



# UI TECHNOLOGIES



# **INDEX**

<b>1)</b>	<b>HTML .....</b>	<b>3</b>
<b>2)</b>	<b>CSS .....</b>	<b>23</b>
<b>3)</b>	<b>Bootstrap .....</b>	<b>60</b>
<b>4)</b>	<b>JavaScript .....</b>	<b>84</b>
<b>5)</b>	<b>DOM .....</b>	<b>123</b>
<b>6)</b>	<b>jQuery .....</b>	<b>140</b>



# HTML

## Study Material



# HTML (HYPER TEXT MARKUP LANGUAGE)

## Agenda

- 1) HTML
- 2) JS
- 3) CSS
- 4) Bootstrap
- 5) JQuery
- 6) Python
- 7) Django
- 8) SQLite | Oracle | MySQL

## Web Application:

The applications which will provide services over the web are called web applications.

Eg: gmail.com, facebook.com, durgasoftvideos.com et

Every web application contains 2 main components

- 1) Front-End
- 2) Back-End

### 1) Front-End:

- It represents what user is seeing on the website
- We can develop Front-End content by using the following technologies:
- HTML, JS, CSS, JQuery and Bootstrap
- JQuery and Bootstrap are advanced front-end technologies, which are developed by using HTML, CSS and JavaScript only.

### HTML: HyperText Markup Language

Every web application should contain HTML. i.e HTML is the mandatory technology for web development. It represents structure of web page



## CSS: Cascading Style Sheets

- It is optional technology; still every web application contains CSS.
- The main objective of CSS is to add styles to the HTML Pages like colors, fonts, borders etc.

## Java Script:

- It allows to add interactivity to the web application including programming logic.
- The main objective of Java Script is to add functionality to the HTML Pages. ie to add dynamic nature to the HTML Pages.
- HTML → Meant for Static Responses
- HTML+JS → Meant for Dynamic Responses

Eg 1: To display "Welcome to DURGASOFT" response to the end user only HTML is enough, because it is static response.

Eg 2: To display current server date and time to the end user, only HTML is not enough we required to use some extra technology like JavaScript, JSP, ASP, PHP etc as it is dynamic response.

## Static Response vs Dynamic Response:

- If the response is not varied from time to time and person to person then it is considered as static response.  
Eg: login page of gmail  
home page of icici bank
- If the response is varied from time to time and person to person then it is considered as dynamic response.  
Eg: inbox page of gmail  
balance page of icicibank

## 2) Back End:

- It is the technology used to decide what to show to the end user on the Front-End.
- ie Backend is responsible to generate required response to the end user, which is displayed by the Front-End.

Back-End has 3 important components:

- 1) The Language like Java, Python etc
- 2) The Framework like Django, Pyramid, Flask etc
- 3) The Database like SQLite, Oracle, MySQL etc



- For the Backend language Python is the best choice because of the following reasons: Simple and easy to learn, libraries and concise code.
- For the Framework Django is the best choice because it is Fast, Secure and Scalable. Django is the most popular web application framework for Python.
- Django provides inbuilt database which is nothing but SQLite, which is the best choice for database.
- The following are various popular web applications which are developed by using Python and Django.
- YouTube, Dropbox, Quora, Instagram, Reddit, Yahoo Maps etc

## HTML Basics:

- HTML stands for HyperText Markup language.
- This is the most basic building block of every web application. Without using HTML we cannot build web applications. It is the mandatory technology.
- We can use CSS to style HTML Pages.
- We can use Java Script to add functionality to the HTML pages.
- In general we will add django template tags to HTML for generating dynamic content based on our requirement.

## Structure of HTML Page:

- Every HTML page contains 2 parts
  - 1) Head
  - 2) Body
- Head contains meta data like title of the page, keywords etc. CSS files and Java Script files information we have to specify in the Head Part only.
- Body contains actual content.

- 1) `<!doctype html> // to indicate that it is HTML page`
- 2) `<html >`
- 3) `<head>`
- 4) Meta Data like keywords, author, title...
- 5) css files information
- 6) js files information
- 7) `</head>`
- 8) `<body>`
- 9) Actual Data
- 10) `</body>`
- 11) `</html>`



## HTML Comment:

`<!-- Anything here is considered as Comment -->`

## Title:

- 1) `<head>`
- 2) `<title>Basic HTML for Django Classes</title>`
- 3) `</head>`

## Heading Tags:

HTML supports 6 heading tags

`<h1>`, `<h2>`, ....

`<h1>This is Heading1</h1>`

## Paragraph tag: `<p>`:

We can use this tag to represent paragraph of text.

`<p> This is first paragraph </p>`

### Case-1:

- 1) `<p>This is First Line`
- 2) `This is Second Line`
- 3) `This is Third Line`
- 4) `This is Four Line`
- 5) `</p>`

Total Data will come in a single line, because we are using only one paragraph tag.

### Case-2:

- 1) `<p>This is First Line</p>`
- 2) `<p>This is Second Line</p>`
- 3) `<p>This is Third Line</p>`
- 4) `<p>This is Fourth Line</p>`

Output will come in 4 lines

### Case-3:

`<p>This is First Line</p><p>This is Second Line</p>`

output will come in multiple lines



**Note:** In HTML indentation is not important but tags are important. Blocking also takes care by html tags only.

```
1) <body>
2)   <h1>This is HTML Demo Class</h1>
3)       <p>This is First Line</p>
4)   <p>This is Second Line</p>
5)       <p>This is Third Line</p>
6)   <p>This is Fourth Line</p>
7) </body>
```

## Bold and Italic:

### Legacy Tags:

<b> for bold

<i> for italic

These are old (IEgacy)html tags and not recommended to use.

**Eg:** <p><b><i>This is First Line</i></b></p>

### Advanced Tags:

We can use the following HTML 5 advanced tags for bold and italic

<strong> for strong text(bold)

<em> for emphasis (italic)

**Eg:** <p><strong><em>This is Second Line</em></strong></p>

## HTML Lists:

There are 2 types of lists

- 1) Ordered List
- 2) Unordered List

### 1) Ordered List:

All list items will be displayed with numbers

```
1) <ol>
2)   <li>Chicken</li>
3)   <li>Mutton</li>
4)   <li>Fish</li>
5)   <li>Beer</li>
6) </ol>
```





## Output:

1. Chicken
2. Mutton
3. Fish
4. Beer

## 2) Unordered List:

Instead of numbers bullet symbol will come. Here order is not important.

```
1) <ul>
2)   <li>Chicken</li>
3)   <li>Mutton</li>
4)   <li>Fish</li>
5)   <li>Beer</li>
6) </ul>
```

## Nested Lists:

We can take list inside list, which are nothing but nested lists.

```
1) <ol>
2)   <li>Chicken</li>
3)   <li>Mutton</li>
4)   <li>Fish</li>
5)   <li>Beer</li>
6)   <ul>
7)     <li>KF</li>
8)     <li>KO</li>
9)     <li>RC</li>
10)  </ul>
11) </ol>
```

**Note:** For outer and inner lists we can take both ordered and unordered lists

**Eg:** We can take unordered list inside ordered list

## Div and Span Tags:

- We can use div and span tags to group related tags into a single unit.
- In general we can use these tags with CSS.

```
1) <div class="groupone">
2)   <h1> Group One Content</h1>
3)   <p>This division tag helpful to style a group of
4)     html tags with css</p>
5) </div>
```



- div means division
- `<span>` tag is exactly same as division tag except that it is inline. i.e to define group within the line of text we can use `<span>` tag.
- `<p>This <span>division tag helpful</span> to style a group of html tags with css</p>`
- `<div>` will work for group of lines where as `<span>` will work within the line.

**Note:** `<div>` and `<span>` tags are helpful only for styling html. Hence they will always work with css only.

## Attributes:

- HTML Attributes will provide extra information to HTML tags.
- To insert image in the html page, src attribute specify location of the image to the `<img>` tag.

## Setting Image inside HTML:

- ``
- src means source where we have to specify the image source(complete location). We can take image address from the google also.
- alt means alternate. If image is missing then broken link image will display. In this case if we want to display some meaningful text information then we should go for alt attribute.

**Note:** We have to open the tag and we are not responsible to close the tag, such type of tags are called self closing tags.

**Eg:** `<img>` tag

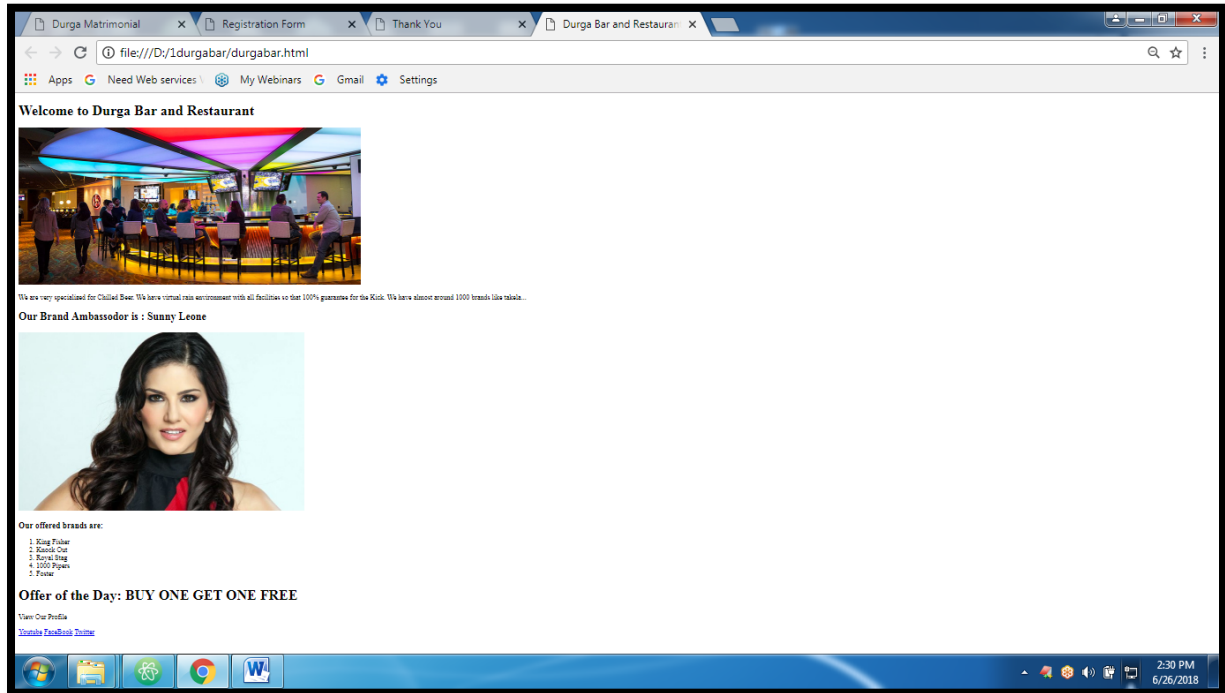
## Creating Hyperlinks by using anchor tag: `<a>`:

`<a href="second.html">Click Here to go Second Page</a>`

`<a href="https://facebook.com">FaceBook</a>`



## Durga Bar and Restaurant Application:



- 1) `<!doctype html>`
- 2) `<html lang="en">`
- 3) `<head>`
- 4) `<meta charset="UTF-8">`
- 5) `<meta name="Generator" content="EditPlus@">`
- 6) `<meta name="Author" content="">`
- 7) `<meta name="Keywords" content="">`
- 8) `<meta name="Description" content="">`
- 9) `<title>Durga Bar and Restaurant</title>`
- 10) `</head>`
- 11) `<body>`
- 12) `<h1>Welcome to Durga Bar and Restaurant</h1>`
- 13) ``
- 14) `<p>We are very specialized for Chilled Beer. We have virtual rain environment with all facilities so that 100% guarantee for the Kick</p>`
- 15) `<h2>Our Brand Ambassador is : Sunny Leone</h2>`
- 16) ``
- 17) `<h3>Our offered Products:</h3>`
- 18) `<ol>`
- 19) `<li>KF</li>`
- 20) `<li>KO</li>`
- 21) `<li>RC</li>`
- 22) `<li>FO</li>`



```
23) </ol>
24) <h1>Offer of the Day: Buy One Get One FREE</h1>
25) <p>View Our Profile:</p>
26) <a href="http://youtube.com">YouTube</a>
27) <a href="http://twitter.com">Twitter</a>
28) <a href="https://facebook.com">Facebook</a>
29) </body>
30) </html>
```

## Table Creation:

We can use

- <table> to create table
- <thead> to specify head row
- <th> to specify column data in head row(column name)
- <tr> to insert row in the table
- <td> to specify column data in the row/record




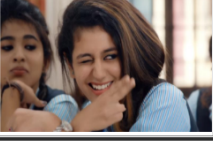

We can use border attribute inside <table> tag

```
1) <table border="1">
2)   <thead>
3)     <th>ENO</th>
4)     <th>ENAME</th>
5)     <th>ESAL</th>
6)     <th>EADDR</th>
7)   </thead>
8)   <tr>
9)     <td>100</td>
10)    <td>DURGA</td>
11)    <td>1000</td>
12)    <td>Hyd</td>
13)  </tr>
14)  <tr>
15)    <td>200</td>
16)    <td>SUNNY</td>
17)    <td>2000</td>
18)    <td>Mumbai</td>
19)  </tr>
20)  <tr>
21)    <td>300</td>
22)    <td>Bunny</td>
23)    <td>3000</td>
24)    <td>Chennai</td>
25)  </tr>
```



26) `</table>`

## Oscar Awards Winners List 2018 Application (Assignment on tables):

S.No	Winner	Winner Pic	Category	Country Flag	View Profile
1	Sunny Leone		Best Actress		<a href="#">Click Here</a>
2	Deepika		Best Actress		<a href="#">Click Here</a>
3	Priya		Best Actress		<a href="#">Click Here</a>

- 1) `<!doctype html>`
- 2) `<html lang="en">`
- 3) `<head>`
- 4) `<title>Oscar awards winners list</title>`
- 5) `</head>`
- 6) `<body>`
- 7) `<h1> Oscar Awards Winners List 2018</h1>`
- 8) `<table border="1">`
- 9) `<thead>`
- 10) `<th>S.No</th>`
- 11) `<th>Winner</th>`
- 12) `<th>Winner Pic</th>`
- 13) `<th>CatEgory</th>`
- 14) `<th>Country Log</th>`
- 15) `<th>View Profile</th>`
- 16) `</thead>`
- 17) `<tr>`
- 18) `<td>1</td>`
- 19) `<td>Sunny</td>`



```
20) <td></td>
21) <td>Best Actress</td>
22) <td></td>
23) <td><a href="http://youtube.com/durgasoftware">View Info</a></td>
24) </tr>
25) </table>
26) </body>
27) </html>
```

## Creation of HTML Forms:

- As the part of web application development, we have to develop several forms like login form, registration form etc
- We can create Html form by using <form> tag.

```
<form class="" action="" method="">
.....
</form>
```

- Within the form to collect end user input, we have to use <input> tag.
- This <input> tag will play very important role in form creation.

**Syntax:** <input type="" name="" value=""/>

- type attribute can be used to specify the type of input end user has to provide. The main important types are:
  - text
  - email
  - password
  - color
  - submit
  - checkbox
  - radio
  - etc
- name attribute represents the name of input tag. By using this name, in the next target page we can access end user provided input value.
- value attribute represents default value will be displayed in the form.

**Eg:**

```
<input type="text" name="username" value="Enter User Name"/>
<input type="email" name="mailid" value=""/>
<input type="password" name="pwd" value=""/>
<input type="checkbox" name="course" value=""/>
<input type="radio" name="married" value=""/>
```



- To provide default value it is highly recommended to use placeholder attribute because end user is not required to delete default value while entering data.  
Eg: `<input type="text" name="username" placeholder="Enter User Name"/>`

## Creation of Labels for HTML Elements:

We can define Label Text for our HTML Elements like Radio Buttons, Text Box etc by using `<label>` tag.

Syntax: `<label for="name">Enter Name:</label>`

Eg: `<p>Enter Name:</p>`  
`<input type="text" name="username" placeholder="Enter Name">`

The result looks like

In this case there is no relation between text box and data.

To link data to text box, we have to use `<label>` tag.

`<label for="name">Enter Name:</label>`  
`<input id="name" type='text' name='username' placeholder='Name to Contact' >`

The result looks like

## required Attribute:

If end user compulsory should required to provide input value then we should go for required attribute.

Eg:

`<input id="name" type='text' name='username' placeholder='Name to Contact' required>`

## action Attribute:

Once we fill the form and click submit, then to which page it will go is decided by action attribute. The value of action attribute can be either local resource or web url.

Eg:

`<form action="target.html" >`  
`<form action="https://facebook.com" >`



## Demo Program:

### test.html

```
1) <!doctype html>
2) <html lang="en">
3) <head>
4) <title>Form Input</title>
5) </head>
6) <body>
7) <h1>User Contact Form</h1>
8) <form action="target.html">
9) <label for="name">Enter Name:</label>
10) <input id="name" type='text' name='username' placeholder='Name to Contact' r
    equired>
11) <input type="submit" value="ClickToContact">
12) </form>
13) </body>
14) </html>
```

### target.html:

```
1) <!doctype html>
2) <html lang="en">
3) <head>
4) <title>Target HTML</title>
5) </head>
6) <body>
7) <h1>Thanks for Confirmation</h1>
8) </body>
9) </html>
```

## Implementing Radio Buttons:

```
1) <h3>Are you Married:</h3>
2) <label for='ma'>Yes</label>
3) <input id='ma' type='radio' name="married" value="">
4) <label for='ma1'>No</label>
5) <input id='ma1' type='radio' name="married" value="">
```





## Eg 2:

- 1) `<h3>Gender:</h3>`
- 2) `<label for='m'>Male</label>`
- 3) `<input id='m' type='radio' name="gender" value="Male">`
- 4) `<label for='f'>Female</label>`
- 5) `<input id='f' type='radio' name="gender" value="Female">`
- 6) `<label for='t'>Transgender</label>`
- 7) `<input id='t' type='radio' name="gender" value="Transgender">`

## How to implement Drop down box/select box:

- 1) `<h3>Please Select Your Payment Mode:</h3>`
- 2) `<select name='pmode'>`
- 3) `<option value='ccard'>Credit Card</option>`
- 4) `<option value='dcard'>Dedit Card</option>`
- 5) `<option value='ppal'>Paypal</option>`
- 6) `<option value='otransfer'>OnlineTransfer</option>`
- 7) `</select>`

## textarea element:

- 1) `<h2>Enter Your FeedBack:</h2>`
- 2) `<textarea name="feedback" rows="8" cols="80">`

## checkbox:

- 1) `<form>`
- 2) `<h2>Choose Your Known Languages:</h2>`
- 3) `<input type="checkbox" name="languages" value="english" checked> English<br>`
- 4) `<input type="checkbox" name="languages" value="telugu"> Telugu<br>`
- 5) `<input type="checkbox" name="languages" value="hindi"> Hindi<br>`
- 6) `<input type="checkbox" name="languages" value="tamil"> Tamil<br>`
- 7) `</form>`

## How to include spaces in HTML:

- 1) Type "nbsp" to add a single space.
- 2) Type "ensp" to add 2 spaces.
- 3) Type "emsp" to add 4 spaces.
- 4) Use the non-breaking space (nbsp) 4 times to insert a tab.
- 5) Use "br" to add a line break.

