
color: purple;
}
```

```
/* <a> elements with an href matching "https://example.org" */
a[href="https://example.org"] {
```



```

color: green;
}

/* <a> elements with an href containing "example" */
a[href*="example"] {
  font-size: 2em;
}

/* <a> elements with an href ending ".org" */
a[href$=".org"] {
  font-style: italic;
}

```

### CREATING STYLE SHEETS

CSS provide easy and effective alternatives to specify various attributes for the HTML tags. Using CSS, you can specify a number of style properties for a given HTML element. Each property has a name and a value, separated by a colon (:). Each property declaration is separated by a semi-colon (;).

Let's consider an example of HTML document which makes use of <font> tag

```

<!DOCTYPE html>
<html>
<head>
<title>HTML CSS</title>
</head>
<body>
<p><font color = "red" size = "7">Hello, World!</font></p>
</body>
</html>

```

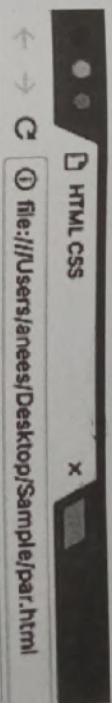
We can re-write above example with the help of Style Sheet as follows.

```

<!DOCTYPE html>
<html>
<head>
<title>HTML CSS</title>
</head>
<body>
<p style = "color:red; font-size:28px;" >Hello, World!</p></body>
</html>

```

Both will produce the same result as follows.



# Hello, World!

There are three ways of inserting a style sheet:

- External style sheet
- Internal style sheet
- Inline style

#### External style sheet

External style sheets can define style sheet rules in a separate .css file and then include that file in your HTML document using HTML <link> tag.

Consider we define a style sheet file mystyle.css which has following rules



```

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

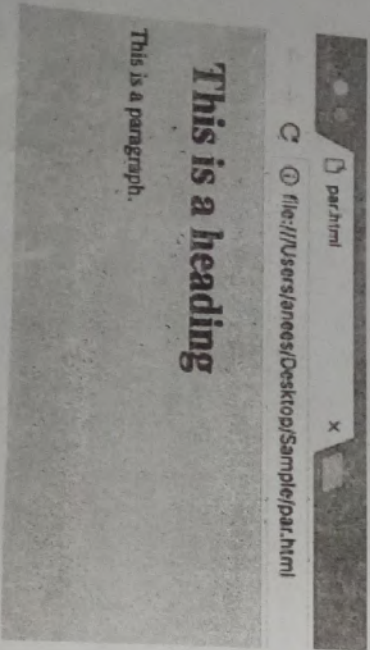
```

Use the above external CSS file in our following HTML document.

```

<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" type="text/css"
        href="mystyle.css">
</head>
<body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
</body>
</html>

```



### Internal Style Sheet

An internal style sheet may be used if one single page has a unique style. Internal styles are defined within the `<style>` element, inside the `<head>` section of an HTML page. Let's re-write above example using Internal style sheet.

```

<!DOCTYPE html>
<html>
<head>
    <style>
        body {
            background-color: lightblue;
        }
        h1 {
            color: navy;
            margin-left: 20px;
        }
    </style>
</head>
<body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
</body>
</html>

```

This will also produce the same result

### Inline style

An inline style may be used to apply a unique style for a single element. To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property. The example below shows how to change the color and the left margin of a `<h1>` element:



```

<!DOCTYPE html>
<html>
  <body>
    <h1 style="color:blue;margin-left:30px;">This is a
    heading</h1>
    <p>This is a paragraph.</p>
  </body>
</html>

```

### CSS Background

CSS background property is used to define the background effects on element. There are 3 CSS background properties that affects the HTML elements:

1. background-color
2. background-image
3. background-repeat
4. background-attachment
5. background-position

### Set the Background Color

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

```

<!DOCTYPE html>
<html>
  <head>
    <style>
      h2,p{ background-color: #ff0000; }
    </style>
  </head>
  <body>
    <h2>CSS Background Example</h2>

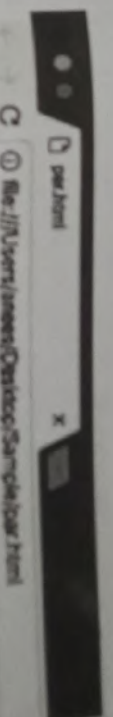
```

```

<p>This is an example of CSS background-color.</p>
</html>

```

It will produce the following output



### CSS background Example

This is an example of CSS background-color.

### Set the Background Image

We can set the background image by calling local stored images as shown below