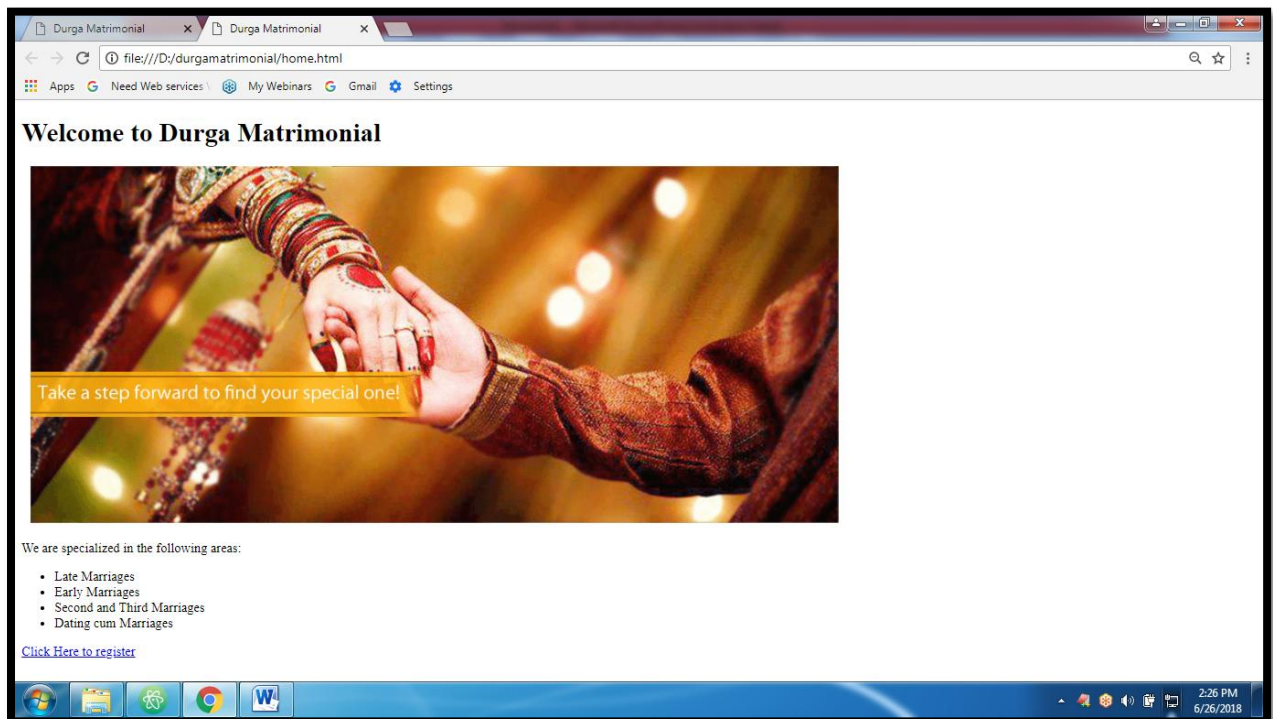


## ATOM IDE/Editor:

<https://atom.io/>

freeware  
open source  
supports cross platform  
several auto completion short-cuts  
etc

## Durga Matrimonial Application:





**Registration Form**

All Fields are Mandatory

Name

Email

Age

Are You Interested In Dating

Yes ☐ No ☐

Your Expections like:

How many marraiges you want:

Are You Heavy Alocoholic:

☐ YES ☐ NO ☐ Only on week-ends

Your Preferences and Extra Information

**Thanks for Registration**

You will get match details soon by email

## home.html:

- 1) `<!DOCTYPE html>`
- 2) `<html lang="en" dir="ltr">`
- 3) `<head>`
- 4) `<meta charset="utf-8">`



```
5) <link rel="stylesheet" href="form.css">
6) <title>Durga Matrimonial</title>
7) </head>
8) <body>
9) <h1>Welcome to Durga Matrimonial</h1>
10) 
11) <p>We are specialized in the following areas:</p>
12) <ul>
13) <li>Late Marriages</li>
14) <li>Early Marriages</li>
15) <li>Second and Third Marriages</li>
16) <li>Dating cum Marriages</li>
17) </ul>
18) <a href="rEgister.html">Click Here to rEgister</a>
19) </body>
20) </html>
```

## register.html:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <meta charset="utf-8">
5) <title>REgistration Form</title>
6) </head>
7) <body>
8) <h1>Registration Form</h1>
9) <h2>All Fields are Mandatory</h2>
10) <form class="" action="thankyou.html" method="post">
11) <label for="name">Name</label>
12) <input id="name" type="text" name="name" placeholder="Enter Name" requir
    ed><br>
13) <label for="mail">Email</label>
14) <input id="mail" type="email" name="mail" placeholder="Enter Email" require
    d><br>
15) <label for="age">Age</label>
16) <input id="age" type="text" name="age" placeholder="Enter Age" required>
17) <h3>Are You Interested In Dating</h3>
18) <label for="dat">Yes</label>
19) <input id="dat" type="radio" name="dating" value="">
20) <label for="dat1">No</label>
21) <input id="dat1" type="radio" name="dating" value="">
22) <h3>Your Expections like:</h3>
23) <select class="" name="expe">
24) <option value="katrina">Katrina Kaif</option>
```



```
25) <option value="kareena">Kareena Kapoor</option>
26) <option value="sunny">Sunny Leone</option>
27) <option value="mallika">Mallika Sherawat</option>
28) </select>
29) <h3>How many marriages you want:</h3>
30) <select class="" name="nom">
31) <option value="One">1</option>
32) <option value="Two">2</option>
33) <option value="Three">3</option>
34) <option value="Four">4</option>
35) </select>
36) <h3>Are You Heavy Alcoholic:</h3>
37) <input type="checkbox" name="Alcoholic" value="yes">YES
38) <input type="checkbox" name="Alcoholic" value="no">NO
39) <input type="checkbox" name="Alcoholic" value="week-end">Only on week-
    ends
40) <h3>Your Preferences and Extra Information</h3>
41) <textarea name="pref" rows="8" cols="80"></textarea>
42) <input type="submit" name="" value="Register">
43) </form>
44) </body>
45) </html>
```

### thankyou.html:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <meta charset="utf-8">
5) <title>Thank You</title>
6) </head>
7) <body>
8) <h1>Thanks for REgistration</h1>
9) <h2>You will get match details soon by email</h2>
10) </body>
11) </html>
```

### form.css:

```
1) h1{
2) color:red;
3) }
4) ul{
5) color:blue;
6) }
```



google drive link: <https://goo.gl/6iq64Y>

## How to declare multiple submit buttons in the form:

Within the form we can define multiple submit buttons. But based on value we can implement corresponding action in the back-end.

```
1) <form action="target">
2) .....
3) <input type="submit" name="action" value="update"/>
4) <input type="submit" name="action" value="edit"/>
5) <input type="submit" name="action" value="delete"/>
6) </form>
```

In the target, based on submit button value we have to implement corresponding action.

```
1) choice = req.getParameter("action")
2) if choice=="update":
3)     perform update related action
4) elif choice=="edit":
5)     perform edit related action
6) elif choice=="delete":
7)     perform delete related action
```

## HTML Validations:

By using HTML, we can perform the following validations

- 1) required
- 2) email
- 3) min and max length

Eg: password should be minimum 5 characters and maximum 10 characters

Eg:

```
<label for = "password">Password:</label>
<input type = "password" name = "password" id = "password" pattern = ".{5,10}"
        placeholder = "your password" required title = "5 to 10 characters">
```

Pattern for only digits:[0-9]{5,10}

Pattern for only word characters:\w{5,10}



# CSS

# Study Material



# CSS (Cascading Style Sheets)

- The main objective of CSS to add styles to HTML. CSS describes how HTML elements are displayed on a page.
- Styling includes colors, fonts, size, borders etc
- We can define CSS styling inside HTML. But it is highly recommended to define styling inside a separate CSS file(.css extension) and link that file to HTML.
- The power of CSS demo link: [https://www.w3schools.com/Css/css\\_intro.asp](https://www.w3schools.com/Css/css_intro.asp)

## Various ways to define Style for HTML Elements:

We can define style in 3 ways

- 1) In Line
- 2) By using style tag
- 3) By using css file

### 1) In Line:

- `<h1 style="color:red">This is My First Part of Data</h1>`
- define style at tag level is not recommended b'z it increases complexity as every html page contains 1000s of tags

### 2) By using Style Tag:

```
1) <html >
2) <head>
3) ...
4) <style type="text/css">
5)   h1{
6)     color:blue;
7)   }
8) </style>
9) </head>
10) ...
11) </html>
```

This way of defining style is not recommended because it is applicable only for current page but not for remaining pages.





### 3) By using CSS File:

#### style1.css

```
1) h1{
2)  color:red;
3) }
4)
5) <head>
6)  <link rel="stylesheet" href="style1.css">
7) </head>
```

We can reuse same css file for every html page so that we can promote code reusability. Hence this approach is highly recommended to use.

### Basic Structure of CSS File:

```
1) tagname{
2)  property:value;
3) }
```

#### Eg:

```
1) h1{
2)  color:red;
3) }
```

Once we defined css file, we can link it to html by using <link> tag inside <head> part of html.

```
1) <head>
2)  <link rel="stylesheet" href="form.css">
3) </head>
```

### How to define Comments in CSS File:

```
/*
   This is CSS comment
*/
```



## Demo Application-1:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <meta charset="utf-8">
5) <link rel="stylesheet" href="style1.css">
6) <title>CSS Demo</title>
7) </head>
8) <body>
9) <h1>This is First CSS Example</h1>
10) <p>Here we are going to discuss just basics of css like how to define and how to link etc</p>
11) <h2>The List of Topics are:</h2>
12) <ul>
13) <li>Topic-1</li>
14) <li>Topic-2</li>
15) <li>Topic-3</li>
16) <li>Topic-4</li>
17) <li>Topic-5</li>
18) </ul>
19) <h3>Soon we will discuss remaining things</h3>
20) </body>
21) </html>
```

### style1.css:

```
1) h1{
2) color: red;
3) }
4) h2{
5) color: blue;
6) }
7) h3{
8) color:green;
9) }
10) p{
11) color:blue;
12) }
13) ul{
14) color:cyan;
15) }
```



## Various Possible Ways to Specify Color:

- 1) `color:red;`
- 2) `color:rgb(244,66,220);`  
we have to collect values from google color picker  
The allowed values are: 0 to 255  
(0,0,0) → black  
(255,255,255) → white
- 3) `color:#f44e42`  
This 6-digit number represents hexa code of color
- 4) `color:rgba(244,66,220,0.9)`  
a means alpha  
The allowed values for a attribute are: 0.0 to 1.0  
1.0 means full dark and 0.0 means full light (transparent)  
  
<http://www.december.com/html/spec/colorrrgbadec.html>

**Note:** The most commonly used approach is: hexa code

## Setting Background and Borders:

In CSS, we can set Background as follows:

```
1) body{  
2)  background-color:red;  
3) }
```

We can set Border as follows:

```
1) div{  
2)  border-color:blue;  
3)  border-width:thick;  
4)  border-style:double;  
5) }
```

The allowed values for border-width are:

medium,thin,thick.

We can also set our own width with pixels.

**Eg:** `border-width:10px;`

The allowed values for border-style are: dashed,dotted,groove,double etc.



```
1) span{
2)  color:yellow;
3)  background: green;
4) }
```

## Color vs Background:

- color attribute meant for text color.
- background attribute meant for background color.

```
1) h1{
2)  color:white;
3)  background:blue;
4) }
```

## How to Set Background Image:

```
1) body {
2)  background:url(https://image.freepik.com/free-psd/abstract-background-
   design_1297-73.jpg);
3) }
```

## Various properties while setting image:

```
1) body{
2)  color:white;
3)  background:url(https://images.pexels.com/photos/257360/pexels-photo-
   257360.jpeg?auto=compress&cs=tinysrgb&h=350);
4)  background-repeat: no-repeat;
5)  background-size: cover;
6) }
```

By default background image will be repeated. If we don't want to repeat then we should specify: no-repeat

## How to Set Border:

### Normal way:

```
1) h1{
2)  border-color: orange;
3)  border-width: 5px;
4)  border-style: solid;
```



5) }

## short-cut way:

```
1) h1{
2)  border: solid red 5px;   order is not important
3) }
```

## To set border for the image:

```
1) img{
2)  border: groove 10px blue;
3) }
```

## Basic CSS Selectors:

### 1) Element Selectors:

Select all instances of given element. i.e style is applicable for every tag of the specified type.

```
1) h1{
2)  color:red;
3) }
```

This style applicable for every h1 tag of the html page.

### 2) ID Selectors:

Selects an element with the given Id. But with in the HTML Page ID is always unique.

#### html:

<h1 id="specialh1">Hello This is First Line</h1>

#### CSS:

```
1) #specialh1{
2)  color:white;
3)  background: blue;
4) }
```



### 3) Class Selector:

Select all elements with the given class.

#### html:

```
1) <body>
2) <h1 class="specialh1">Hello This is First Line</h1>
3) <h1>Hello This is Second Line</h1>
4) <h1 class="specialh1">Hello This is Third Line</h1>
5) </body>
```

#### CSS:

```
1) specialh1{
2)   color:white;
3)   background: blue;
4) }
```

**Note:** element, id and class selectors are considered as basic css selectors.

### Advanced CSS Selectors:

The following are main important advanced selectors

- 1) \* selector
- 2) Descendant Selector
- 3) Adjacent Selector
- 4) Attribute Selector
- 5) nth of type selector

### Demo HTML Page for CSS Selectors:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4)   <meta charset="utf-8">
5)   <title>CSS Demo</title>
6)   <link rel="stylesheet" href="style1.css">
7) </head>
8) <body>
9)   <h1>Advanced Selectors Demo</h1>
10)  <a href="http://google.com">Click Here to go to Google</a>
11)  <ul>
12)    <li>CAT</li>
```



```
13) <li>RAT</li>
14) <li>DOG</li>
15) </ul>
16) <h4>List of Top Movies</h4>
17) <ul>
18) <li>Bahubali</li>
19) <li>MAHANATI</li>
20) <li>RANGASTALAM</li>
21) </ul>
22) <h4>List of Top Websites</h4>
23) <ul>
24) <li><a href="http://amazon.com">AMAZON</a> </li>
25) <li><a href="http://flipkart.com">FlipKart</a> </li>
26) <li><a href="http://paytm.com">PAYTM</a> </li>
27) </ul>
28) </body>
29) </html>
```

## 1) \* selector:

\* means everything. This style is applicable for every thing in the web page.

```
1) *{
2)   color:blue;
3) }
```

## 2) Descendant Selector:

```
1) li a{
2)   color:white;
3)   background: blue;
4) }
```

li is considered as parent tag and a is considered as child tag.  
For every anchor tag present in li tag this style is applicable

## 3) Adjacent Selector:

```
1) a+ul{
2)   color: red;
3) }
```



For every ul tag which is adjacent to a tag, this style is applicable.

```
1) div+p{  
2)  color:blue;  
3)  }
```

For every paragraph tag, which is adjacent to div tag this style is applicable.

## 4) Attribute Selector:

We can define style based on attributes.

### Eg 1:

```
1) a[href]{  
2)  color:red;  
3)  background: yellow;  
4) }
```

For all href attributes of anchor tag this style is applicable.

### Eg 2:

```
1) a[href="http://amazon.com"]{  
2)  color:red;  
3)  background: yellow;  
4) }
```

If the value of href attribute is http://amazon.com then only this style is applicable.

```
1) input[type="password"]  
2) {  
3)  background:red;  
4) }
```

This style is applicable for all password fields of input tag.

## 5) nth of type selectors:

```
1) li:nth-of-type(2){  
2)  color:red;  
3) }
```

For every 2nd li tag this style is applicable.





```
1) ul:nth-of-type(2){  
2)  color:red;  
3)  background: yellow;  
4) }
```

For every 2nd ul tag this style is applicable.

```
1) li:nth-of-type(even){  
2)  color:red;  
3) }
```

For every even numbered li this style is applicable.

## CSS Inheritance:

- All properties of the parent are by default available to the child and we are not required to redefine. This property is called inheritance.
- Inheritance concept applicable for css styles also. i.e what every styles are defined for the parent automatically available to the child tags also.

```
1) body{  
2)  color:red;  
3) }
```

This style is applicable for all elements present in side body tag.

```
1) ul{  
2)  color:red;  
3) }
```

This style is also applicable for all <li> tags inside <ul> tag.

## CSS Specificity:

If multiple styles are available for element then most specific style will be considered. This property is called Specificity of CSS.

```
1) body{  
2)  color:red;  
3) }  
4) ul{  
5)  color:blue;  
6) }  
7) li{  
8)  color:green;
```



9) }

For <li> tag 3 styles are available but css will consider most specific style from <li> tag which is nothing but green color.

## CSS Styles Assignment-1:



### assignment.html:

- 1) <!DOCTYPE html>
- 2) <html lang="en" dir="ltr">
- 3) <head>
- 4) <meta charset="utf-8">
- 5) <title>CSS Assignment</title>
- 6) <link rel="stylesheet" href="assignment.css">
- 7) </head>
- 8) <body>
- 9) <h1>This is h1 Data</h1>
- 10) <h2>This is h2 tag</h2>
- 11) <h2>This is h2 tag</h2>
- 12) <h2>This is h2 tag</h2>
- 13) <h3>The important Brands are:</h3>
- 14) <ul>
- 15) <li>King Fisher</li>
- 16) <li>Royal Challenge</li>



```
17) <li>Foster</li>
18) </ul>
19) <p class="hello">Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
    eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim ve
    niam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo cons
    equat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore e
    u fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culp
    a qui officia deserunt mollit anim id est laborum.</p>
20)
21) <p id="special">Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do e
    iusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veni
    am, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo conseq
    uat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu f
    ugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa q
    ui officia deserunt mollit anim id est laborum.</p>
22) <label for="name">Name</label>
23) <input type="text" id="name" name="username" value=""><br><br>
24) <label for="pwd">Password</label>
25) <input id="pwd" type="password" name="password" value=""><br><br>
26)
27) <a href="http://www.gmail.com">Gmail</a>
28) <a href="http://www.youtube.com">Youtube</a>
29) <a href="http://www.facebook.com">Facebook</a>
30)
31) <h3>Select Your Favourite Heroine:</h3>
32) <ul>
33) <li>Sunny: <input type="checkbox" name="" value="" checked> </li>
34) <li>Mallika: <input type="checkbox" name="" value="" checked> </li>
35) <li>Veena Malik: <input type="checkbox" name="" value="" > </li>
36) </ul>
37) </body>
38) </html>
```

## assignment.css:

```
1) body{
2) background:#bdc646;
3) }
4) h1{
5) color:#9b79b8;
6) }
7) h2{
8) color:yellow;
9) }
10) li{
```



```
11) color:#0518c4;
12) }
13) p{
14) background:white;
15) }
16) input{
17) border:5px red solid;
18) }
19) .hello{
20) background:blue;
21) color:white;
22) }
23)
24) #special{
25) border: 10px orange solid;
26) }
27) input[type="text"]{
28) background:green;
29) }
30) input:checked{
31) margin-left:50px;
32) }
33)
34) label{
35) text-transform: uppercase;
36) }
37) #special:first-letter{
38) color:green;
39) font-size:100px;
40) }
41) h1:hover{
42) color:white;
43) border:10px red solid;
44) }
45) a:visited{
46) color:yellow;
47) }
```

## Notes:

- 1) Set body tag background color with #bdc646

```
1) body{
2) background:#bdc646;
3) }
```



2) Set h1 text color with #9b79b8

```
1) h1{  
2) color:#9b79b8;  
3) }
```

3) Make all h2 tags with yellow color

```
1) h2{  
2) color:yellow;  
3) }
```

4) Make all <li> elements with blue color, but consider hexa code

```
1) li{  
2) color:#0518c4;  
3) }
```

5) Change background for every paragraph with white color

```
1) p{  
2) background:white;  
3) }
```

6) Make all inputs have a 5px border with red color

```
1) input{  
2) border:5px red solid;  
3) }
```

7) Set blue background with class attribute hello and text color as white

```
1) .hello{  
2) background:blue;  
3) color:white;  
4) }
```

8) Give 10px orange solid border for id attribute "special"

```
1) #special{  
2) border: 10px orange solid;  
3) }
```



9) Make only inputs with type 'text' have a green background

```
1) input[type="text"]{  
2) background:green;  
3) }
```

10) Make all "checked" check boxes have a left margin of 50px

```
1) input:checked{  
2) margin-left:100px;  
3) }
```

11) Make the <label> elements all UPPERCASE without changing in html

```
1) label{  
2) text-transform:uppercase;  
3) }
```

12) Make the first letter of the element with id "special" with green color and 100 px font-size

```
1) #special:first-letter{  
2) color:green;  
3) font-size:100px;  
4) }
```

13) Make <h1> element color change to white when hovered and give 10px solid red border

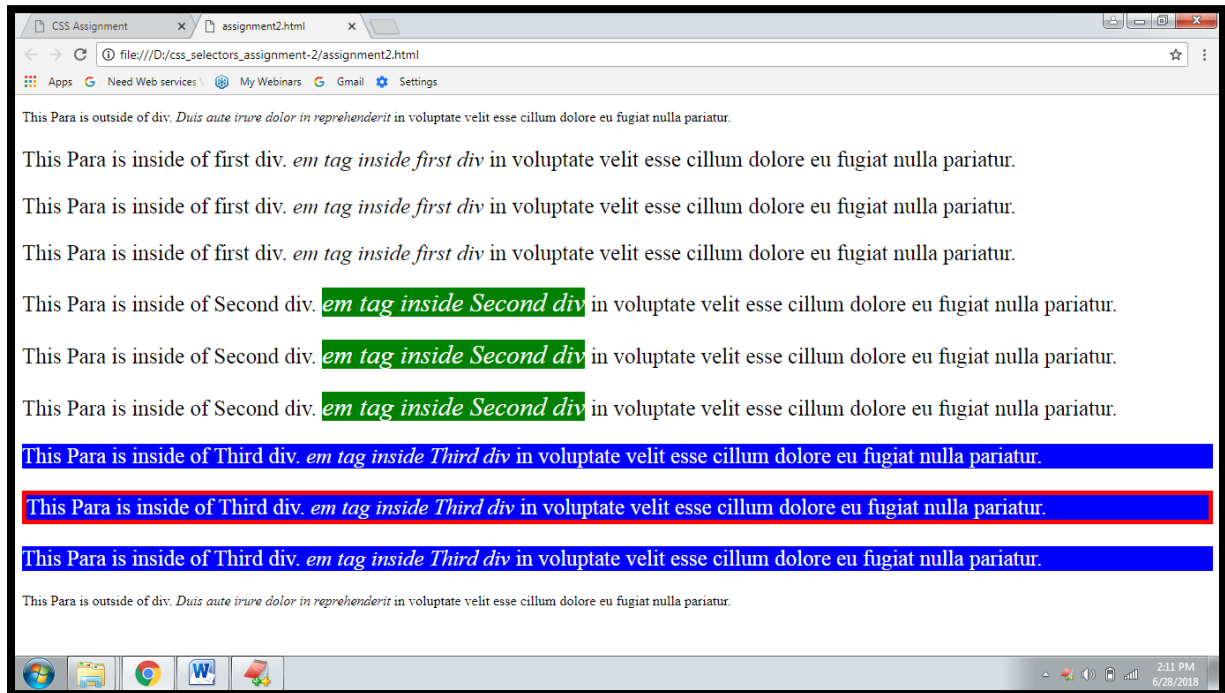
```
1) h1:hover{  
2) color:white;  
3) border:10px red solid;  
4) }
```

14) Make all <a> elements that have been visited with yellow color

```
1) a:visited{  
2) color:yellow;  
3) }
```



## CSS Styles Assignment-2:



### assignment2.html:

- 1) `<!DOCTYPE html>`
- 2) `<html lang="en" dir="ltr">`
- 3) `<head>`
- 4) `<meta charset="utf-8">`
- 5) `<link rel="stylesheet" href="assignment2.css">`
- 6) `<title></title>`
- 7) `</head>`
- 8) `<body>`
- 9) `<p>This Para is outside of div. <em> Duis aute irure dolor in reprehenderit</em> in voluptate velit esse cillum dolore eu fugiat nulla pariatur. </p>`
- 10)
- 11) `<div>`
- 12) `<p>This Para is inside of first div. <em> em tag inside first div</em> in voluptat e velit esse cillum dolore eu fugiat nulla pariatur. </p>`
- 13)
- 14) `<p>This Para is inside of first div. <em> em tag inside first div</em> in voluptat e velit esse cillum dolore eu fugiat nulla pariatur. </p>`
- 15)
- 16) `<p>This Para is inside of first div. <em> em tag inside first div</em> in voluptat e velit esse cillum dolore eu fugiat nulla pariatur. </p>`
- 17) `</div>`
- 18)



```
19) <div>
20) <p>This Para is inside of Second div. <em> em tag inside Second div</em> in vol
    uptate velit esse cillum dolore eu fugiat nulla pariatur. </p>
21)
22) <p>This Para is inside of Second div. <em> em tag inside Second div</em> in vol
    uptate velit esse cillum dolore eu fugiat nulla pariatur. </p>
23)
24) <p>This Para is inside of Second div. <em> em tag inside Second div</em> in vol
    uptate velit esse cillum dolore eu fugiat nulla pariatur. </p>
25) </div>
26)
27) <div>
28) <p>This Para is inside of Third div. <em> em tag inside Third div</em> in volupt
    ate velit esse cillum dolore eu fugiat nulla pariatur. </p>
29)
30) <p>This Para is inside of Third div. <em> em tag inside Third div</em> in volupt
    ate velit esse cillum dolore eu fugiat nulla pariatur. </p>
31)
32) <p>This Para is inside of Third div. <em> em tag inside Third div</em> in volupt
    ate velit esse cillum dolore eu fugiat nulla pariatur. </p>
33) </div>
34) <p>This Para is outside of div. <em> Duis aute irure dolor in reprehenderit</em>
    in voluptate velit esse cillum dolore eu fugiat nulla pariatur. </p>
35) </body>
36) </html>
```

## assignment2.css:

```
1) div p{
2)  font-size: 25px;
3) }
4)
5) div:nth-of-type(3) p{
6)  background:blue;
7)  color:white;
8) }
9)
10) div:nth-of-type(3) p:nth-of-type(2){
11)  border: 5px red solid;
12) }
13)
14) div:nth-of-type(2) em{
15)  color: white;
16)  font-size:30px;
17)  background:green;
```





```
| 18) }
```

## Notes:

**Case-1:** Make all <p>s That are nested inside div with 25px font-size

```
| 1) div p{  
| 2)  font-size: 25px;  
| 3) }
```

**Case-2:** Give all <p>s in side 3rd div blue background and text color is white

```
| 1) div:nth-of-type(3) p{  
| 2)  background:blue;  
| 3)  color:white;  
| 4) }
```

**Case-3:** Give 2nd <p> inside 3rd div, 5px red solid border

```
| 1) div:nth-of-type(3) p:nth-of-type(2){  
| 2)  border: 5px red solid;  
| 3) }
```

**Case-4:** Make <em> tag in the 2nd <div> with white color 30 px font-size and green background

```
| 1) div:nth-of-type(2) em{  
| 2)  color: white;  
| 3)  font-size:30px;  
| 4)  background:green;  
| 5) }
```

## Fonts and Text in CSS:

The following are very important properties related to fonts and text in css

- 1) font-family
- 2) font-size
- 3) font-weight
- 4) line-height
- 5) text-align
- 6) text-decoration



## 1) font-family:

We can select desired font from default css system fonts in the following link  
<https://www.cssfontstack.com/>

```
1) h1{  
2)  font-family: Arial Black;  
3) }
```

**Note:** If we are not satisfied with default css system fonts, then we can use external fonts also.

## 2) font-size:

```
1) p{  
2)  font-size: 20px;  
3) }
```

We can also specify font-size in em units, which is also known as dynamic font-size (relative font-size)

```
1) span{  
2)  font-size: 2.0em;  
3) }
```

2.0em means double of parent tag font-size

## 3) font-weight:

```
1) p{  
2)  font-weight: 600;  
3) }
```

something like bold font, light font etc

The different allowed values are:

bold, bolder, lighter, normal

100 to 900 where 100 means light and 900 means too much bold.

## 4) line-height:

The space between 2 lines is called line height.

```
1) p{  
2)  line-height: 1.5;  
3) }
```



## 5) text-align:

- 1) p{
- 2) text-align:center;
- 3) }

The allowed values are: left, right, center, justify

## 6) text-decoration:

Like underlined, strike through

- 1) p{
- 2) text-decoration: line-through;
- 3) }

The allowed values are: underline, overline, line-through

## How to use Custom Fonts in CSS:

Most of the times we can select custom fonts from the google <https://fonts.google.com/>

select required fonts and add that link in html header part.  
Inside css file specify that font with font-family attribute.

### demo.html:

- 1) <!DOCTYPE html>
- 2) <html lang="en" dir="ltr">
- 3) <head>
- 4) <meta charset="utf-8">
- 5) <title>Demo for CSS</title>
- 6) <link rel="stylesheet" href="demo.css">
- 7) <link href="https://fonts.googleapis.com/css?family=Eater|Great+Vibes|Indie+Flower" rel="stylesheet">
- 8) </head>
- 9) <body>
- 10) <h1>This is Heading</h1>
- 11) <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. <span>Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat</span>. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
- 12) </body>
- 13) </html>



## demo.css:

```
1) body{  
2)  font-family: 'Great Vibes', cursive;  
3)  font-size:30px;  
4) }
```

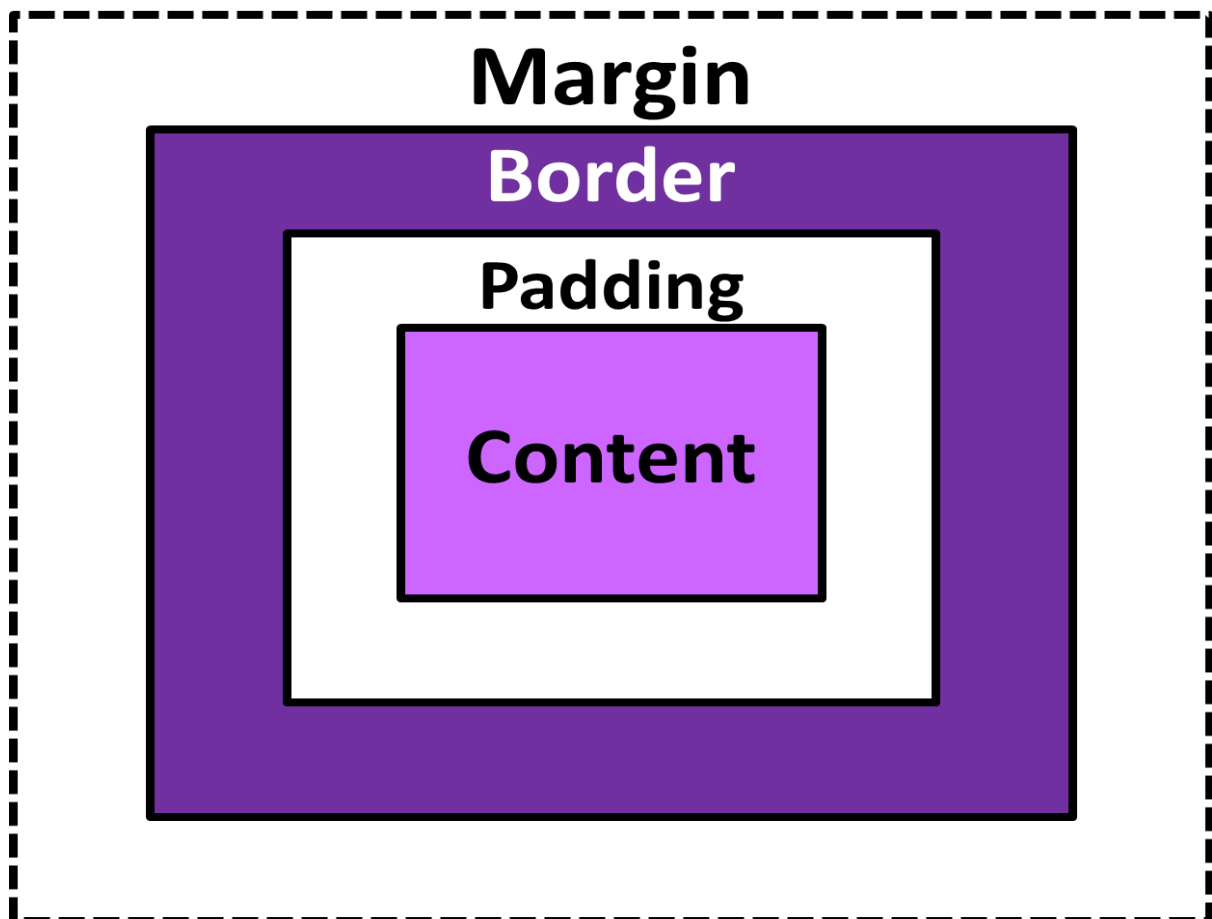
## The Box Model:

In a document, each element is represented as a rectangular box. In CSS, each of these rectangular boxes is described by using the standard box model.

Each box has 4 edges:

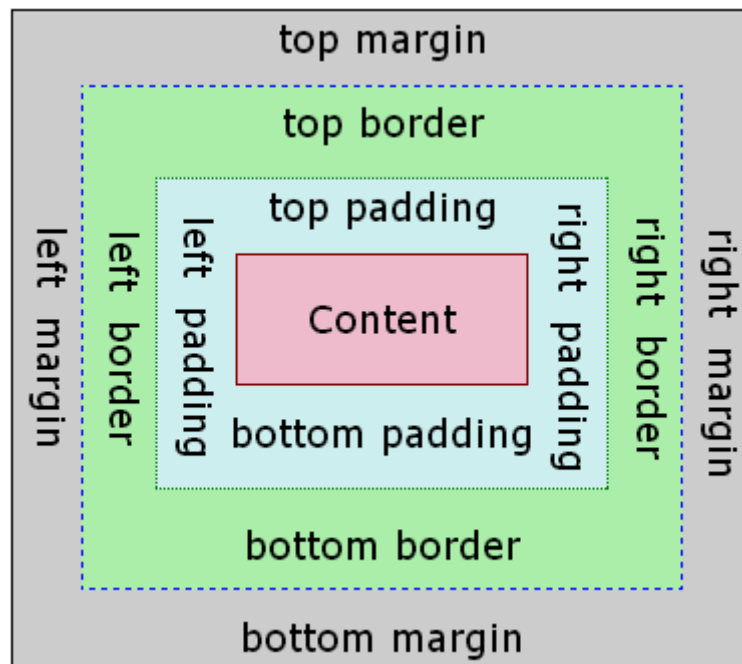
- 1) Content Edge
- 2) Border Edge
- 3) Padding Edge
- 4) Margin Edge

CSS Box Model Diagram -1





## CSS Box Model Diagram -2



### Demo Program:

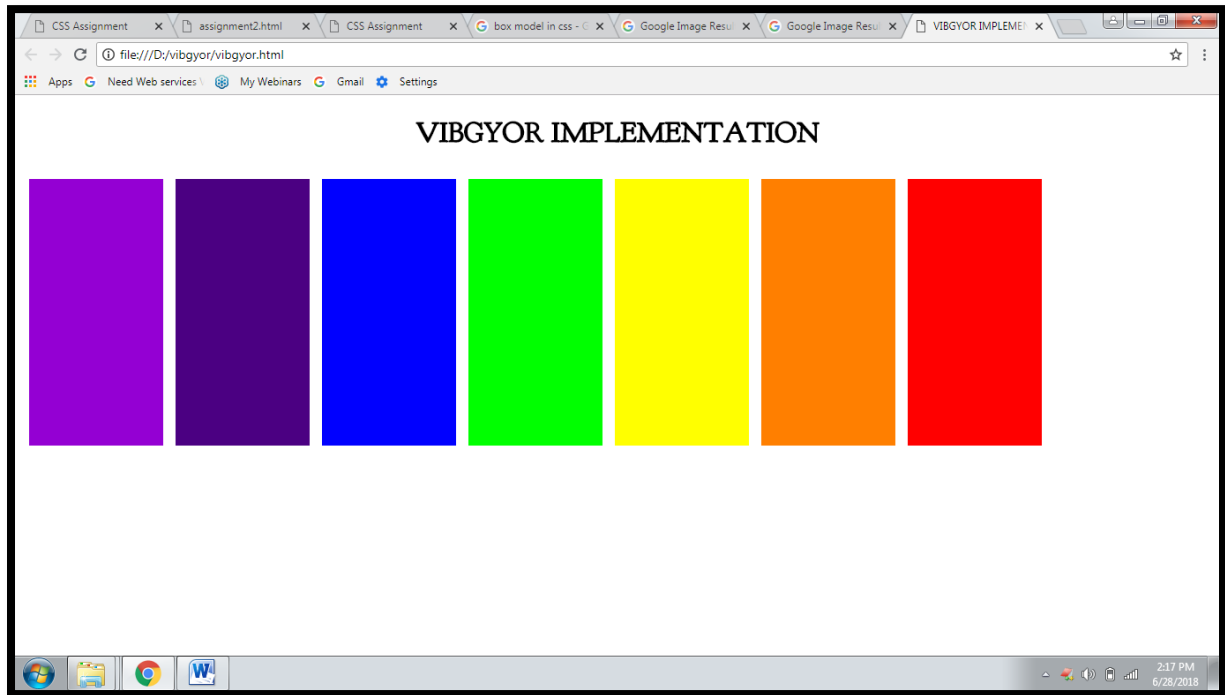
- 1) `<p>This is Paragraph</p>`
- 2) `<p>This is Paragraph</p>`

### CSS:

- 1) `p{`
- 2) `width: 200px;`
- 3) `border: 3px solid blue;`
- 4) `padding: 40px 50px 60px 70px;`
- 5) `margin: 100px 5px 300px 400px;`
- 6) `}`



## Vibgyor Implementation



### vibgyor.html:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <meta charset="utf-8">
5) <title>VIBGYOR IMPLEMENTATION</title>
6) <link rel="stylesheet" href="vibgyor.css">
7) <link href="https://fonts.googleapis.com/css?family=Goudy+Bookletter+1911" rel="stylesheet">
8) </head>
9) <body>
10) <h1>VIBGYOR IMPLEMENTATION</h1>
11) <table class='tab'>
12) <tr>
13) <td id="one"></td>
14) <td id="two"></td>
15) <td id="three"></td>
16) <td id="four"></td>
17) <td id="five"></td>
18) <td id="six"></td>
19) <td id="seven"></td>
20) </tr>
21) </table>
```



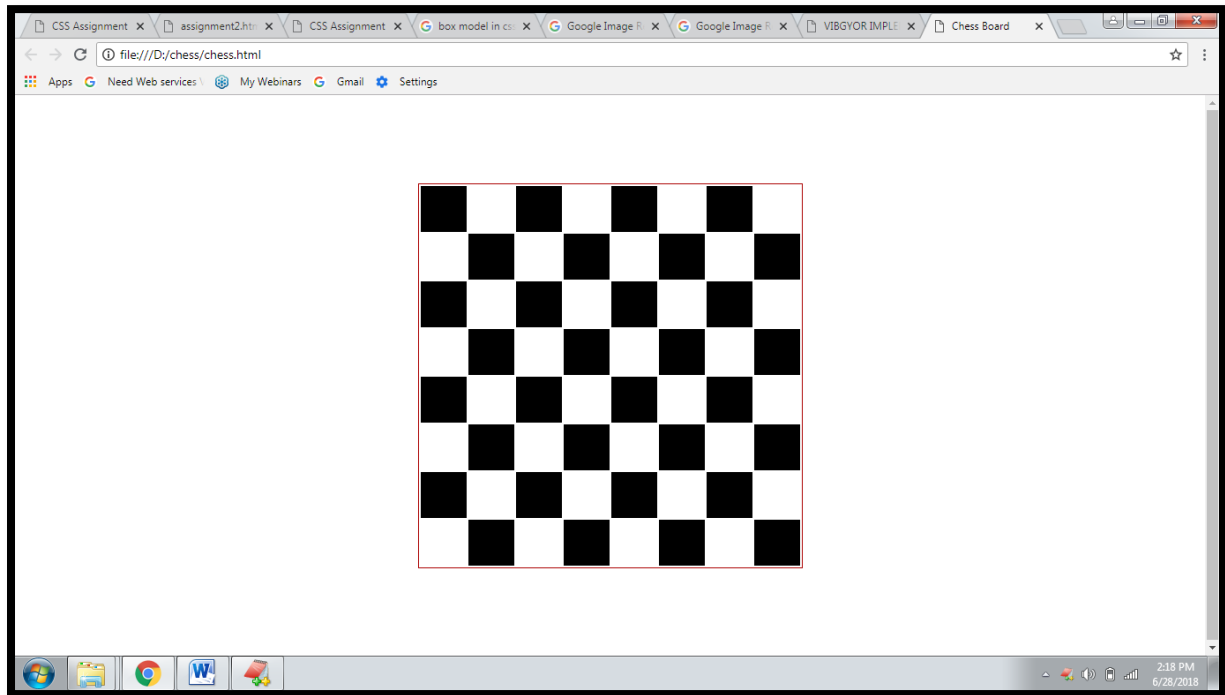
```
22) </body>
23) </html>
```

## vibgyor.css:

```
1) h1{
2)  font-family: 'Goudy Bookletter 1911', serif;
3)  text-align: center;
4) }
5)
6) td {
7)  height:300px;
8)  width:150px;
9)  border: 6px solid white;
10) }
11) #one{
12)  background: #9400D3;
13) }
14) #two{
15)  background: #4B0082;
16) }
17) #three{
18)  background: #0000FF ;
19) }
20) #four{
21)  background: #00FF00 ;
22) }
23) #five{
24)  background: #FFFF00;
25) }
26) #six{
27)  background: #FF7F00;
28) }
29) #seven{
30)  background: #FF0000;
31) }
```



## Chess Board Implementation



### chess.html:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <meta charset="utf-8">
5) <title>Chess Board</title>
6) <link rel="stylesheet" href="chess.css">
7) </head>
8) <body>
9) <table>
10) <tr>
11) <td id="black"></td>
12) <td id="white"></td>
13) <td id="black"></td>
14) <td id="white"></td>
15) <td id="black"></td>
16) <td id="white"></td>
17) <td id="black"></td>
18) <td id="white"></td>
19) </tr>
20) <tr>
21)
22) <td id="white"></td>
```





```
23) <td id="black"></td>
24) <td id="white"></td>
25) <td id="black"></td>
26) <td id="white"></td>
27) <td id="black"></td>
28) <td id="white"></td>
29) <td id="black"></td>
30) </tr>
31) <tr>
32) <td id="black"></td>
33) <td id="white"></td>
34) <td id="black"></td>
35) <td id="white"></td>
36) <td id="black"></td>
37) <td id="white"></td>
38) <td id="black"></td>
39) <td id="white"></td>
40) </tr>
41) <tr>
42)
43) <td id="white"></td>
44) <td id="black"></td>
45) <td id="white"></td>
46) <td id="black"></td>
47) <td id="white"></td>
48) <td id="black"></td>
49) <td id="white"></td>
50) <td id="black"></td>
51) </tr>
52) <tr>
53) <td id="black"></td>
54) <td id="white"></td>
55) <td id="black"></td>
56) <td id="white"></td>
57) <td id="black"></td>
58) <td id="white"></td>
59) <td id="black"></td>
60) <td id="white"></td>
61) </tr>
62) <tr>
63)
64) <td id="white"></td>
65) <td id="black"></td>
66) <td id="white"></td>
67) <td id="black"></td>
```



```
68) <td id="white"></td>
69) <td id="black"></td>
70) <td id="white"></td>
71) <td id="black"></td>
72) </tr>
73) <tr>
74) <td id="black"></td>
75) <td id="white"></td>
76) <td id="black"></td>
77) <td id="white"></td>
78) <td id="black"></td>
79) <td id="white"></td>
80) <td id="black"></td>
81) <td id="white"></td>
82) </tr>
83) <tr>
84)
85) <td id="white"></td>
86) <td id="black"></td>
87) <td id="white"></td>
88) <td id="black"></td>
89) <td id="white"></td>
90) <td id="black"></td>
91) <td id="white"></td>
92) <td id="black"></td>
93) </tr>
94) </table>
95)
96) </body>
97) </html>
```

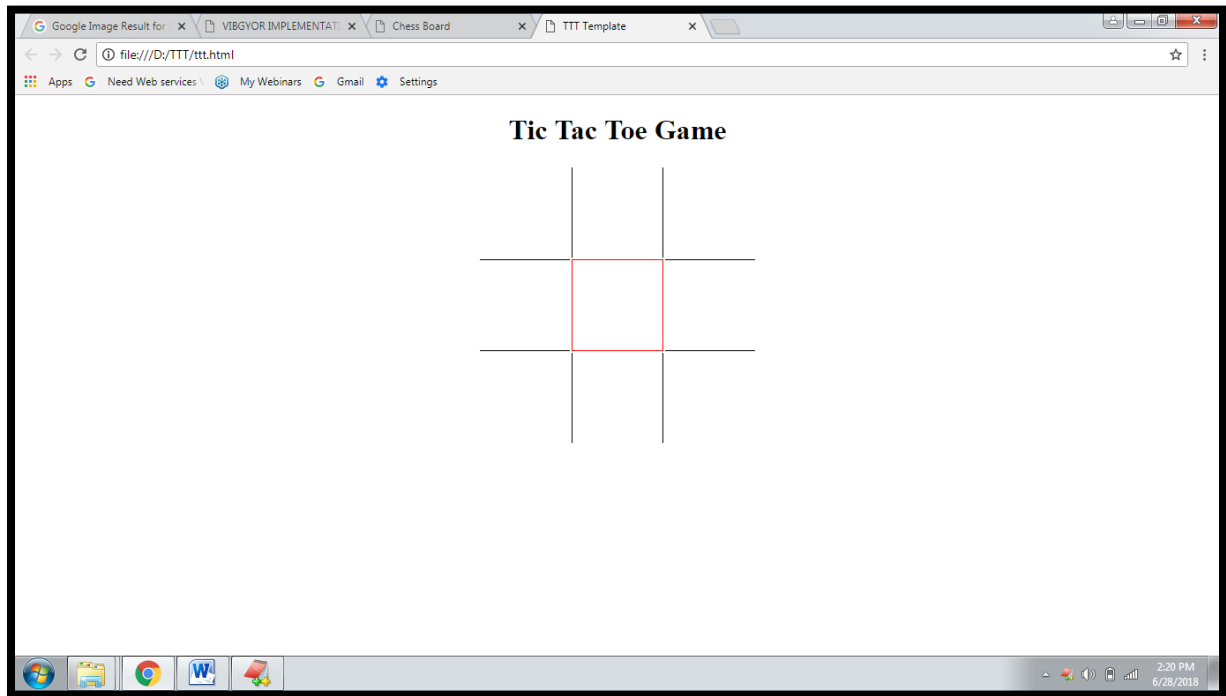
## chess.css:

```
1) table{
2)   border:1px groove red;
3)   margin: 100px auto;
4) }
5) td{
6)   height:50px;
7)   width:50px;
8) }
9) #white{
10)  background: white;
11) }
12) #black{
```