```

<!DOCTYPE html>
<html>
  <head>
    <style>
      body { background-image: url("image.jpg");
        margin-left:100px;}
    </style>
  </head>
  <body>
    <h1>Hello Students</h1>
  </body>
</html>

```



It will produce the following output.



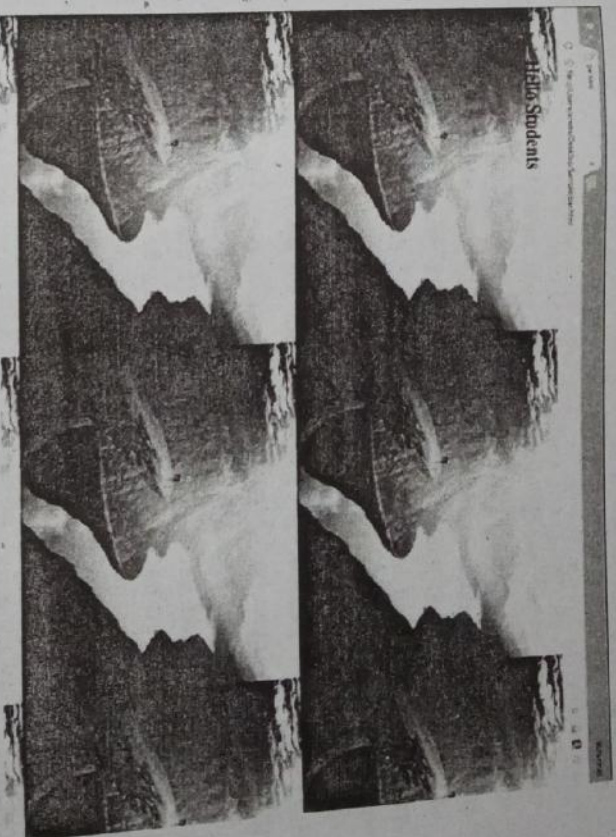
### Repeat the Background Image

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

```
<!DOCTYPE html>
<html>
<head>
<style>
  body { background-image: url("image.jpg");
        background-repeat: repeat;
        margin-left: 100px;}
</style>
```

```
</head>
<body>
  <h1>Hello Students</h1>
</body>
</html>
```

This will produce the following output.

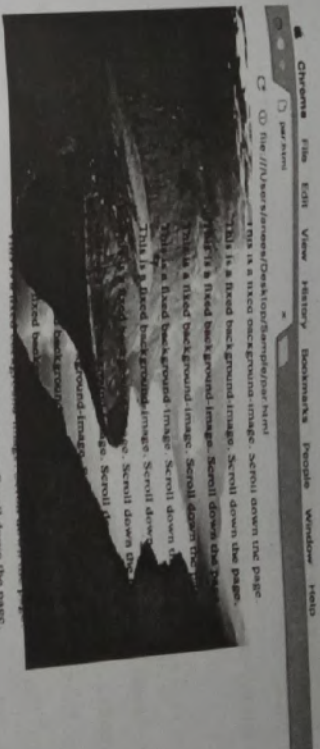


### Set the Background Attachment

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.



```
<!DOCTYPE html>
<html>
<head>
<style>
body {
    background: white url(image.jpg);
    background-repeat: no-repeat;
    background-attachment: fixed;
    margin-left: 200px;
}
</style>
</head>
<body>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>If you do not see any scrollbars, Resize the browser window.</p>
</body>
</html>
```



### Set the Background Image Position

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

1. center
2. top
3. bottom
4. left
5. right

### Example

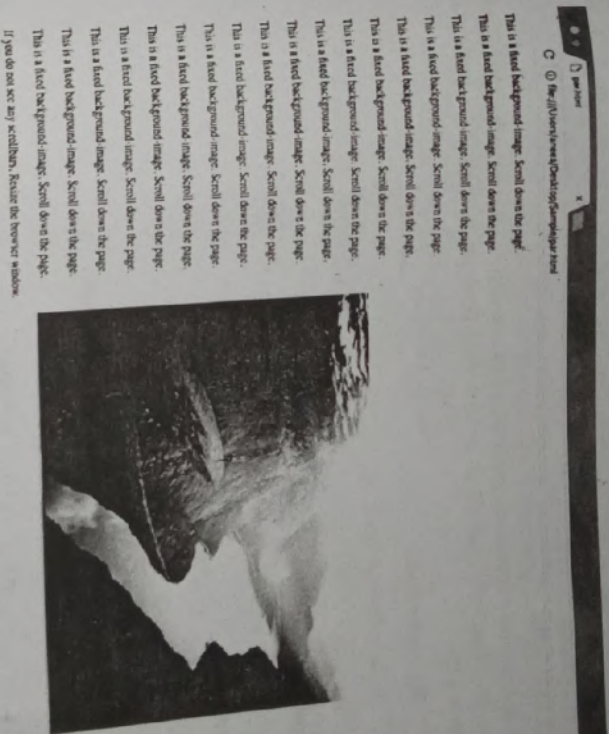
```
<!DOCTYPE html>
<html>
<head>
<style>
body {
```



```
background: white url('image.jpg'),
background-repeat: no-repeat;
background-attachment: fixed;
background-position: center;
}
```