- 13) background: black;
- 14) }

## Tic-Tac-Toe Implementation



### ttt.html:

- 1) `<!DOCTYPE html>`
- 2) `<html lang="en" dir="ltr">`
- 3) `<head>`
- 4) `<link rel="stylesheet" href="ttt.css">`
- 5) `<meta charset="utf-8">`
- 6) `<title>TTT Template</title>`
- 7) `</head>`
- 8) `<body>`
- 9) `<h1>Tic Tac Toe Game</h1>`
- 10) `<table>`
- 11) `<tr>`
- 12) `<td></td>`
- 13) `<td class="vertical"></td>`
- 14) `<td></td>`
- 15) `</tr>`
- 16) `<tr>`
- 17) `<td class="horizontal"></td>`
- 18) `<td class="vertical horizontal"></td>`
- 19) `<td class="horizontal"></td>`



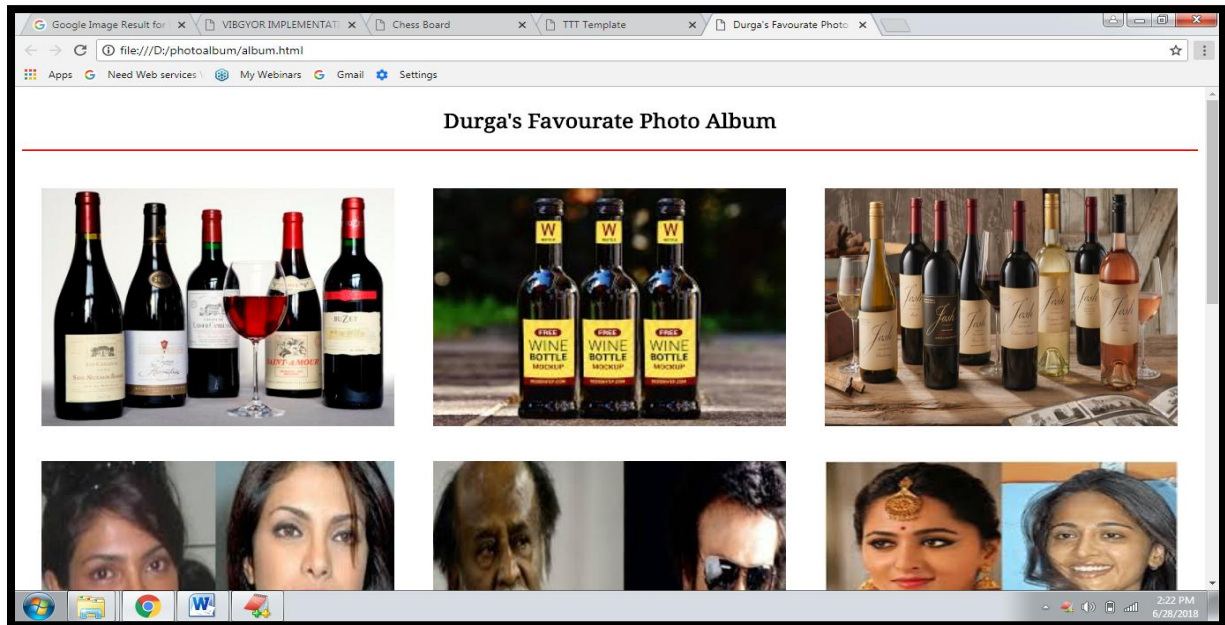
```
20) </tr>
21) <tr>
22) <td></td>
23) <td class="vertical"></td>
24) <td></td>
25) </tr>
26) </table>
27) </body>
28) </html>
```

## ttt.css:

```
1) td{
2) width: 100px;
3) height: 100px;
4) }
5) .vertical{
6) border-left: 1px solid black;
7) border-right: 1px solid black;
8) }
9) .horizontal{
10) border-top: 1px solid black;
11) border-bottom: 1px solid black;
12) }
13) td:hover{
14) border: 1px solid red;
15) }
16) table{
17) margin:auto;
18) }
19) h1{
20) text-align: center;
21) }
```



## Durga's Favourite Photo Album



### album.html:

- 1) `<!DOCTYPE html>`
- 2) `<html lang="en" dir="ltr">`
- 3) `<head>`
- 4) `<meta charset="utf-8">`
- 5) `<title>Durga's Favourite Photo Album</title>`
- 6) `<link rel="stylesheet" href="album.css">`
- 7)
- 8) `<link href="https://fonts.googleapis.com/css?family=Goudy+Bookletter+1911|Not+Serif" rel="stylesheet">`
- 9)
- 10) `</head>`
- 11) `<body>`
- 12) `<p>Durga's Favourite Photo Album</p>`
- 13) ``
- 14) ``
- 15) ``
- 16) ``
- 17) ``
- 18) ``
- 19) ``
- 20) ``
- 21) ``
- 22) ``
- 23) ``
- 24) ``

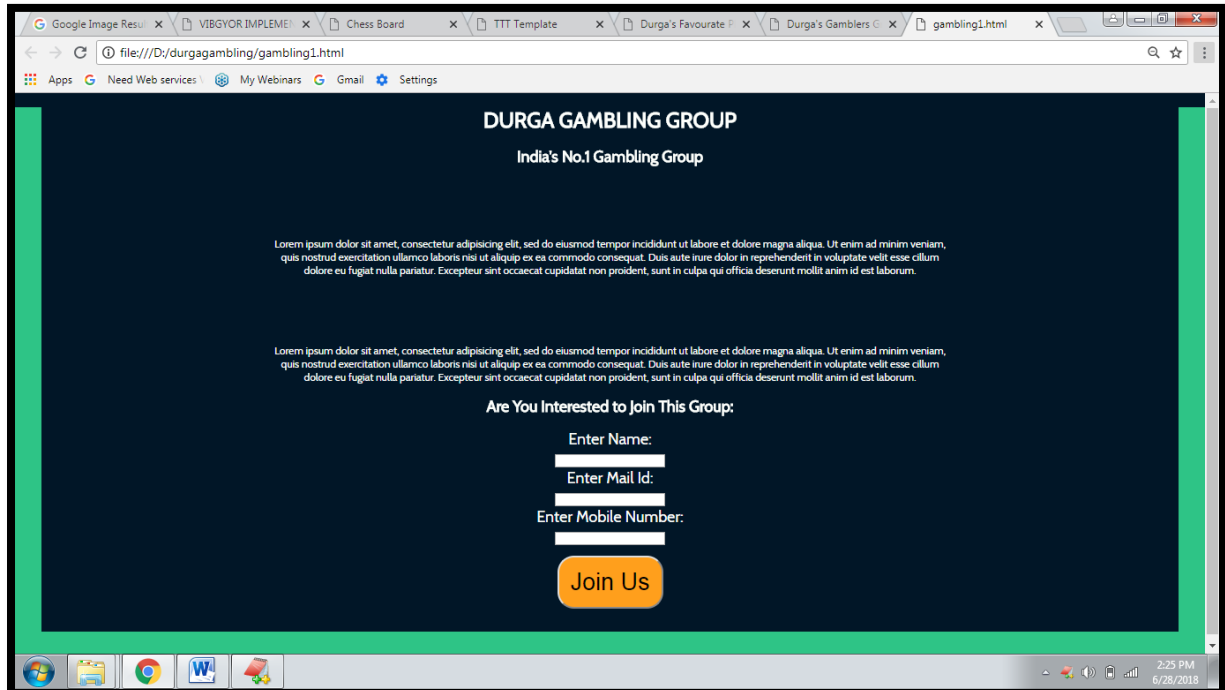


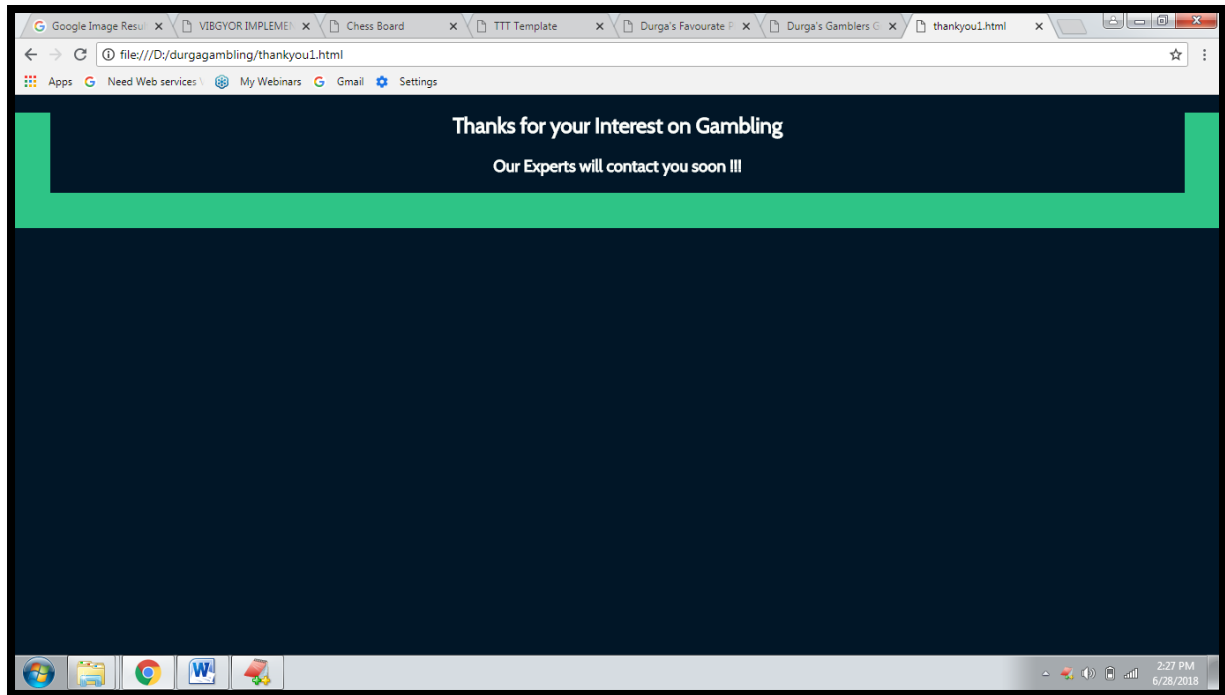
```
25) </body>
26) </html>
```

## album.css:

```
1) img{
2)   width: 30%;
3)   height:300px;
4)   float: left;
5)   margin: 1.66%;
6) }
7) p{
8)   font-family: 'Noto Serif', serif;
9)   text-align: center;
10)  font-size: 25px;
11)  font-weight: 800;
12)  border-bottom: 2px solid red;
13)  padding-bottom: 20px;
14) }
```

## Durga Gambling Website





## gambling.html:

- 1) `<!DOCTYPE html>`
- 2) `<html lang="en" dir="ltr">`
- 3) `<head>`
- 4) `<meta charset="utf-8">`
- 5) `<title>Durga's Gamblers Group</title>`
- 6) `<link rel="stylesheet" href="gambling.css">`
- 7) `<link href="https://fonts.googleapis.com/css?family=Josefin+Sans" rel="stylesheet">`
- 8) `</head>`
- 9) `<body>`
- 10) `<h1>DURGA GAMBLING GROUP</h1>`
- 11) `<h2>India's No.1 Gambling Group</h2>`
- 12) `<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>`
- 13) `<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>`
- 14) `<h2>Are You Interested to Join This Group:</h2>`



```
15) <form class="" action="thanks.html" method="post">
16) <label for="name">Enter Name:</label><br>
17) <input id="name" type="text" name="name" value=""><br>
18) <label for="mail">Enter Mail Id:</label><br>
19) <input id="mail" type="email" name="" value=""><br>
20) <label for="mobile">Enter Mobile Number:</label><br>
21) <input id="mobile" type="text" name="" value=""><br>
22) <input id="sub" type="submit" name="" value="Join Us">
23) </form>
24) </body>
25) </html>
```

## thanks.html:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <meta charset="utf-8">
5) <title>Thanking Page</title>
6) <link href="https://fonts.googleapis.com/css?family=Josefin+Sans" rel="stylesheet">
7) <link rel="stylesheet" href="gambling.css">
8) </head>
9) <body>
10) <h2>Thanks for Your Interest on Gambling</h2>
11) <h3>Our Experts will contact You soon !!!!</h3>
12) </body>
13) </html>
```

## gambling.css:

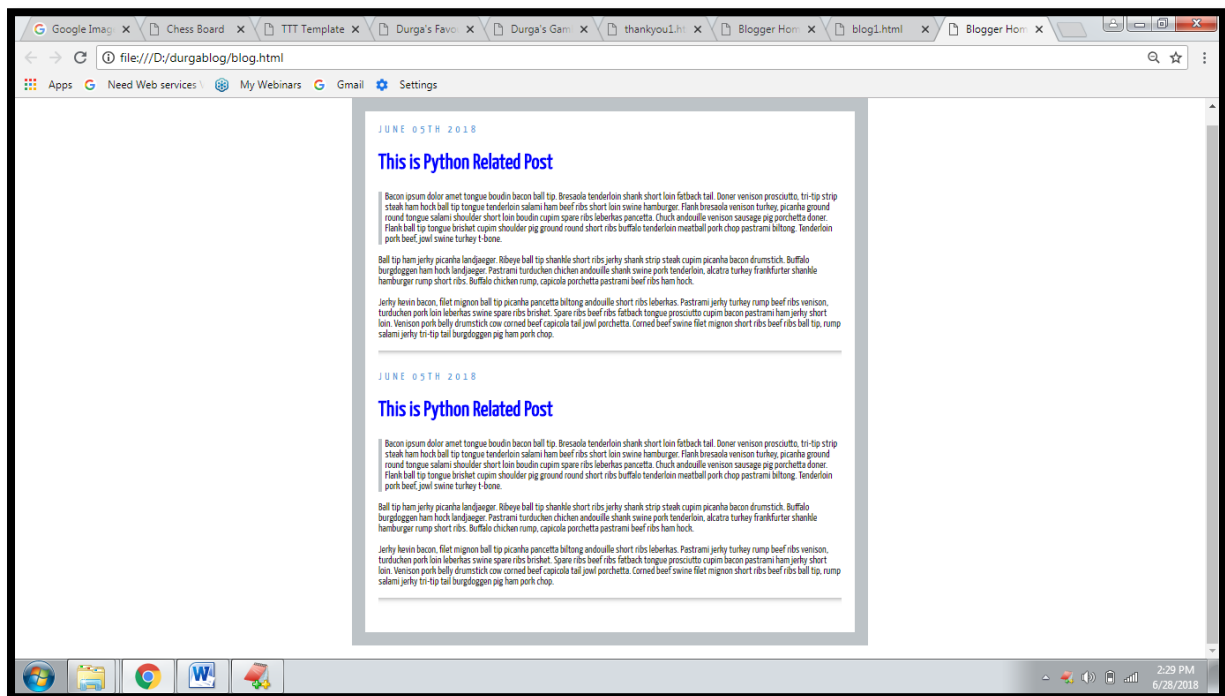
```
1) body{
2) font-family: 'Josefin Sans', sans-serif;
3) background: #011627;
4) color: #fdfffc;
5) text-align: center;
6) border: 40px solid #2EC486;
7) border-top: 0px;
8) margin: 0px;
9) }
10) p{
11) padding-top: 5%;
12) padding-left: 20%;
13) padding-right: 20%;
14) }
15) form{
```





```
16) font-size: 1.5em;
17) margin: 20px;
18) }
19) #sub{
20) background: #FF9F1C;
21) height: 80px;
22) width: 150px;
23) margin: 10px;
24) font-size: 1.5em;
25) border-radius: 25px;
26) }
```

## Blog design by using HTML and CSS



### blog.html:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <meta charset="utf-8">
5) <title>Blogger Home Page</title>
6) <link rel="stylesheet" href="blog.css">
7) <link href="https://fonts.googleapis.com/css?family=Indie+Flower|Lobster|Yan
one+Kaffeesatz" rel="stylesheet">
8) </head>
9) <body>
```



- 10) `<div class="post">`
- 11) `<div class='date'> June 05th 2018</div>`
- 12) `<h2>This is Python Related Post</h2>`
- 13) `<p class="mainp">`Bacon ipsum dolor amet tongue boudin bacon ball tip. Bresaol  
a tenderloin shank short loin fatback tail. Doner venison prosciutto, tri-  
tip strip steak ham hock ball tip tongue tenderloin salami ham beef ribs short loin s  
wine hamburger. Flank bresaola venison turkey, picanha ground round tongue sala  
mi shoulder short loin boudin cupim spare ribs leberkas pancetta. Chuck andouille  
venison sausage pig porchetta doner. Flank ball tip tongue brisket cupim shoulder  
pig ground round short ribs buffalo tenderloin meatball pork chop pastrami biltong  
. Tenderloin pork beef, jowl swine turkey t-bone.
- 14) `</p>`
- 15) `<p>`
- 16) Ball tip ham jerky picanha landjaeger. Ribeye ball tip shankle short ribs jerky sh  
ank strip steak cupim picanha bacon drumstick. Buffalo burgdoggen ham hock land  
jaeger. Pastrami turducken chicken andouille shank swine pork tenderloin, alcatra  
turkey frankfurter shankle hamburger rump short ribs. Buffalo chicken rump, capic  
ola porchetta pastrami beef ribs ham hock.
- 17) `</p>`
- 18) `<p>`
- 19) Jerky kevin bacon, filet mignon ball tip picanha pancetta biltong andouille short  
ribs leberkas. Pastrami jerky turkey rump beef ribs venison, turducken pork loin leb  
erkas swine spare ribs brisket. Spare ribs beef ribs fatback tongue prosciutto cupim  
bacon pastrami ham jerky short loin. Venison pork belly drumstick cow corned bee  
f capicola tail jowl porchetta. Corned beef swine filet mignon short ribs beef ribs ba  
ll tip, rump salami jerky tri-tip tail burgdoggen pig ham pork chop.
- 20) `</p>`
- 21) `<hr>`
- 22) `</div>`
- 23) `<div class="post">`
- 24) `<div class='date'> June 05th 2018</div>`
- 25) `<h2>This is Python Related Post</h2>`
- 26) `<p class="mainp">`Bacon ipsum dolor amet tongue boudin bacon ball tip. Bresaola  
tenderloin shank short loin fatback tail. Doner venison prosciutto, tri-  
tip strip steak ham hock ball tip tongue tenderloin salami ham beef ribs short loin s  
wine hamburger. Flank bresaola venison turkey, picanha ground round tongue sala  
mi shoulder short loin boudin cupim spare ribs leberkas pancetta. Chuck andouille  
venison sausage pig porchetta doner. Flank ball tip tongue brisket cupim shoulder  
pig ground round short ribs buffalo tenderloin meatball pork chop pastrami biltong  
. Tenderloin pork beef, jowl swine turkey t-bone.
- 27) `</p>`
- 28) `<p>`
- 29) Ball tip ham jerky picanha landjaeger. Ribeye ball tip shankle short ribs jerky sha  
nk strip steak cupim picanha bacon drumstick. Buffalo burgdoggen ham hock landj  
aeger. Pastrami turducken chicken andouille shank swine pork tenderloin, alcatra t



urkey frankfurter shankle hamburger rump short ribs. Buffalo chicken rump, capicola porchetta pastrami beef ribs ham hock.

30) `</p>`

31) `<p>`

32) Jerky kevin bacon, filet mignon ball tip picanha pancetta biltong andouille short ribs leberkas. Pastrami jerky turkey rump beef ribs venison, turducken pork loin leberkas swine spare ribs brisket. Spare ribs beef ribs fatback tongue prosciutto cupim bacon pastrami ham jerky short loin. Venison pork belly drumstick cow corned beef capicola tail jowl porchetta. Corned beef swine filet mignon short ribs beef ribs ball tip, rump salami jerky tri-tip tail burgdoggen pig ham pork chop.

33) `</p>`

34) `<hr>`

35) `</div>`

36) `</body>`

37) `</html>`

## blog.css:

```
1) body{
2)   border: 20px solid #bdc3c7;
3)   padding:20px;
4)   width:700px;
5)   margin:20px auto;
6)   font-family: 'Yanone Kaffeesatz', sans-serif;
7) }
8) .mainp{
9)   border-left: 5px solid #bdc3c7;
10)  padding-left: 5px;
11) }
12) .date{
13)  text-transform: uppercase;
14)  color: #3498db;
15)  letter-spacing: 5px;
16) }
17) h2{
18)  color: blue;
19)  font-size: 2.0em;
20) }
21) .post{
22)  margin-bottom: 20px;
23) }
24) hr {
25)  height: 12px; border: 0; box-shadow: inset 0 12px 12px -12px rgba(0, 0, 0, 0.5);
26) }
```



**Bootstrap**

# **Study Material**



# Bootstrap

- Bootstrap is the most commonly used framework for Front-End Development. [Django is the most commonly used web framework for back-end development with Python]
- Bootstrap providing several pre defined libraries for css and java script.
- Current version of Bootstrap is: 4.1.1

## Bootstrap 3 vs Bootstrap 4:

- 1) Panels are replaced with Cards
- 2) Larger default font sizes
- 3) New Grid Tier(XL)
- 4) Use of Flexbox
- 5) Moved from Less to Sass

Website: [getbootstrap.com](https://getbootstrap.com)

## How to connect Bootstrap with HTML:

We can connect Bootstrap with HTML by using the following 2 ways

- 1) By using CDN
- 2) Locally

### 1) By using CDN:

- Content Delivery Network
- just add the following in the <head> part of our html
- `<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css">`

### demo.html

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css">
5) <meta charset="utf-8">
6) <title></title>
```



```
7) </head>
8) <body>
9) <h1>This is First Bootstrap Demo</h1>
10) </body>
11) </html>
```

## 2) Locally:

- Download bootstrap.css file from the link [getbootstrap.com](https://getbootstrap.com) and download button <https://github.com/twbs/bootstrap/releases/download/v4.1.1/bootstrap-4.1.1-dist.zip>
- zip file contains bootstrap.css file.
- copy this file in our application folder and add the following link in html  
`<link rel="stylesheet" href="Bootstrap.css">`

## demo.html:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <link rel="stylesheet" href="Bootstrap.css">
5) <meta charset="utf-8">
6) <title></title>
7) </head>
8) <body>
9) <h1>This is First Bootstrap Demo</h1>
10) </body>
11) </html>
```

## Agenda

- 1) Buttons
- 2) Forms
- 3) Nav
- 4) Grid

## 1) Bootstrap- Buttons:

- Most of the styles in Bootstrap are implemented as class styles.(ie class selectors)
- We can use these classes directly in our web application to improve look and feel.
- The following are commonly used classes
  - 1) container class
  - 2) button related classes
  - 3) jumbotron classes



## 🚫 Container Class:

To center our elements we should use container class.

- 1) `<div class="container">`
- 2) `<h1>This is First Bootstrap Demo</h1>`
- 3) `</div>`

## 🚫 Button Classes:

We can use button related classes for `<a>`, `<button>` and `<input>` tags.

## How to use Bootstrap Success Style for Our Buttons:

We have to use `btn btn-success`

```
<button type="button" name="button" class="btn btn-success ">Login</button>
```

To make our button as large: `btn-lg`

```
<button type="button" name="button" class="btn btn-success btn-lg ">Login</button>
```

## To make Button as Active:

We have to use active class

```
<button type="button" name="button" class="btn btn-success btn-lg active">  
Login</button>
```

## To make Button as disabled:

```
<button type="button" name="button" class="btn btn-success"  
disabled="disabled">Login</button>
```

## Usage of Button related Classes for Anchor Tag:

```
<a href="http://getbootstrap.com" class="btn btn-success btn-lg">Click Here for  
Bootstrap Documentation</a>
```

## Usage of Button related Classes for Input Tag:

We can use button related classes for input tag also, but not recommended to use.

Name: `<input type="text" class="btn btn-success btn-lg" name="" value="">`

## How to Change Default Styles of Bootstrap:

Based on our requirement we can customize/change bootstrap default styles.

- 1) `<html lang="en" dir="ltr">`
- 2) `<head>`
- 3) `<link rel="stylesheet" href="Bootstrap.css">`
- 4) `<style type="text/css">`
- 5) `.btn-success{`



```
6)    background: orange;
7)    color: blue;
8)    }
9)    </style>
10)   </head>
11)   <body>
12)   <div class="container">
13)   <button type="button" name="button" class="btn btn-
    success ">Login</button>
14)   </div>
15)   </body>
16)  </html>
```

## 🚫 Jumbotron Classes:

A lightweight, flexible component that can optionally extend the entire viewport to showcase key content on our site.

```
1) <div class="jumbotron">
2)   <h1>Hello, world!</h1>
3)   <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, dsfsafd.</p>
4)   <p><a class="btn btn-primary btn-lg" href="#" role="button">
    Learn more</a></p>
5) </div>
6) </div>
```

**Note:** Several scenarios with HTML, CSS and JS <https://www.w3schools.com/howto/>

## 2) Bootstrap Forms:

Bootstrap provides several classes for forms. The main important classes are

### 1) form-group:

It is responsible to maintain proper space between elements so that elements will be arranged properly.

### 2) form-control:

This class is responsible to make width as 100% for elements <input>, <textarea>, and <select>. It is also responsible for border styling.

## Basic Example Form:

```
1) <form>
2) <div class="form-group">
3)   <label for="exampleInputEmail1">Email address</label>
```





```
4) <input type="email" class="form-control" id="exampleInputEmail1"
placeholder="Email">
5) </div>
6) <div class="form-group">
7) <label for="exampleInputPassword1">Password</label>
8) <input type="password" class="form-
control" id="exampleInputPassword1" placeholder="Password">
9) </div>
10) <div class="form-group">
11) <label for="exampleInputFile">File input</label>
12) <input type="file" id="exampleInputFile">
13) <p class="help-block">Example block-level help text here.</p>
14) </div>
15) <div class="checkbox">
16) <label>
17) <input type="checkbox"> Check me out
18) </label>
19) </div>
20) <button type="submit" class="btn btn-default">Submit</button>
21) </form>
```

## Inline Form:

```
1) <form class="form-inline">
2) <div class="form-group">
3) <label for="exampleInputName2">Name</label>
4) <input type="text" class="form-
control" id="exampleInputName2" placeholder="Jane Doe">
5) </div>
6) <div class="form-group">
7) <label for="exampleInputEmail2">Email</label>
8) <input type="email" class="form-
control" id="exampleInputEmail2" placeholder="jane.doe@example.com">
9) </div>
10) <button type="submit" class="btn btn-default">Send invitation</button>
11) </form>
```

## Form Development by using Bootstrap Elements:

```
1) <form>
2) <!-- EMAIL SUBMISSION -->
3) <div class="form-group">
4) <label for="exampleInputEmail1">Email address</label>
```



```
5) <input type="email" class="form-control" id="exampleInputEmail1" aria-  
describedby="emailHelp" placeholder="Enter email">  
6) <small id="emailHelp" class="form-text text-  
muted">We'll never share your email with anyone else.</small>  
7) </div>  
8)  
9) <!-- PASSWORD -->  
10) <div class="form-group">  
11) <label for="exampleInputPassword1">Password</label>  
12) <input type="password" class="form-  
control" id="exampleInputPassword1" placeholder="Password">  
13) </div>  
14)  
15) <!-- DROPDOWN SELECT -->  
16) <div class="form-group">  
17) <label for="exampleSelect1">Example select</label>  
18) <select class="form-control" id="exampleSelect1">  
19) <option>1</option>  
20) <option>2</option>  
21) <option>3</option>  
22) <option>4</option>  
23) <option>5</option>  
24) </select>  
25) </div>  
26)  
27) <!-- MULTIPLE SELECT OPTIONS -->  
28) <div class="form-group">  
29) <label for="exampleSelect2">Example multiple select</label>  
30) <select multiple class="form-control" id="exampleSelect2">  
31) <option>1</option>  
32) <option>2</option>  
33) <option>3</option>  
34) <option>4</option>  
35) <option>5</option>  
36) </select>  
37) </div>  
38)  
39) <!-- TEXT AREA -->  
40) <div class="form-group">  
41) <label for="exampleTextarea">Example textarea</label>  
42) <textarea class="form-control" id="exampleTextarea" rows="3"></textarea>  
43) </div>  
44)  
45) <!-- FILE UPLOAD INPUT -->  
46) <div class="form-group">
```



```
47) <label for="exampleInputFile">File input</label>
48) <input type="file" class="form-control-file" id="exampleInputFile" aria-
    describedby="fileHelp">
49) <small id="fileHelp" class="form-text text-
    muted">This is some placeholder block-
    level help text for the above input. It's a bit lighter and easily wraps to a new line.<
    /small>
50) </div>
51)
52) <!-- RADIO BUTTONS -->
53) <fieldset class="form-group">
54) <legend>Radio buttons</legend>
55) <div class="form-check">
56) <label class="form-check-label">
57) <input type="radio" class="form-check-
    input" name="optionsRadios" id="optionsRadios1" value="option1" checked>
58) Option one is this and that—be sure to include why it's great
59) </label>
60) </div>
61)
62) <div class="form-check">
63) <label class="form-check-label">
64) <input type="radio" class="form-check-
    input" name="optionsRadios" id="optionsRadios2" value="option2">
65) Option two can be something else and selecting it will deselect option one
66) </label>
67) </div>
68)
69) <div class="form-check disabled">
70) <label class="form-check-label">
71) <input type="radio" class="form-check-
    input" name="optionsRadios" id="optionsRadios3" value="option3" disabled>
72) Option three is disabled
73) </label>
74) </div>
75)
76) </fieldset>
77)
78) <!-- CHECK BUTTON -->
79) <div class="form-check">
80) <label class="form-check-label">
81) <input type="checkbox" class="form-check-input">
82) Check me out
83) </label>
84) </div>
```



```
85) <button type="submit" class="btn btn-primary">Submit</button>  
86) </form>
```

**Note:** The <fieldset> tag is used to group related elements in a form.

### 3) Bootstrap: Navbars:

Navbars are nothing but Navigation Bars. Usually the navbars present on the top of the web page, which can be used for navigation purposes.

#### How to create Navbar Template:

```
<nav class="navbar navbar-default">  
</nav>
```

#### How to add Brand to the Navbar:

```
1) <nav class="navbar navbar-default">  
2)   <div class="navbar-header">  
3)     <a href="http://durgasoft.com" class="navbar-brand">DURGASOFT</a>  
4)   </div>  
5) </nav>
```

#### How to add remaining items to the Navbar:

The remaining items will be added in the form of unordered list. Each List item acts as Navbar item

```
1) <nav class="navbar navbar-default">  
2)   <div class="navbar-header">  
3)     <a href="http://durgasoft.com" class="navbar-brand">DURGASOFT</a>  
4)   </div>  
5)   <ul class="nav navbar-nav">  
6)     <li> <a href="#">Home</a></li>  
7)     <li> <a href="#">About Us</a></li>  
8)     <li> <a href="#">Gallary</a></li>  
9)     <li> <a href="#">Services</a></li>  
10)  </ul>  
11) </nav>
```

All these items will be added from Left Hand side



## How to add items to the Right Hand Side:

These items also will be added in the form of unordered list.

```
1) <nav class="navbar navbar-default">
2)   <div class="navbar-header">
3)     <a href="http://durgasoft.com" class="navbar-brand">DURGASOFT</a>
4)   </div>
5)   <ul class="nav navbar-nav">
6)     <li> <a href="#">Home</a></li>
7)     <li> <a href="#">About Us</a></li>
8)     <li> <a href="#">Gallery</a></li>
9)     <li> <a href="#">Services</a></li>
10)  </ul>
11)  <ul class="nav navbar-nav navbar-right">
12)    <li> <a href="#">Contact Us</a></li>
13)    <li> <a href="#">Logout</a></li>
14)  </ul>
15) </nav>
```

## How to position navbar items properly:

We have to take all items inside container class.

```
1) <nav class="navbar navbar-default">
2)   <div class="container">
3)     header/brand, items..
4)   </div>
5) </nav>
```

**Note:** It is not recommended to take total navbar inside container class

```
1) <div class="container">
2)   <nav class="navbar navbar-default">
3)     ...
4)   </nav>
5) </div>
```

## How to fix navbar always at top even in scroll down:

```
<nav class="navbar navbar-default navbar-fixed-top">
```

## How to inverse color of Navbar:

```
<nav class="navbar navbar-default navbar-fixed-top navbar-inverse">
```



## How to implement Hamburger to Navbar:

We have to enclose the collapsed items inside the following div tag.

```
1) <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
2)     <ul class="nav navbar-nav">
3)         <li><a href="#">Home</a></li>
4)         <li><a href="#">About Us</a></li>
5)         <li><a href="#">Services</a></li>
6)         <li><a href="#">Services</a></li>
7)         <li><a href="#">Services</a></li>
8)     </ul>
9)     <ul class="nav navbar-nav navbar-right">
10)        <li><a href="#">Contact Us</a></li>
11)        <li><a href="#">Logout</a></li>
12)    </ul>
13) </div>
```

To add hamburger, we have to add the following tag inside <div class="navbar-header">

```
1) <div class = "navbar-header">
2)     <button type = "button" class = "navbar-toggle collapsed"
3)         data-toggle="collapse" data-target="#bs-example-navbar-collapse-1"
4)         aria-expanded="false">
5)         <span class="sr-only">Toggle navigation</span>
6)         <span class="icon-bar"></span>
7)         <span class="icon-bar"></span>
8)         <span class="icon-bar"></span>
9)     </button>
10)    <a href="http://durgasoftonline.com" class="navbar-brand"> DURGASOFT</a>
11) </div>
```

### \*\*\*Note:

To work Bootstrap navbar hamburger, compulsory Bootstrap Javascript must be required. But to work Bootstrap Javascript, jQuery must be required. Hence we have to include Bootstrap Java script and jQuery CDNs in our html. But order is important first jquery cdn followed by javascript cdn

```
1) <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
2)
3) <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js" integrity = "sha384-
```



```
Tc5IQib027qvjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNlpG9mGCD8wGNlcPD7Txa"  
crossorigin="anonymous"></script>
```

We can write inside <body> tag anywhere.

## Demo HTML:

```
1) <!DOCTYPE html>  
2) <html lang="en" dir="ltr">  
3) <head>  
4) <!-- Latest compiled and minified CSS -->  
5) <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7  
/css/bootstrap.min.css" integrity="sha384-  
BVYiISiFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"  
crossorigin="anonymous">  
6) <meta charset="utf-8">  
7) <title></title>  
8) </head>  
9) <body>  
10) <nav class="navbar navbar-default navbar-fixed-top navbar-inverse">  
11) <div class="container">  
12) <div class="navbar-header">  
13) <button type="button" class="navbar-toggle collapsed"  
data-toggle="collapse" data-target="#bs-example-navbar-collapse-1"  
aria-expanded="false">  
14) <span class="sr-only">Toggle navigation</span>  
15) <span class="icon-bar"></span>  
16) <span class="icon-bar"></span>  
17) <span class="icon-bar"></span>  
18) <span class="icon-bar"></span>  
19) <span class="icon-bar"></span>  
20) </button>  
21) <a href="http://durgasoftonline.com" class="navbar-brand">DURGASOFT</a>  
22) </div>  
23) <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">  
24) <ul class="nav navbar-nav">  
25) <li><a href="#">Home</a></li>  
26) <li><a href="#">About Us</a></li>  
27) <li><a href="#">Services</a></li>  
28) <li><a href="#">Services</a></li>  
29) <li><a href="#">Services</a></li>  
30) </ul>  
31) <ul class="nav navbar-nav navbar-right">  
32) <li><a href="#">Contact Us</a></li>  
33) <li><a href="#">Logout</a></li>
```



```
34)         </ul>
35)     </div>
36) </div>
37) </nav>
38) <p></p><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>
    <br><br><br>
39) <nav class="navbar navbar-default">
40) <div class="container-fluid">
41) <!-- Brand and toggle get grouped for better mobile display -->
42) <div class="navbar-header">
43) <button type="button" class="navbar-toggle collapsed" data-toggle="collapse"
    data-target="#bs-example-navbar-collapse-1" aria-expanded="false">
44) <span class="sr-only">Toggle navigation</span>
45) <span class="icon-bar"></span>
46) <span class="icon-bar"></span>
47) <span class="icon-bar"></span>
48) </button>
49) <a class="navbar-brand" href="#">Brand</a>
50) </div>
51) <!-- Collect the nav links, forms, and other content for toggling -->
52) <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
53) <ul class="nav navbar-nav">
54) <li class="active"><a href="#">Link <span class="sr-only">
    (current)</span></a></li>
55) <li><a href="#">Link</a></li>
56) <li class="dropdown">
57) <a href="#" class="dropdown-toggle" data-toggle="dropdown"
    role="button" aria-haspopup="true" aria-expanded="false">
    Dropdown <span class="caret"></span></a>
58) <ul class="dropdown-menu">
59) <li><a href="#">Action</a></li>
60) <li><a href="#">Another action</a></li>
61) <li><a href="#">Something else here</a></li>
62) <li role="separator" class="divider"></li>
63) <li><a href="#">Separated link</a></li>
64) <li role="separator" class="divider"></li>
65) <li><a href="#">One more separated link</a></li>
66) </ul>
67) </li>
68) </ul>
69) <form class="navbar-form navbar-left">
70) <div class="form-group">
71) <input type="text" class="form-control" placeholder="Search">
72) </div>
73) <button type="submit" class="btn btn-default">Submit</button>
```





```
74) </form>
75) <ul class="nav navbar-nav navbar-right">
76) <li><a href="#">Link</a></li>
77) <li class="dropdown">
78) <a href="#" class="dropdown-toggle" data-toggle="dropdown"
   role="button" aria-haspopup="true" aria-expanded="false">
   Dropdown <span class="caret"></span></a>
79) <ul class="dropdown-menu">
80) <li><a href="#">Action</a></li>
81) <li><a href="#">Another action</a></li>
82) <li><a href="#">Something else here</a></li>
83) <li role="separator" class="divider"></li>
84) <li><a href="#">Separated link</a></li>
85) </ul>
86) </li>
87) </ul>
88) </div><!-- /.navbar-collapse -->
89) </div><!-- /.container-fluid -->
90) </nav>
91) <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
   q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
   crossorigin="anonymous"></script>
92) <!-- Latest compiled and minified JavaScript -->
93) <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.j
   s" integrity="sha384-
   Tc5lQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNlPG9mGCD8wGNlCPD7Txa"
   crossorigin="anonymous"></script>
94) </body>
95) </html>
```

## 4) Bootstrap: Grid System

- Grid System is the most fundamental and commonly used concept of Bootstrap.
- To display our application layout properly on multiple devices of multiple screen sizes like desktop, laptop, tab, mobile etc, we should go for grid system.
- In Bootstrap we can split total screen into 12 columns. Within this 12 columns length we can take any number of elements of same size or different sizes.

**Eg:** 12X1 → 12 elements and each element is of 1 column length

6X2 → 6 elements and each element is of 2 column length

4X3 → 4 elements and each element is of 3 column length

e1X3+e2X4+e3X5 → total 3 elements

First element is of 3 columns length

Second element is of 4 columns length

Third element is of 5 columns length



## How to implement Grid:

We can implement grid by using 2 classes

1. row class: to define row

```
<div class="row">
```

2. Within the row we can define columns by using the following class  
col-screenSize-noOfColumns

The allowed screen sizes in Bootstrap 3 are:

lg → Large Size (like Desktop screens)

md → Medium Size (like Laptop screens)

sm → Small Size (like Tab screens)

xs → Extra Small Size (like Mobile screens)

But in Bootstrap-4 Extra Large(xl) is also allowed.

```
1) <div class="row">
2)   <div class="col-lg-3">Element-1</div>
3)   <div class="col-lg-3">Element-1</div>
4)   <div class="col-lg-3">Element-1</div>
5)   <div class="col-lg-3">Element-1</div>
6) </div>
```

Here row contains 4 elements and each element taking 3 columns length.

### demo.html:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4)   <link rel="stylesheet" href="Bootstrap.css">
5)   <meta charset="utf-8">
6)   <title></title>
7)   <style>
8)     .box{
9)       background:orange;
10)      border:1px solid blue;
11)    }
12)  </style>
13) </head>
14) <body>
15)   <div class="container">
16)     <div class="row">
17)       <div class="col-lg-3 box">Element-1</div>
```



```
18) <div class="col-lg-3 box">Element-2</div>
19) <div class="col-lg-3 box">Element-3</div>
20) <div class="col-lg-3 box">Element-4</div>
21) </div>
22) </div>
23) </div>
24) </body>
25) </html>
```

We can take elements of different column lengths also

```
1) <div class="row">
2) <div class="col-lg-3 box">Element-1</div>
3) <div class="col-lg-6 box">Element-2</div>
4) <div class="col-lg-3 box">Element-3</div>
5) </div>
```

**Note:** The total column length per row should be:12

**Case-1:** For medium or large screens each row should contains 4 elements

```
1) <div class="row">
2) <div class="col-md-3 box">Element-1</div>
3) <div class="col-md-3 box">Element-2</div>
4) <div class="col-md-3 box">Element-3</div>
5) <div class="col-md-3 box">Element-4</div>
6) </div>
```

**Case-2:** On every screen (ls,md,sm,xs) row should contain 3 elements

```
1) <div class="row">
2) <div class="col-xs-4 box">Element-1</div>
3) <div class="col-xs-4 box">Element-2</div>
4) <div class="col-xs-4 box">Element-3</div>
5) </div>
```

xs means either extra small or higher

## How to define Grid for multiple screens simultaneously:

For Large screens 6 Elements

For Medium Screens 4 Elements

For Small Screens 3 Elements

For Extra Small Screens 2 Elements



```
1) <div class="row">
2)   <div class="col-lg-2 col-md-3 col-sm-4 col-xs-6 box">Element-1</div>
3)   <div class="col-lg-2 col-md-3 col-sm-4 col-xs-6 box">Element-2</div>
4)   <div class="col-lg-2 col-md-3 col-sm-4 col-xs-6 box">Element-3</div>
5)   <div class="col-lg-2 col-md-3 col-sm-4 col-xs-6 box">Element-4</div>
6)   <div class="col-lg-2 col-md-3 col-sm-4 col-xs-6 box">Element-5</div>
7)   <div class="col-lg-2 col-md-3 col-sm-4 col-xs-6 box">Element-6</div>
8) </div>
```

## Nested Grids:

Inside a grid, we can define another grid, which is nothing but Nested Grid.

Nested Grid: Grid inside Grid

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3)   <head>
4)     <link rel="stylesheet" href="Bootstrap.css">
5)     <meta charset="utf-8">
6)     <title></title>
7)     <style>
8)       .box{
9)         background:orange;
10)        border:1px solid blue;
11)      }
12)    </style>
13)  </head>
14)  <body>
15)    <div class="container">
16)      <div class="row">
17)        <div class="col-lg-6 box">
18)          <div class="row">
19)            <div class="col-lg-4 box">Nested Element-1</div>
20)            <div class="col-lg-4 box">Nested Element-2</div>
21)            <div class="col-lg-4 box">Nested Element-3</div>
22)          </div>
23)        </div>
24)        <div class="col-lg-6 box">Element-2 </div>
25)      </div>
26)    </div>
27)  </body>
28) </html>
```