```
</style>
</head>
<body>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>This is a fixed background-image. Scroll down the page.</p>
<p>If you do not see any scrollbars, Resize the browser window.</p>
</body>
```

This will result the following output.



## CSS - Fonts

CSS Font property is used to control the look of texts. By the use of CSS font property you can change the text size, color, style and more. You have already studied how to make text bold or underlined. Here, you will also know how to resize your font using percentage.

These are some important font attributes:

1. CSS Font color: This property is used to change the color of the text.
2. CSS Font family: This property is used to change the face of the font.
3. CSS Font size: This property is used to increase or decrease the size of the font.
4. CSS Font style: This property is used to make the font bold, italic or oblique.



5. CSS Font variant: This property creates a small-caps effect.
6. CSS Font weight: This property is used to increase or decrease the boldness and lightness of the font.

### CSS Font Color

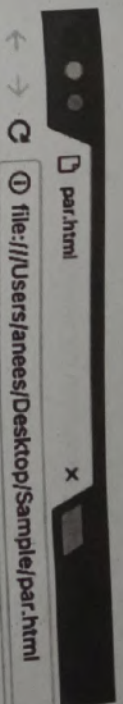
CSS font color is a standalone attribute in CSS although it seems that it is a part of CSS fonts. It is used to change the color of the text. There are three different formats to define a color:

1. By a color name
2. By hexadecimal value
3. By RGB

#### Example

```
<!DOCTYPE html>
<html>
<head>
<style>
  body { font-size: 100%; }
  h1 { color: red; }
  h2 { color: #9000A1; }
  p { color: rgb(0, 220, 98); }
</style>
</head>
<body>
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<p>This is a paragraph.</p>
</body>
</html>
```

This will result the following output.



## This is heading 1

## This is heading 2

This is a paragraph.

### CSS Font Family

CSS font family can be divided in two types: Generic family: It includes Serif, Sans-serif, and Monospace. Font family: It specifies the font family name like Arial, New Times Roman etc.

#### Example

```
<!DOCTYPE html>
<html>
<head>
<style>
  body {
    font-size: 100%; }
  h1 { font-family: sans-serif; }
  h2 { font-family: serif; }
  p { font-family: monospace; }
</style>
</head>
<body>
```



**CSS - Text**

You can set following text properties of an element.

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

An Example is given below.

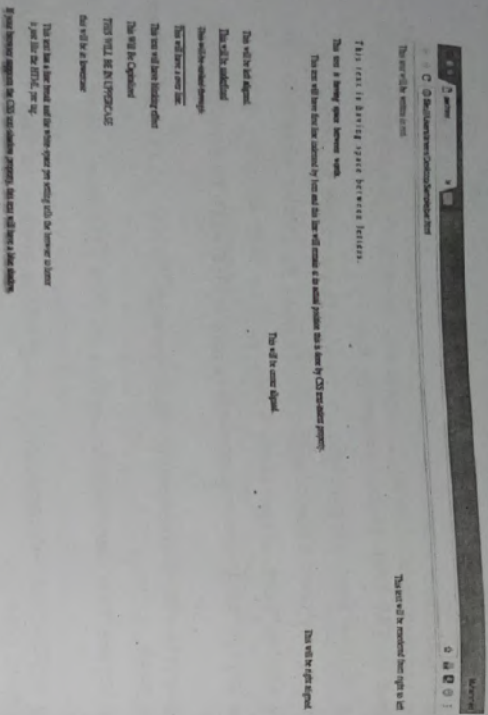
```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
<p style="color:red;">This text will be written in red. </p>
```

```
<p style="direction:rtl;">This text will be rendered from right to left</p>
<p style="letter-spacing:5px;">This text is having space between letters.</p>
<p style="word-spacing:5px;">This text is having space between words.</p>
<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>
<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>
<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>
<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>
<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>
<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>
</body>
```



```
</html>
```

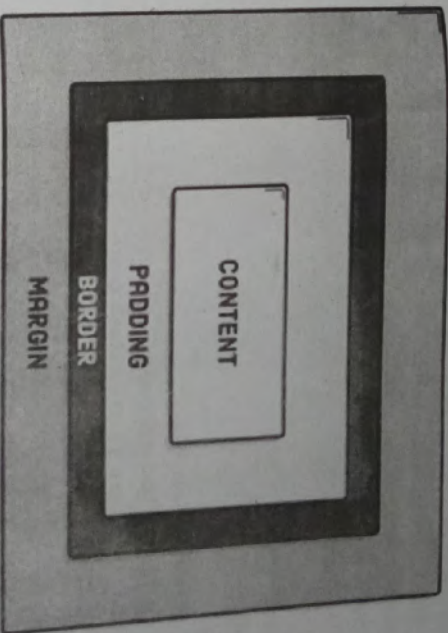
This will result the following output.



Text Format, Controlling Fonts - Working with Block Elements and Objects, CSS ID and Class.

### 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 every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:



A block-level element always starts on a new line and takes up the full width available. The <div> element is a block-level element. When used together with CSS, the <div> element can be used to style blocks of content. The box model allows us to add a border around elements, and to define space between elements. The Explanation of the different parts is given below

- Content - The content of the box, where text and images appear
- Padding - Clears an area around the content. The padding is transparent
- Border - A border that goes around the padding and content
- Margin - Clears an area outside the border. The margin is transparent

An Example is given below.

```
<!DOCTYPE html>
<html>
<head>
```