## Bootstrap Photo Gallery Application:

**Step-1:** Add CSS, jQuery and JavaScript cdn files to our HTML inside the <head> tag

```
1) <head>
2) <!-- To link our css to HTML -->
3) <link rel="stylesheet" href="gallery2.css">
4)
5) <!-- Latest compiled and minified CSS -->
6) <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/
  /css/bootstrap.min.css" integrity="sha384-
  BVYiISiFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" c
  rossorigin="anonymous">
7)
8) <!-- For Font Awesome (icons) -->
9) <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.1.0/css/
  all.css" integrity="sha384-
  IKuwvrZot6UHsBSfcMvOkWwlCMgc0TaWr+30HWe3a4ltaBwTZhyTEggF5tJv8tbt" cr
  ossorigin="anonymous">
10) </head>
```

**Step-2:** Add jumbotron to our HTML inside <body> tag

```
1) <div class="container">
2) <div class="jumbotron">
3) <h1> Heaven Photo Gallery</h1>
4) <p>This is short cut to go to heaven... already crores of people tested... why don't y
  ou test</p>
5) </div>
6) </div>
```

**Step-3:** Add camera icon (fontawesome) inside <h1> tag of jumbotron

<h1><i class="fas fa-camera"></i> Heaven Photo Gallery</h1>

**Step-4:** Add Navigation Bar to the html at the top

```
1) <nav class="navbar navbar-default navbar-fixed-top " id="xxx">
2) <div class="container">
3) <div class="navbar-header">
4) <a href="http://durgasoftonline.com" class="navbar-brand" id="yyy">
  DURGABAR</a>
5) </div>
6) <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
7) <ul class="nav navbar-nav">
```



```
8) <li><a href="#">Home</a></li>
9) <li><a href="#">About Us</a></li>
10) <li><a href="#">Services</a></li>
11) </ul>
12) <ul class="nav navbar-nav navbar-right">
13) <li><a href="#">Contact Us</a></li>
14) <li><a href="#">Logout</a></li>
15) </ul>
16) </div>
17) </div>
18) </nav>
```

**Step-5:** Add picture icon (fontawesome) inside <a> tag of navbar for our brand  
<a href="http://durgasoftonline.com" class="navbar-brand" id="yyy"><span  
class="glyphicon glyphicon-picture" aria-hidden="true"></span> DURGABAR</a>

**Step-6:** Add hamburger button to our navbar inside navbar-header class <div> tag at the top for responsive navbar

```
1) <button type="button" class="navbar-toggle collapsed" data-
toggle="collapse" data-target="#bs-example-navbar-collapse-1" aria-
expanded="false">
2) <span class="sr-only">Toggle navigation</span>
3) <span class="icon-bar"></span>
4) <span class="icon-bar"></span>
5) <span class="icon-bar"></span>
6) <span class="icon-bar"></span>
7) </button>
```

**Step-7:** Add Grid after jumbotron's <div> tag

```
1) <div class="row">
2) <div class="col-lg-4 col-sm-6">
3) <div class="thumbnail"> </div>
4) </div>
5) <div class="col-lg-4 col-sm-6">
6) <div class="thumbnail"> </div>
7) </div>
8) <div class="col-lg-4 col-sm-6">
9) <div class="thumbnail"> </div>
10) </div>
11) <div class="col-lg-4 col-sm-6">
12) <div class="thumbnail"> </div>
13) </div>
14) <div class="col-lg-4 col-sm-6">
```



```
15) <div class="thumbnail"> </div>
16) </div>
17) <div class="col-lg-4 col-sm-6">
18) <div class="thumbnail"> </div>
19) </div>
20) <div class="col-lg-4 col-sm-6">
21) <div class="thumbnail"> </div>
22) </div>
23) <div class="col-lg-4 col-sm-6">
24) <div class="thumbnail"> </div>
25) </div>
26) <div class="col-lg-4 col-sm-6">
27) <div class="thumbnail"> </div>
28) </div>
29) </div>
```

## gallery2.css

### Step-8: Add padding in between navbar and jumbotron

```
1) body{
2) padding-top: 70px;
3) }
```

### Step-9: Add css styles to navbar

```
1) .navbar{
2) /* just remove navbar-default from navbar */
3) background: blue;
4) }
5) .navbar a{
6) color:white;
7) }
```

### Step-10: Add css styles to jumbotron

```
1) .container .jumbotron {
2) background:green;
3) color:white;
4) }
```



## How to add Bootstrap Glyphicons to HTML:

Select the following tag and add to the html where ever it is required.

- 1) `<span class="glyphicon glyphicon-camera" aria-hidden="true"></span>`
- 2) `<span class="glyphicon glyphicon-picture" aria-hidden="true"></span>`

Class names will be changed from icon to icon

## How to change jumbotron background color and text color:

- 1) `.jumbotron{`
- 2) `background-color: red;`
- 3) `color: blue;`
- 4) `}`

If it is not working then right click and inspect element on the element and choose the corresponding style.

- 1) `.container .jumbotron, .container-fluid .jumbotron {`
- 2) `padding-right: 15px;`
- 3) `padding-left: 15px;`
- 4) `border-radius: 6px;`
- 5) `background:red;`
- 6) `}`

## How to add icons from fontawesome.com:

<https://fontawesome.com/icons?from=io>

**Step-1:** Add CDN to our html:

```
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.1.0/css/all.css"
integrity=
"sha384-lKuWvrZot6UHsBSfcMvOkWwlCMgc0TaWr+30HWe3a4ltaBwTZhyTEggF5tJv8tbt"
crossorigin="anonymous">
```

**Step-2:** Add i tag where ever icon is required

```
<i class="fas fa-camera"></i>
```

user-plus

icon to icon only class name will be changed (eg camera)





## Durga Dating App:

### dating.html

**Step-1:** Add all CDNs(Bootstrap,jQuery,JavaScript) and css to our html

```
1) <head>
2)   <link rel="stylesheet" href="dating.css">
3)   <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.1.0/css/
    all.css" integrity="sha384-
    lKuwvvrZot6UHsBSfcMvOkWwlCMgc0TaWr+30HWe3a4ltaBwTZhyTEggF5tJv8tbt"
    crossorigin="anonymous">
4)   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7
    /css/bootstrap.min.css" integrity="sha384-
    BVYiISiFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
    crossorigin="anonymous">
5) </head>
```

**Step-2:** Add Grid to our HTML with h1,h3 and hr tag

```
1) <div class="container">
2)   <div class="row">
3)     <div class="col-lg-12">
4)       <div id="content">
5)         <h1 id="tc1">DURGA Dating App</h1>
6)         <h3 id="tc">Only for Senious Citizens and Kids... Just for fun...</h3>
7)         <hr>
8)       </div>
9)     </div><!-- End col-lg-12 -->
10)  </div><!-- End row -->
11) </div><!-- End container -->
```

**Step-3:** Add button to grid after <hr> tag

`<button type="button" name="button" class="btn btn-default btn-lg"> Get Started</button>`

**Step-4:** Add fontawesome smile before "Get Started"

`<button type="button" name="button" class="btn btn-default btn-lg"><i class="fas fa-smile"></i>&nbsp;Get Started</button>`  
Here &nbsp; -->for one space



## Step-5: Add navbar to HTML at the top

```
1) <nav class="navbar navbar-default">
2)   <div class="container">
3)     <!-- Brand and toggle get grouped for better mobile display -->
4)     <div class="navbar-header">
5)       <button type="button" class="navbar-toggle collapsed" data-
toggle="collapse" data-target="#bs-example-navbar-collapse-1" aria-
expanded="false">
6)       <span class="sr-only">Toggle navigation</span>
7)       <span class="icon-bar"></span>
8)       <span class="icon-bar"></span>
9)       <span class="icon-bar"></span>
10)    </button>
11)    <a class="navbar-brand" href="#">DURGA Dating App</a>
12)  </div>
13)  <!-- Collect the nav links, forms, and other content for toggling -->
14)  <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
15)    <ul class="nav navbar-nav">
16)      <li class="active"><a href="#">Home</a></li>
17)      <li><a href="#">About us</a></li>
18)      <li><a href="#">Services</a></li>
19)    </ul>
20)    <ul class="nav navbar-nav navbar-right">
21)      <li><a href="#">Sign up</a></li>
22)      <li><a href="#">Login</a></li>
23)    </ul>
24)  </div><!-- /.navbar-collapse -->
25) </div><!-- /.container-fluid -->
26) </nav>
```

## Step-6: Add fontawesomes for Sign up and Login

```
<li><a href="#">Sign up<span class="fas fa-user-plus"> </span></a></li>
<li><a href="#">Login <span class="fas fa-sign-in-alt"> </span></a></li>
```

## dating.css:

## Step-7: Add css styles for Grid's content

```
1) #content {
2)   text-align: center;
3)   padding-top: 25%;
4) }
```



## Step-8: Add background image and it's css styles

```
1) body{
2)  background: url(https://images.unsplash.com/photo-1529610453335-
   db84df315b29?ixlib=rb-
   0.3.5&ixid=eyJhchBfaWQiOjE5MDd9&s=1336a860de7a33d2f9b4281a13a48b68&a
   uto=format&fit=crop&w=400&q=60);
3)  background-size: cover;
4)  background-position: center;
5)  background-repeat: no-repeat;
6) }
```

## Step-9: Add css styles for <h1> tag using tc1 id

```
1) #tc1{
2)  color:white;
3)  font-weight: 700;
4)  font-size: 5em;
5) }
```

## Step-10: Add css styles for <h3> tag using tc id

```
1) #tc{
2)  color:white;
3) }
```

## Step-11: Add 100% height for remove bottom space while xs size

```
1) html{
2)  height: 100%
3) }
```

## Step-12: Add css styles for hr tag

```
1) hr{
2)  width: 500px;
3)  border-bottom: 10px solid blue;
4)  border-top: 10px solid red;
5) }
```

To get images  
unsplash.com

<i class="fas fa-user-plus"></i>



# JavaScript

## Study Material



# JavaScript

**HTML is for Nouns**

Like Anushka, input fields, buttons, forms etc

**CSS is for Adjectives**

Like Styling with colors, borders, background images etc

**Java Script is for Verbs/Actions**

Like Dance, Eat....

**Java Script is Full pledged Programming Language.**

The main purpose of java script is to add functionality (actions) to the HTML.

Usually we can Java Script in the Front-end and we can use Node JS for Back-end.

## Agenda

- 1) Java Script Developer's Console
- 2) The 5 Basic Javascript Primitive Data Types
- 3) Declaring variables with var keyword
- 4) The 3 most commonly used Java Script Functions



# **1) JavaScript Developer's Console**

We can use this developer's console to test our java script coding snippets. This is just for testing purpose only and usually not recommended for main coding.

## **How to Launch JavaScript Console:**

Browser → Right Click → Inspect → Console

Short-cut: Ctrl+Shift+j

### **Note:**

- 1) To clear console we have to use clear() function.
- 2) ; at end of statement is not mandatory in newer versions.

# **2) The 5 Basic JavaScript Primitive Data Types**

Java Script defines the following 5 primitive data types

## **1) Numbers:**

- 10
- -10
- 10.5
  
- All these are of "number" type
- Java Script never cares whether it is integral or float-point or signed and unsigned.
  
- General Mathematical operators are applicable for numbers
  
- 10+20
- 10-20
- 10/20
- 10\*20
- 10%3
- 10\*\*2
  
- General Operator precedence also applicable.
- 10+20\*3 → 70
- 10\*(2+3) → 50

**Note:** We can check the type of variable by using typeof keyword  
typeof x;



## 2) string:

- Any sequence of characters within either single quotes or double quotes is treated as string.
- 'durga'
- "durga"
- We can apply + operator for Strings also and it acts as concatenation operator.
- Rule:** If both arguments are number type then + operator acts as arithmetic addition operator.
- If atleast one argument is of string type then + operator acts as concatenation operator.
- 10+20 → 30
- 'durga'+10 → durga10
- 'durga'+true → durgatrue
- We can use escape characters also in the string.  
Eg: 'durga\nsoft'  
'durga\tsoft'  
'This is \' symbol'  
'This is \'\" symbol'  
'This is \\' symbol'

### Q) How to find the number of characters present in the string?

We can find by using length variable

Eg: 'durgasoft'.length

### Q) How to access characters of the String?

By using index

index of first character is zero

'durga'[2] → r

'durga'[200] → undefined but no error

'durga'[-1] → undefined

**Note:** If we are trying to access string elements with out of range index or negative index then we will get undefined value and we won't get any Error.

## 3) boolean:

The only allowed values are: true and false (case sensitive)



### 3) JavaScript Variables

Variables are containers to store values.

**Syntax:** var variableName=variableValue

**Eg:** var name = "durga"

var age = 60

var isMarried = false

**Note:** JavaScript variables follow CamelCase Convention

studentMobileNumber → Camel case(Java,JavaScript etc)

student\_mobile\_number → Snake Case(Python )

student-mobile-number → Kebab Case(Lisp)

Based on provided value automatically type will be considered for variables.

**Eg:** var x = 10

typeof x → number

x = false

typeof x → boolean

Hence Java Script is dynamically Typed Programming Language like Python

#### null and undefined:

Variables that are declared but not initialized, are considered as undefined

**Eg:** var x;

typeof x → undefined

null value means nothing.

If the value of a variable null means that variable not pointing to any object.

var currentPlayer = 'durga'

currentPlayer = null //game over

### 4) The 3 most commonly used methods of Java Script

#### 1) alert():

To display alerts to the end user

alert('Hello there')

alert(100000)

alert(10.5)





## 2) console.log():

To print messages to the developer's console

Eg: `console.log('Hello there')`

`console.log(10*20)`

These console message not meant for end user.

## 3) prompt():

- To get input from the end user  
`prompt('What is Your Name:')`
- Here we are not saving the value for the future purpose. But we can save as follows  
`var name= prompt('What is Your Name:')`
- Based on our requirement we can use this name  
`console.log('Hello '+name+' Good Evening')`  
`alert('Hello '+name+' Good Evening')`

## How to write JavaScript to a separate File and connect to HTML:

### demo.js:

`alert('Hello everyone good evening')`

### html:

- We can link javascript file to html by using the following `<script>` tag.
- `<script type="text/javascript" src="demo.js"></script>`
- We can take this script tag either inside head tag or body tag. If we are taking inside head tag then javascript code will be executed before processing body.
- If we are taking inside body tag then javascript code will be executed as the part of body execution.

## Demo Application: Age and Death Calculator

### demo.js:

```
1) var name=prompt('Enter Your Name:');
2) var age=prompt('Enter Your Age:');
3) agedays=age*365.25
4) remainingdays=(60-age)*365.25;
5) alert("Hello "+name+"...\nYour Current Age:"+agedays+" days\nYou will be there on the earth only "+remainingdays+" days. No one can change including God also");
```



## demo.html:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4)   <meta charset="utf-8">
5)   <title></title>
6) </head>
7) <body>
8)   <h1>The power of Java Script</h1>
9)   <script type="text/javascript" src="demo.js"> </script>
10) </body>
11) </html>
```

## Operators:

1) Arithmetic Operators: +, -, \*, /, %, \*\*

2) Comparison Operators: <, <=, >, >=, ==, !=, ===, !==

10<20 → true  
10<=20 → true  
10>20 → false  
10>=20 → false  
10==20 → false  
10 != 20 → true

### Difference between == and ===:

In the case of == operator internally type coercion will be performed. Hence if arguments are different types first both arguments will be converted to same type and then comparison will be performed. Hence both arguments need not be same type.

Eg: 10 == "10" → true  
10 == 10 → true

Here only values are important but not types.

But in the case === operator, type coercion won't be performed. Hence argument types must be same, otherwise we will get false.

10 === 10 → true  
10 === "10" → false

Here both content and type are important.



## Note:

== → Normal equality operator where types are not important but values must be same

=== → Strict equality operator where both types and values must be same

It is recommended to use === operator because it is more safer and more specific.

## Example:

true == "1" → true

false == "0" → true

null == undefined → true

true === "1" → false

false === "0" → false

null === undefined → false

## NaN (Not a Number):

If the result is undefined then we will get NaN

Eg: 0/0 → NaN

For any x value including NaN the following expressions returns false

- x < NaN
- x <= NaN
- x > NaN
- x >= NaN
- x == NaN

For any x value including NaN the following expression returns true x != NaN

Eg: NaN == NaN → false

NaN != NaN → true

## 3) Logical Operators:

- && → AND
- || → OR
- ! → Not
  
- X && Y → If both arguments are true then only result is true. i.e if atleast one argument is false then the result is always false
  
- X || Y → If atleast one argument is true then the result is true. i.e if both arguments are false then only result is false.

Note: For Logical Operators



- 1) Zero Value always treated as false  
non-zero value always treated as true
- 2) Empty string treated as false where as non-empty string treated as true
- 3) null, undefined, NaN are treated as false

### Examples:

```
var x =10;  
var y =20;
```

`x<10 && x != 5` → false

`y>9 || x== 10` → true

`!(x==y)` → true

`!(x=="10" || x=== y) && !( y!= 8 && x<=y))` → false

`!(x !==1)&& y === "20"` → false

## Conditional Statements:

Based on available options, one option will be selected and executed in conditional statements/selection statements.

- 1) if
- 2) if else
- 3) else if

### Syntax:

```
if(b){  
    action if b is true;  
}  
else{  
    action if b is false;  
}
```

**Eg 1:** Write Java Script code to check given number is even or not?

### demo.html:

- 1) `<!DOCTYPE html>`
- 2) `<html lang="en" dir="ltr">`
- 3) `<head>`



```
4) <meta charset="utf-8">
5) <script type="text/javascript" src="demo.js"></script>
6) <title></title>
7) </head>
8) <body>
9) <h1>The power of Java Script</h1>
10) </body>
11) </html>
```

## demo.js:

```
1) var num = Number(prompt('Enter Any Number:'))
2) if(num%2===0){
3)   console.log('Given Number is Even')
4)   alert('Given Number is Even')
5) }
6) else{
7)   console.log('Given Number is Odd')
8)   alert('Given Number is Odd')
9) }
```

## Q) Write Java Script code to print proper meaningful message based on provided age regarding matrimonial website?

### demo.js:

```
1) var age = Number(prompt('Enter Your Age:'))
2) if(age>60){
3)   alert('Plz wait some more time..Defenitely You will get Best Match')
4) }
5) else if(age<18){
6)   alert("Your age already crossed marriage age..No chance of getting marriage")
7) }
8) else{
9)   alert('Thanks for registration..You will get match details soon by email')
10) }
```

## Q) Write Java Script code to print proper meaningful message based on provided brand regarding beer application?

```
1) var brand = prompt('Enter Your Favourite Brand:')
2) if(brand=="KF"){
3)   alert("It is Children's Brand")
4) }
```



```
5) else if(brand=="KO"){
6)   alert("It is too light")
7) }
8) else if(brand=="RC"){
9)   alert("It is not that much kick")
10) }
11) else if(brand=="FO"){
12)   alert("Buy One get One FREE")
13) }
14) else{
15)   alert('Other brands are not recommended')
16) }
```

## Q) Number Guess Application

```
1) var sno=4
2) var num = Number(prompt('Enter your guess between 1 to 9:'))
3) if(num>sno){
4)   alert("It is too high..Guess again")
5) }
6) else if(num<sno){
7)   alert("It is too low guess again")
8) }
9) else{
10)  alert('Your Guess Matched')
11) }
```

## Iterative Statements:

If we want to execute a group of statements iteratively, then we should go for iterative statements.

**DRY Principle: Don't Repeat Yourself**

It is highly recommended to follow DRY principle, otherwise it increases development time

It increases length of the code and reduces readability

In JavaScript there are 2 types of iterative statements

- 1) While Loop
- 2) For Loop



## 1) While Loop:

As long as some condition is true execute code then we should go for while loop.

### Syntax:

```
while(condition){  
    body  
}
```

**Eg 1:** To print Hello 10 times to the console

### demo.js:

```
1) var count=1  
2) while(count<=10){  
3)   console.log("Hello")  
4)   count++  
5) }
```

**Eg 2:** To print first 10 numbers

### demo.js:

```
1) var count=1  
2) while(count<=10){  
3)   console.log(count)  
4)   count++  
5) }
```

**Eg 3:** To print each character present inside a string?

### demo.js:

```
1) var s="durga"  
2) var i =0  
3) while(i<s.length){  
4)   console.log(s[i])  
5)   i++  
6) }
```



**Eg 4:** To print all numbers divisible by 3 AND 5 between 5 and 100?

**demo.js:**

```
1) var n=5
2) while(n<=100){
3)   if(n%3==0 && n%5==0){
4)     console.log(n)
5)   }
6)   n++
7) }
```

**Eg 4:** Write program to read actress name from the end user until entering 'sunny' by using while loop.

```
1) var name=prompt("Enter Your Favourite Actress:")
2) while(name !== "sunny"){
3)   name=prompt("Enter Your Favourite Actress:")
4) }
5) alert("Thanks for Confirmation as your Favourite actress: Sunny")
```

**Note:** If we don't know the number of iterations in advance and if we want to execute body as long as some condition is true then we should go for while loop.

## 2) Iterative Statements : For Loop

If we know the number of iterations in advance then we should use for loop.

**Syntax:**

```
for(initialization section; conditional check; increment/decrement section)
{
  body;
}
```

**Eg 1:** To print "Hello" 10 times

```
for(var i=0;i<10;i++){
  console.log("Hello");
}
```

**Eg 2:** To print First 1 to 10

```
for(var i=1;i<=10;i++){
  console.log(i);
}
```





**Eg 3:** To print all numbers which are divisible by 7 from 1 to 100:

```
for(var i=1;i<=100;i++){  
  if(i%7==0){  
    console.log(i)  
  }  
}
```

**Eg 4:** To print each character from the given string

```
var word=prompt("Enter Some Word:")  
for(var i=0;i<word.length;i++){  
  console.log(word[i])  
}
```

## while vs for loop:

- If we don't know the number of iterations in advance and as long as some condition is true keep on execute body then we should go for while loop.
- If we know the number of iterations in advance then we should use for loop.

## Secret Agent Application:

### Rules:

- 1) The first character of Name should be 'd'
- 2) The Last character of Favourite Actor should be 'r'
- 3) The lucky number should be 7
- 4) The length of the dish should be  $\geq 6$

If the above conditions are satisfied then user is valid secret agent and share information about operation, otherwise just send thanks message.

### demo.js:

```
1) var name=prompt("Enter Your Name:")  
2) var actor=prompt("Enter Your Favourite Actor:")  
3) var lucky=prompt("Enter Your Lucky Number:")  
4) var dish=prompt("Enter Your Favourite Dish:")  
5)  
6) var nameConition=false  
7) var actorCondition=false  
8) var luckyConition=false  
9) var dishConition=false  
10)  
11) if(name[0]=="d") {  
12) nameConition=true
```



```
13) }
14) if(actor[actor.length-1]=="r"){
15)  actorCondition=true
16) }
17) if(lucky==7){
18)  luckyConition=true
19) }
20) if(dish.length>=6){
21)  dishConition=true
22) }
23) alert("Hello:"+name+"\nThanks For Your Information")
24) if(nameConition && actorCondition && luckyConition && dishConition){
25)  console.log("Hello Secret Agent our next operation is:")
26)  console.log("We have to kill atleast 10 sleeping students in the class room b'z thes
    e are burdent to country")
27) }
```



# FUNCTIONS

- If any piece of code repeatedly required in our application, then it is not recommended to write that code separately every time. We have to separate that code into a function and we can call that function wherever it is required.
- Hence the main advantage of functions is code Reusability.

## Syntax of JavaScript Function:

```
function functionName(arguments){  
    body  
    return value;  
}
```

**Eg 1:** To print "Good Morning" message

```
1) function wish(){  
2)   console.log("Good Morning!!!")  
3) }  
4) wish()  
5) wish()  
6) wish()
```

## Functions with Arguments:

A function can accept any number of arguments and these are inputs to the function. Inside function body we can use these arguments based on our requirement.

**Eg:** Write a function to accept user name as input and print wish message.

```
1) function wish(name){  
2)   console.log("Hello "+name+" Good Morning!!!")  
3) }  
4)   
5) var name= prompt("Enter Your Name:")  
6) wish(name)
```



## Functions with Default Arguments:

We can provide default values for arguments. If we are not passing any value then only default values will be considered.

```
1) function wish(name="Guest"){  
2)   console.log("Hello "+name+" Good Morning!!!")  
3) }  
4)  
5) wish("Durga")  
6) wish()
```

## Function with Return Values:

A function can return values also.

**Eg:** Write a Javascript function to take a number as argument and return its square value

```
1) function square(num){  
2)   return num*num;  
3) }  
4) var result=square(4)  
5) console.log("The Square of 4:"+result)  
6) console.log("The Square of 5:"+square(5))
```

**Eg:** Write a Javascript function to take 2 numbers as arguments and return sum.

```
1) function sum(num1,num2){  
2)   return num1+num2;  
3) }  
4) var result=sum(10,20)  
5) console.log("The sum of 10,20 :"+result)  
6) console.log("The sum of 100,200 :"+sum(100,200))
```

**Eg:** Write a Javascript function to take a string as argument and return Capitalized string.

```
1) function capitalize(str){  
2)   return str[0].toUpperCase()+str.slice(1);  
3) }  
4) console.log(capitalize('sunny'))  
5) console.log(capitalize('bunny'))
```



**Eg:** Write a Javascript function to check whether the given number is even or not?

```
1) function isEven(num){  
2)   if(num%2==0){  
3)     return true;  
4)   }  
5)   else{  
6)     return false;  
7)   }  
8) }  
9) console.log(isEven(15))  
10) console.log(isEven(10))
```

**Eg:** Write javascript function to find factorial of given number?

```
1) function factorial(num){  
2)   result=1;  
3)   for (var i = 2; i <= num; i++) {  
4)     result=result*i;  
5)   }  
6)   return result;  
7) }  
8) console.log("The Factorial of 4 is:"+factorial(4))  
9) console.log("The Factorial of 5 is:"+factorial(5))
```

**Eg:** Write a JavaScript Function to convert from Snake Case to Kebab Case of given String.

Snake Case: total\_number

Kebab Case: total-number

```
1) function snakeToKebab(str){  
2)   var newstring=str.replace('_', '-')  
3)   return newstring;  
4) }  
5) console.log(snakeToKebab('total_number'))
```

**Note:** Inside function if we are writing any statement after return statement, then those statements won't be executed, but we won't get any error.

```
1) function square(n){  
2)   return n*n;  
3)   console.log("Function Completed!!!")  
4) }  
5) console.log(square(4));
```

**Output:** 16



## JavaScript Scopes:

In Javascript there are 2 scopes.

- 1) Global Scope
- 2) Local Scope

### 1) Global Scope:

The variables which are declared outside of function are having global scope and these variables are available for all functions.

```
1) var x=10
2) function f1(){
3)   console.log(x);
4) }
5) function f2(){
6)   console.log(x);
7) }
8) f1();
9) f2();
```

### 2) Local Scope:

The variables which are declared inside a function are having local scope and are available only for that particular function. Outside of the function we cannot use these local scoped variables.

```
1) function f1(){
2)   var x=10
3)   console.log(x); //valid
4) }
5) f1();
6) console.log(x); //Uncaught ReferenceError: x is not defined
```

Eg:

```
1) var x=10
2) function f1(){
3)   x=777;
4)   console.log(x);
5) }
6) function f2(){
7)   console.log(x);
8) }
9) f1();
10) f2();
```



## Output:

777

777

```
1) var x=10
2) function f1(){
3)   x=777;
4)   console.log(x);
5) }
6) function f2(){
7)   console.log(x);
8) }
9) f2();
10) f1();
```

## Output:

10

777

```
1) var x=10
2) function f1(){
3)   var x=777;
4)   console.log(x);
5) }
6) function f2(){
7)   console.log(x);
8) }
9) f1();
10) f2();
```

## Output:

777

10

**Q) If Local and Global Variables having the Same Name then within the Function Local Variable will get Priority. How to access Global Variable?**

## **Higher Order Functions:**

We can pass a function as argument to another function. A function can return another function. Such type of special functions are called Higher Order Functions.

**Eg:** setInterval()



```
setInterval(function, time_in_milliseconds)
```

The provided function will be executed continuously for every specified time.

```
setInterval(singAsong, 3000)
```

singAsong function will be executed for every 3000 milli seconds.

We can stop this execution by using `clearInterval()` function.

Eg: demo.js

```
1) function singAsong(){  
2)   console.log('Rangamma...Mangamma..')  
3)   console.log('Jil..Jil...Jigel Rani..')  
4) }
```

On developer's console:

```
setInterval(singAsong,3000)
```

```
1
```

```
demo.js:2 Rangamma...Mangamma..
```

```
demo.js:3 Jil..Jil...Jigel Rani..
```

```
demo.js:2 Rangamma...Mangamma..
```

```
demo.js:3 Jil..Jil...Jigel Rani..
```

```
demo.js:2 Rangamma...Mangamma..
```

```
demo.js:3 Jil..Jil...Jigel Rani..
```

```
clearInterval(1)
```

```
undefined
```

## Anonymous Functions:

- Some times we can define a function without name, such type of nameless functions are called anonymous functions.
- The main objective of anonymous functions is just for instant use (one time usage)

Eg:

```
setInterval(function(){console.log("Anonymous Function");},3000);
```

```
8
```

```
Anonymous Function
```

```
Anonymous Function
```

```
Anonymous Function
```

```
Anonymous Function
```

```
..
```

```
clearInterval(8);
```





## Coding Examples from codebat.com:

### Problem-1: sleep\_in

Write a function called sleep\_in that takes 2 boolean parameters: weekday and vacation.

The parameter weekday is True if it is a weekday, and the parameter vacation is True if we are on vacation. We sleep in if it is not a weekday or we're on vacation. Return True if we sleep in.

sleep\_in(false, false) → true

sleep\_in(true, false) → false

sleep\_in(false, true) → true

```
1) function sleep_in(weekday,vacation) {  
2)   return !weekday || vacation;  
3) }  
4) console.log("Is Employee Sleeping:"+sleep_in(true,true))  
5) console.log("Is Employee Sleeping:"+sleep_in(true,false))  
6) console.log("Is Employee Sleeping:"+sleep_in(false,true))  
7) console.log("Is Employee Sleeping:"+sleep_in(false,false))
```

### Problem-2: monkey\_trouble

We have two monkeys, a and b, and the parameters a\_smile and b\_smile indicate if each is smiling. We are in trouble if they are both smiling or if neither of them is smiling. Return True if we are in trouble.

monkey\_trouble(true, true) → true

monkey\_trouble(false, false) → true

monkey\_trouble(true, false) → false

#### Solution:

```
1) function monkey_trouble(aSmile,bSmile){  
2)   return (aSmile && bSmile) || (!aSmile && !bSmile)  
3) }  
4) console.log("Is Person In Trouble:"+monkey_trouble(true,true))  
5) console.log("Is Person In Trouble:"+monkey_trouble(true,false))  
6) console.log("Is Person In Trouble:"+monkey_trouble(false,true))  
7) console.log("Is Person In Trouble:"+monkey_trouble(false,false))
```



## Output:

```
Is Person In Trouble:true  
demo.js:5 Is Person In Trouble:false  
demo.js:6 Is Person In Trouble:false  
demo.js:7 Is Person In Trouble:true
```

## Problem-3: Warmup-2 > string\_times

Given a string and a non-negative int n, return a larger string that is n copies of the original string.

```
string_times('Hi', 2) --> 'HiHi'  
string_times('Hi', 3) --> 'HiHiHi'  
string_times('Hi', 1) --> 'Hi'
```

## Solution:

```
1) function string_times(str,n){  
2)   result="";  
3)   var count=1;  
4)   while(count<=n){  
5)     result=result+str;  
6)     count++;  
7)   }  
8)   return result;  
9) }  
10) console.log(string_times("durga",3))  
11) console.log(string_times("hello",2))
```

## Output:

```
durgadurgadurga  
hellohello
```

## Problem-4: Logic-2 > lucky\_sum

Given 3 int values, a b c, return their sum. However, if one of the values is 13 then it does not count towards the sum and values to its right do not count. So for example, if b is 13, then both b and c do not count.

```
lucky_sum(1, 2, 3) --> 6  
lucky_sum(1, 2, 13) --> 3  
lucky_sum(1, 13, 3) --> 1
```



## Solution:

```
1) function lucky_sum(a,b,c){
2)   if(a==13){
3)     return 0;
4)   }
5)   if(b==13){
6)     return a;
7)   }
8)   if(c==13){
9)     return a+b;
10)  }
11) }
12) console.log(lucky_sum(13,10,5))//0
13) console.log(lucky_sum(5,13,6))//5
14) console.log(lucky_sum(7,5,13))//12
```

## Problem-5:Logic-1 > caught\_speeding

You are driving a little too fast, and a police officer stops you. Write code to compute the result, encoded as an int value: 0=no ticket, 1=small ticket, 2=big ticket. If speed is 60 or less, the result is 0. If speed is between 61 and 80 inclusive, the result is 1. If speed is 81 or more, the result is 2. Unless it is your birthday -- on that day, your speed can be 5 higher in all cases.

caught\_speeding(60, False) --> 0

caught\_speeding(65, False) --> 1

caught\_speeding(65, True) --> 0

## Solution:

```
1) function caught_speeding(speed,isBirthday){
2)   if (isBirthday) {
3)     speed=speed-5;
4)   }
5)   if (speed<=60) {
6)     return 0;
7)   }
8)   else if (speed>=61 && speed<=80) {
9)     return 1;
10)  }
11) else{
12)   return 2;
13) }
```



```
14) }  
15) console.log("Getting Ticket With Number:"+caught_speeding(60, false))//0  
16) console.log("Getting Ticket With Number:"+caught_speeding(65, false))//1  
17) console.log("Getting Ticket With Number:"+caught_speeding(65, true))//0
```

## JavaScript Arrays:

- An array is an indexed collection of elements.
- The main advantage of arrays concept is we can represent multiple values by using a single variable so that length of the code will be reduced and readability will be improved.

### Without Arrays:

```
var n1=10;  
var n2=20;  
var n3=30;  
var n4=40;
```

### With arrays:

```
var numbers=[10,20,30,40]
```

## Accessing Array Elements by using Index:

By using index we can access array elements. JavaScript arrays follow 0-based index. i.e The index of first element is 0

### Eg:

```
var friends=["durga","sunny","bunny","chinny"];  
console.log(friends[0]); //durga  
console.log(friends[3]); //chinny  
console.log(friends[30]); //undefined
```

**Note:** If we are trying to access array elements by using out of range index then we will get undefined value and we won't get any error.

## Updating Array Elements by using Index:

```
var friends=["durga","sunny","bunny","chinny"];  
friends[1]="mallika"  
console.log(friends)// ["durga", "mallika", "bunny", "chinny"]
```

## Adding New Elements to the Array by using Index:

```
var friends=["durga","sunny","bunny","chinny"];  
friends[4]="vinny";  
console.log(friends)//["durga","sunny","bunny","chinny","vinny"]
```



```
friends[40]="pinny";  
console.log(friends)// ["durga", "sunny", "bunny", "chinny", "vinny", empty × 35, "pinny"]
```

**Note:** By using index we can retrieve, update and add elements of array. But in general we can use index to access array elements.

## How to Create an Empty Array:

1<sup>st</sup> Way: `var numbers=[];`

2<sup>nd</sup> Way: `var numbers=new Array();`

## How to find Length of Array:

By using length variable

```
var friends=["durga","sunny","bunny","chinny"];  
console.log(friends.length)//4
```

## Is JavaScript Array can hold only Homogeneous Elements?

Javascript array can hold heterogeneous elements also.

**Eg:** `var random_collection=["durga",10000,true,null]`

## Important Methods related to JavaScript Arrays:

JavaScript defines several methods which are applicable on arrays.

Being a programmer we can use these methods directly and we are not responsible to implement so that our life will become very easy.

The following are important methods

- 1) `push()`
- 2) `pop()`
- 3) `unshift()`
- 4) `shift()`
- 5) `indexOf()`
- 6) `slice()`

### 1) `push()`:

We can use `push()` method to add elements to the end of array. After adding element this method returns length of the array.

**Eg:**

```
var numbers=[10,20,30,40]  
numbers.push(50)  
console.log(numbers)// [10, 20, 30, 40, 50]
```



## 2) pop():

We can use pop() method to remove and return last element of the array

```
var numbers=[10,20,30,40]
console.log(numbers.pop())// 40
console.log(numbers.pop())// 30
console.log(numbers)// [10,20]
```

## 3) unshift():

We can use unshift() method to add element in the first position. It is counter part of push() method.

Eg:

```
var numbers=[10,20,30,40]
numbers.unshift(50)
console.log(numbers)//[50, 10, 20, 30, 40]
```

## 4) shift():

We can use shift() method to remove and return first element of the array. It is counter part to pop() method.

Eg:

```
var numbers=[10,20,30,40]
numbers.shift()
console.log(numbers)//[20, 30, 40]
```

## 5) indexOf():

- We can use indexOf() to find index of specified element.
- If the element present multiple times then this method returns index of first occurrence.
- If the specified element is not available then we will get -1.

Eg:

```
var numbers=[10,20,10,30,40];
console.log(numbers.indexOf(10))//0
console.log(numbers.indexOf(50))// -1
```

## 6) slice():

- We can use slice operator to get part of the array as slice.
- slice(begin,end) → returns the array of elements from begin index to end-1 index.
- slice() → returns total array.This can be used for cloning purposes.



**Eg:**

```
var numbers=[10,20,30,40,50,60,70,80]
var num1=numbers.slice(1,5)
console.log(num1)// [20, 30, 40, 50]
num2=numbers.slice()
console.log(num2)// [10, 20, 30, 40, 50, 60, 70, 80]
```

## **Multi Dimensional Arrays:**

Sometimes array can contain arrays as elements.i.e array inside array is possible. Such type of arrays are considered as multi dimensional arrays or nested arrays.

**Eg:**

```
var nums=[[10,20,30],[40,50,60],[70,80,90]]
console.log(nums[0]//[10,20,30]
console.log(nums[0][0]//10
```

## **Book Management Application:**

**demo.js:**

```
1) var books=[]
2) var input=prompt("Which operation You want to perform [add|list|exit]:")
3) while (input != "exit") {
4)   if (input=="add") {
5)     var newBook= prompt("Enter Name of the Book:")
6)     books.push(newBook);
7)   }
8)   else if (input=="list") {
9)     console.log("List Of Available Books:");
10)    console.log(books);
11)  }
12)  else {
13)    console.log("Enter valid option");
14)  }
15)  input=prompt("What operation You want to perform [add|list|exit]:")
16) }
17) console.log("Thanks for using our application");
```



## Retrieving Elements of Array:

We can retrieve elements of array by using the following ways

- 1) while loop
- 2) for loop
- 3) for-of loop
- 4) forEach method

### 1) while loop:

```
1) var nums=[10,20,30,40,50]
2) var i=0;
3) while (i<nums.length) {
4)   console.log(nums[i]);
5)   i++;
6) }
```

### 2) for loop:

```
1) var nums=[10,20,30,40,50]
2) for (var i = 0; i < nums.length; i++) {
3)   console.log(nums[i]);
4)   //alert(nums[i]);
5) }
```

### 3) for-of loop:

It is the convenient loop to retrieve elements of array.

```
1) var colors=["red","blue","yellow"]
2) for (color of colors) {
3)   console.log('*****');
4)   console.log(color);
5)   console.log('*****');
6) }
```

### 4) forEach Method:

- forEach() is specially designed method to retrieve elements of Array.
- **Syntax:** arrayobject.forEach(function)
- For every element present inside array the specified function will be applied.

```
1) var heroines=['sunny','mallika','samantha','katrina','kareena']
2) function printElement(element){
3)   console.log('*****');
4)   console.log(element);
```





```
5) console.log('*****');  
6) }  
7) heroines.forEach(printElement)
```

### Eg 2:

```
1) var heroines=['sunny','mallika','samantha','katrina','kareena']  
2) heroines.forEach(function (element) {  
3)   console.log('*****');  
4)   console.log(element);  
5)   console.log('*****');  
6) })
```

**Note:** The following are also valid

```
heroines=['sunny','mallika','samantha','katrina','kareena']  
heroines.forEach(console.log)  
heroines.forEach(alert)
```

## for Loop vs forEach Function:

- 1) for loop is general purpose loop and applicable everywhere. But forEach() function is applicable only for arrays.
- 2) By using for loop, we can move either forward direction or backward direction. But by using forEach() function we can move only forward direction.

**Eg:** By using for loop we can print array elements either in original order or in reverse order. But by using forEach() function we can print only in original order.

## How to Delete Array Elements based on Index:

We have to use splice() function.

**Syntax:** arrayobject.splice(index,numberofElements)

It deletes specified number of elements starts from the specified index.

### Eg:

```
var heroines=['sunny','mallika','samantha','katrina','kareena']  
heroines.splice(3,1)  
console.log(heroines);//[ "sunny", "mallika", "samantha", "kareena"]
```



## Immutability vs Mutability:

Once we create an array object, we are allowed to change its content. Hence arrays are Mutable.

Eg:

```
var numbers=[10,20,30,40]
numbers[0]=777
console.log(numbers)//[777,20,30,40]
```

Once we create a string object, we are not allowed to change the content. If we are trying to change with those changes a new object will be created and we cannot change content of existing object. Hence string objects are immutable.

Eg:

```
var name='Sunny'
name[0]='B'
console.log(name)// Sunny
```

**Note:** Mutability means changeable whereas Immutable means Non-Changeable.

## Q1) Write a JavaScript Function to take an Array as Argument and Print its Elements in Reverse Order?

```
1) function reverse(array){
2)   console.log('Elements in Reverse Order:')
3)   for (var i = array.length-1; i >=0 ; i--) {
4)     console.log(array[i])
5)   }
6) }
7) reverse([10,20,30,40,50])
8) reverse(['A','B','C','D','E'])
```

### Output:

Elements in Reverse Order:

50  
40  
30  
20  
10

Elements in Reverse Order:

E  
D  
C



B  
A

**Q2) Write a JavaScript Function to check whether the Elements of given Array are Identical (same) OR not?**

```
1) function identical(array){
2)   var first=array[0]
3)   for (var i = 1; i < array.length; i++) {
4)     if (array[i] != first) {
5)       return false;
6)     }
7)   }
8)   return true;
9) }
10) console.log(identical([10,10,10,10]));//true
11) console.log(identical([10,20,30,40]));//false
```

**Q3) Write a JavaScript Function to find Maximum Value of the given Array?**

```
1) function max(array){
2)   var max=array[0]
3)   for (var i = 1; i < array.length; i++) {
4)     if (array[i] > max) {
5)       max=array[i]
6)     }
7)   }
8)   return max
9) }
10) console.log(max([10,20,30,40]));//40
```

**Q4) Write a JavaScript Function to find the Sum of Elements present in given Array?**

```
1) function sum(array){
2)   var sum=0
3)   for (num of array) {
4)     sum+=num
5)   }
6)   return sum
7) }
8) console.log(sum([10,20,30,40]));//100
```



## Book Management Application:

```
1) var books=[]
2) var input=prompt("Which operation You want to perform [add|delete|list|exit]:")

3) while (input != "exit") {
4)   if (input=="add") {
5)     addBook();
6)   }
7)   else if (input=="list") {
8)     listBooks()
9)   }
10)  else if(input=="delete"){
11)    deleteBook()
12)  }
13)  else {
14)    console.log("Enter valid option");
15)  }
16)  input=prompt("What operation You want to perform [add|delete|list|exit]:")
17) }
18) console.log("Thanks for using our application");
19)
20) function addBook(){
21)   var newBook= prompt("Enter Name of the Book:")
22)   books.push(newBook);
23) }
24) function listBooks(){
25)   console.log("List Of Available Books:");
26)   for (book of books) {
27)     console.log(book);
28)   }
29) }
30) function deleteBook(){
31)   var name=prompt("Enter Book Name to delete:")
32)   var index=books.indexOf(name)
33)   if(index== -1){
34)     console.log("Specified book is not available");
35)   }
36)   else{
37)     books.splice(index,1)
38)     console.log("Specified Book Deleted");
39)   }
40) }
```



## JavaScript Objects:

- By using arrays we can store a group of individual objects and it is not possible to store key-value pairs.
- If we want to represent a group of key-value pairs then we should go for Objects.
- Array: A group of individual objects
- Object: A group of key-value pairs
- JavaScript objects store information in the form of key-value pairs.
- These are similar to Java Map objects and Python Dictionary objects.

**Syntax:** var variableName={ key1:value1,key2:value2,...};

```
1) var movie={  
2)     name:'Bahubali',  
3)     year: 2016,  
4)     hero:'prabhas'  
5) };
```

In the case of JavaScript objects, no guarantee for the order and hence index concept is not applicable.

## How to access Values from Object:

We can access values by using keys in the following 2 ways

### 1) obj["key"]

Here quotes are mandatory

Eg: movie["hero"] → Valid

movie[hero] → Uncaught ReferenceError: hero is not defined

### 2) obj.key

Here we cannot take quotes

Eg: movie.hero

## How to create and initialize JavaScript Objects:

To create empty object var nums={} OR var nums=new Object()

Once we create empty object we can add key-value pairs as follows

### 1<sup>st</sup> Way:

nums["fno"]=100

nums["sno"]=200

### 2<sup>nd</sup> Way:

nums.fno=100

nums.sno=200



## How to Update Values:

nums["fno"]=999 or  
nums.fno=999

## Iterating Objects:

To access all key-value pairs we can use for-in loop.

```
1) var nums={fno=100,sno=200,tno=300}
2) for(key in nums){
3)     console.log(key); //To print only keys
4)     console.log(nums[key]); //To print only values
5)     console.log(key+":"+nums[key]); //To print both key and values
6) }
```

## Differences between Arrays and Objects

Arrays	Object
1) Arrays can be used to represent individual values	1) Objects can be used to represent key-value pairs
2) Order will be maintained in Arrays	2) Order concept not applicable for Objects
3) By using index we can access and update data in arrays	3) By using key we can access and update data in Objects

## Nested Objects and Arrays:

Inside Array, we can take objects. Similarly inside Objects we can take array.  
Hence nesting of objects and arrays is possible.

### Eg 1:

```
1) var movies=[{name:'Bahubali',year:2016,hero:'Prabhas'},
2)             {name:'Sanju',year:2018,hero:'Ranveer'},
3)             {name:'Spider',year:2017,hero:'Mahesh'}
4) ]
```

movies[0]["hero"] → Prabhas  
movies[2]["year"] → 2017

### Eg 2:

```
1) var numbers={
2)     fg:[10,20,30],
3)     sg:[40,50,60],
4)     tg:[70,80,90]}
```



```
| 5)    }
```

numbers.sg[2] → 60

numbers.tg[1] → 80

## Object Methods:

JavaScript Object can contain Methods also.

```
1) var myobj={  
2)   A:'Apple',  
3)   B:'Banana',  
4)   m1:function(){console.log("Object Method");}  
5) }
```

We can invoke this method as follows: myobj.m1()

## this Keyword:

Inside object methods, if we want to access object properties then we should use 'this' keyword.

```
1) var movie={  
2)   name:'Bahubali',  
3)   year: 2016,  
4)   hero:'prabhas',  
5)   getInfo:function(){  
6)     console.log('Movie Name:'+this.name);  
7)     console.log('Released Year:'+this.year);  
8)     console.log('Hero Name:'+this.hero);  
9)   }  
10)  };  
11)  
12) movie.getInfo()
```

## Output:

Movie Name:Bahubali

Released Year:2016

Hero Name:prabhas

It is possible to refer already existing function as object method.



## Eg 1:

```
1) function demo(){
2)   console.log('Demo Function');
3) }
4)
5) var movie={
6)   name:'Bahubali',
7)   year:2016,
8)   hero:'Prabhas',
9)   getInfo:demo
10)  };
11) movie.getInfo()
```

## Output:

Demo Function

## Eg 2:

```
1) function demo(){
2)   console.log('Demo Function:'+this.name);
3) }
4)
5) var movie={
6)   name:'Bahubali',
7)   year:2016,
8)   hero:'Prabhas',
9)   getInfo:demo
10)  };
11) movie.getInfo()
```

## Output: Demo Function:Bahubali

If we call demo() function directly then output is: Demo Function

We can use named functions also.

```
1) var movie={
2)   name:'Bahubali',
3)   year:2016,
4)   hero:'Prabhas',
5)   getInfo: function demo(){
6)     console.log('Demo Function:'+this.name);
7)   }
8) };
```





Even we are not required to use function keyword also for object methods inside object and we can declare function directly without key.[But outside of object compulsory we should use function keyword to define functions]

```
1) var movie =  
2) {  
3)   name:"Rockstar",  
4)   hero:"Ranbeer Kapoor",  
5)   year:"2012",  
6)   myFunction(){  
7)     console.log("kjdf");  
8)   }  
9) }
```

Even we can pass parameters also

```
1) var movie =  
2) {  
3)   name:"Rockstar",  
4)   hero:"Ranbeer Kapoor",  
5)   year:"2012",  
6)   myFunction(a){  
7)     console.log("kjdf:"+a);  
8)   }  
9) }  
10) var a =10;  
11) movie.myFunction(a)
```

## Mini Application:

```
1) var movies=[{name:'Bahubali',isWatched:'true',isHit:'true'},  
2)             {name:'Sanju',isWatched:'false',isHit:'true'},  
3)             {name:'Spider',isWatched:'true',isHit:'false'},  
4)             ]  
5) movies.forEach(function(movie){  
6)   var result=""  
7)   if(movie.isWatched=="true"){  
8)     result=result+"I Watched "  
9)   }  
10)  else{  
11)    result=result+"I have not seen "  
12)  }  
13)  result=result+movie.name  
14)  if(movie.isHit=="true"){  
15)    result=result+" and Movie is Hit!!!"
```



```
16) }  
17) else{  
18) result=result+" and Movie is Flop!!!"  
19) }  
20) console.log(result)  
21) };
```

### Output:

I Watched Bahubali and Movie is Hit!!!

I have not seen Sanju and Movie is Hit!!!

I Watched Spider and Movie is Flop!!!



# DOM

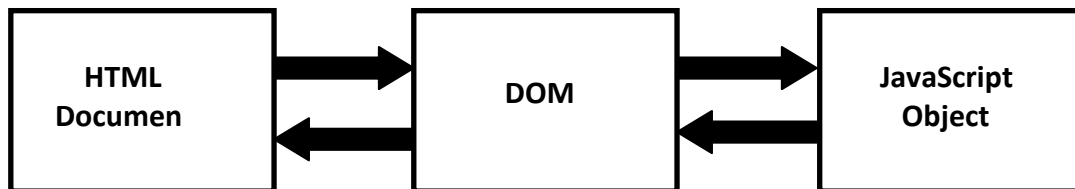
## Study Material



# Document Object Model (DOM)

DOM acts as interface between JavaScript and HTML, CSS.

i.e By using DOM, JavaScript can communicate/connect with HTML and CSS.



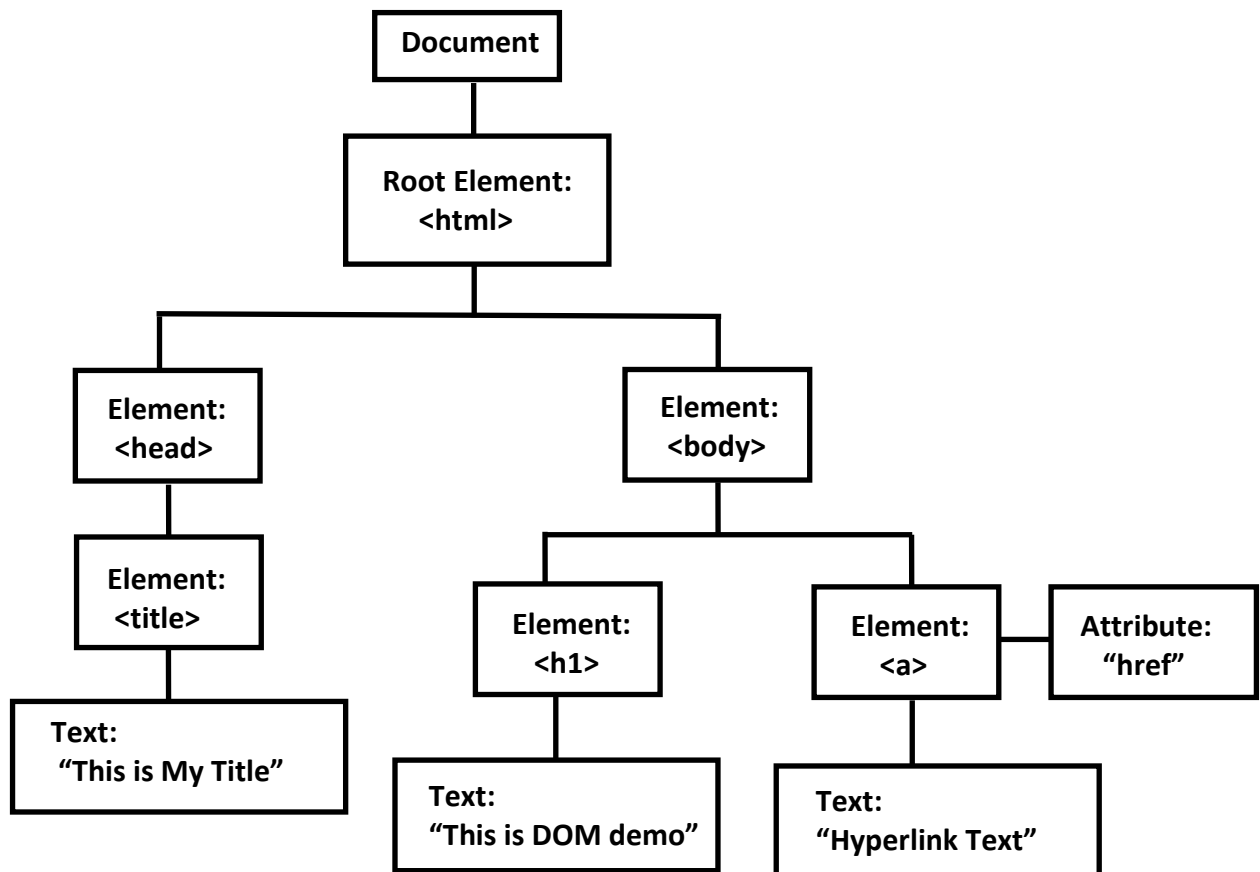
Browser will construct DOM. All HTML tags will be stored as JavaScript objects.

demo.html:

```
1) <!DOCTYPE html>
2) <html>
3) <head>
4) <title>This is My Titile</title>
5) </head>
6) <body>
7) <h1>This is DOM Demo</h1>
8) <a href="#">HeperlinkText</a>
9) </body>
10) </html>
```



The corresponding DOM Structure is:



### To display Document to the console:

Just type on javascript console: document

Then we will get total HTML Page/Document.

### To Display DOM Objects on the Console:

console.dir(document)

Observe that Root element of DOM is document.

### How to grab HTML Elements from the DOM:

### Important DOM Attributes:

- 1) document.URL → This is original URL of the website
- 2) document.body → It returns everything inside body
- 3) document.head → It returns head of the page
- 4) document.links → It returns list of all links of the page



## Important Methods of DOM:

DOM defines the following methods to grab HTML elements

- 1) **document.getElementById()**  
Returns element with the specified id
- 2) **document.getElementsByClassName()**  
Returns list of all elements belongs to the specified class
- 3) **document.getElementsByTagName()**  
Returns list of all elements with the specified tag
- 4) **document.querySelector()**  
Returns the first object matching CSS style selector
- 5) **document.querySelectorAll()**  
Returns all objects Matches the CSS Style Selector

## Demo Application:

### demo.html:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <meta charset="utf-8">
5) <title></title>
6) </head>
7) <body>
8) <h1>DOM important Methods and Attributes</h1>
9) <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tem
por incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis no
strud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis a
ute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla
pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia d
eserunt mollit anim id est laborum.</p>
10) <h2>Favourite Food:</h2>
11) <ul>
12) <li class="first">Chicken</li>
13) <li>Mutton</li>
14) <li>Fish</li>
15) <li id="special">Any animal including Human being</li>
16) </ul>
```



```
17) <h2>Favourite Drink:</h2>
18) <ul>
19) <li class="first">KingFisher</li>
20) <li>KnockOut</li>
21) <li>Milk</li>
22) <li>Thumsup</li>
23) </ul>
24) <a href="http://youtube.com/durgasoftware">View Profile</a>
25) </body>
26) <script type="text/javascript" src="demo.js"></script>
27) </html>
```

## On the JavaScript Developer's Console:

- 1) document.URL
- 2) document.body
- 3) document.header
- 4) document.links
  
- 5) document.getElementById('special')
- 6) document.getElementsByClassName('first')
- 7) document.getElementsByTagName('h2')
- 8) document.querySelector('#special')
- 9) document.querySelectorAll('.first')

## querySelector() vs querySelectorAll():

- If the specified CSS mapped with only one html element then we should use querySelector() method. If multiple html elements matched, then querySelector() method returns only first matched element.
- We can use querySelectorAll() method to grab all matched html elements. If multiple html elements are there then we can use this method.

## Q) Write DOM based JavaScript Code to change Color of h1 Tag

```
var myh1 = document.querySelector('h1');
myh1.style.color = 'red';
```

## Demo Application to grab HTML Elements into JavaScript:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <meta charset="utf-8">
5) <title></title>
6) </head>
```



```
7) <body>
8) <h1>This is h1 Tag Data</h1>
9) <p id='first' class="special">This is First Paragraph Data</p>
10) <p class="special">This is Second Paragraph Data</p>
11) <p>This is Third Paragraph Data</p>
12) <p id='last'>This is Fourth Paragraph Data</p>
13) </body>
14) </html>
```

The following are various possible ways to grab first paragraph

- 1) document.getElementById('first')
- 2) document.getElementsByClassName('special')
- 3) document.getElementsByClassName('special')[0]
- 4) document.getElementsByTagName('p')
- 5) document.getElementsByTagName('p')[0]
- 6) document.querySelector('#first')
- 7) document.querySelectorAll('.special')[0]
- 8) document.querySelector('h1+p')

## Color Changer Application:

To generate Random number from 0 to 15 Java Code is:

```
1) class Test
2) {
3)     public static void main(String[] args)
4)     {
5)         System.out.println((int)(Math.random()*16));
6)         System.out.println((int)(Math.random()*16));
7)         System.out.println((int)(Math.random()*16));
8)         System.out.println((int)(Math.random()*16));
9)         System.out.println((int)(Math.random()*16));
10)        System.out.println((int)(Math.random()*16));
11)        System.out.println((int)(Math.random()*16));
12)        System.out.println((int)(Math.random()*16));
13)    }
14) }
```

### demo.html:

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4)     <meta charset="utf-8">
5)     <title></title>
```





```
6) </head>
7) <body>
8) <h1>DOM important Methods and Attributes</h1>
9) <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tem
por incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis no
strud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis a
ute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla
pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia d
eserunt mollit anim id est laborum.</p>
10) <h2>Favourite Food:</h2>
11) <ul>
12) <li class="first">Chicken</li>
13) <li>Mutton</li>
14) <li>Fish</li>
15) <li id="special">Any animal including Human being</li>
16) </ul>
17)
18) <h2>Favourite Drink:</h2>
19) <ul>
20) <li class="first">KingFisher</li>
21) <li>KnockOut</li>
22) <li>Milk</li>
23) <li>Thumsup</li>
24) </ul>
25)
26) <a href="http://youtube.com/durgasoftware">View Profile</a>
27) </body>
28) <script type="text/javascript" src="demo.js"></script>
29) </html>
```

## demo.js

```
1) var h1=document.querySelector('h1')
2) var p=document.querySelector('p')
3) var h2=document.querySelector('h2')
4) var ul=document.querySelector('ul')
5)
6) function getRandomColor(){
7)   var letters='0123456789ABCDEF';
8)   var color='#';
9)   for (var i = 0; i < 6; i++) {
10)    var r=Math.floor(Math.random()*16);
11)    color=color+letters[r]
12)  }
13)  return color
```



```
14) }  
15) function changeColor(){  
16)   var rc=getRandomColor()  
17)   h1.style.color=rc;  
18)   p.style.color=getRandomColor();  
19)   h2.style.color=getRandomColor();  
20)   ul.style.color=getRandomColor();  
21) }  
22) setInterval(changeColor,1000)
```

## DOM: Content Interaction

By using DOM, we can change Text, HTML Code and attributes.

### How to Change Text

```
var h1 = document.querySelector('h1')  
h1.textContent = "This is Modified Content"
```

### How to Change HTML Code

If we use textContent property then HTML changes won't be effected. Hence we have to use innerHTML property

```
var p = document.querySelector('p')  
  
p.textContent='<strong><em>'+p.textContent+'</em></strong>'//invalid  
p.innerHTML='<strong><em>'+p.textContent+'</em></strong>' //valid
```

### How to Change Attributes

We can use the following methods

#### element.getAttribute('attributeName')

Returns the value associated with specified attribute

#### element.setAttribute('attributeName','attributeValue')

To set new value to the attribute

Eg:

```
var a = document.querySelector('a')  
// console.log(a.getAttribute('href'))  
a.setAttribute('href','https://facebook.com')
```



## Demo Application

### demo.html

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <meta charset="utf-8">
5) <title></title>
6) </head>
7) <body>
8) <h1>This h1 text can be replaced</h1>
9) <p>This data can be converted into bold</p>
10) <a href="http://youtube.com/durgasoftware">View Profile</a>
11) </body>
12) <script type="text/javascript" src="demo.js"></script>
13) </html>
```

### demo.js

```
1) var h1=document.querySelector('h1')
2) h1.textContent="This is Modified Content"
3) var p= document.querySelector('p')
4) p.innerHTML='<strong><em>'+p.textContent+'</em></strong>'
5) var a=document.querySelector('a')
6) // console.log(a.getAttribute('href'))
7) a.setAttribute('href','https://facebook.com')
```

## Changing All Links Properties of google.com:

```
1) var links=document.querySelectorAll('a');
2) for(link of links){
3) link.style.background='green';
4) link.style.color='white';
5) link.setAttribute('href','http://durgasoftonline.com');
6) link.textContent='Sunny Leone';
7) }
```

## To replace google logo without customized logo:

```
1) var logo=document.querySelector('#hplogo');
2) logo.setAttribute('src','http://www.durgasoftonline.com/wp-content/uploads/logo_durgasoft_online-1.png')
```



## Event Handling by using DOM:

- In our web application there may be a chance of occurring several events like mouse hover, single click and double click etc
- Whenever a particular event occurs if we want to perform certain operation automatically then we should go for Listeners.
- Listener listens events and will perform certain operation automatically.

## How to implement Event Handling:

`element.addEventListener(event,function);`

Eg: `myh1.addEventListener('click',colorChanger)`

## To change color of h1 on single click operation:

```
var myh1=document.querySelector('h1')  
myh1.addEventListener('click',changeColor)
```

## To change color of h1 on double click operation:

```
var myh1=document.querySelector('h1')  
myh1.addEventListener('dblclick',changeColor)
```

## To change color of h1 on mouse over operation:

```
var myh1=document.querySelector('h1')  
myh1.addEventListener('mouseover',changeColor)
```

## To change color of h1 on mouse out operation:

```
var myh1=document.querySelector('h1')  
myh1.addEventListener('mouseout',changeColor)
```

## To change content and color as blue on mouse over operation:

```
1) var myh1=document.querySelector('h1')  
2) myh1.addEventListener('mouseover',function(){  
3)   myh1.textContent='HYDERABAD';  
4)   myh1.style.color='blue';  
5) });  
6)  
7) myh1.addEventListener('mouseout',function(){  
8)   myh1.textContent='BANGALORE';  
9)   myh1.style.color='red';  
10) });
```



## Body Color Changer Application

### demo.html

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3)   <head>
4)     <meta charset="utf-8">
5)     <title></title>
6)   </head>
7)   <body>
8)     <h1>Body Color Changer Application</h1>
9)     <button type="button" name="button">Click Here to change Body Color</button>
10)  </body>
11) <script type="text/javascript" src="demo.js"></script>
12) </html>
```

### demo.dss

```
1) function getRandomColor(){
2)   var letters='0123456789ABCDEF';
3)   var color='#';
4)   for (var i = 0; i < 6; i++) {
5)     var r=Math.floor(Math.random()*16);
6)     color=color+letters[r]
7)   }
8)   return color
9) }
10) var b=document.querySelector('button');
11) b.addEventListener('click',function(){
12)   document.body.style.background=getRandomColor();
13) });
```

## Demo Application

### demo.html

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3)   <head>
4)     <meta charset="utf-8">
5)     <title></title>
6)   </head>
7)   <body>
```



```
8) <h1>Event Hanling by using DOM</h1>
9) </body>
10) <script type="text/javascript" src="demo.js"></script>
11) </html>
```

## demo.js

```
1) function getRandomColor(){
2)   var letters='0123456789ABCDEF';
3)   var color='#';
4)   for (var i = 0; i < 6; i++) {
5)     var r=Math.floor(Math.random()*16);
6)     color=color+letters[r]
7)   }
8)   return color
9) }
10) function changeColor(){
11)   var rc=getRandomColor()
12)   h1.style.color=rc;
13)   p.style.color=getRandomColor();
14)   h2.style.color=getRandomColor();
15)   ul.style.color=getRandomColor();
16) }
17) var h1=document.querySelector('h1')
18) h1.addEventListener('mouseover',changeColor)
19) h1.addEventListener('mouseout',changeColor)
```

## Demo Application for Random Names and Random Colors

### demo.html

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3)   <head>
4)     <meta charset="utf-8">
5)     <title></title>
6)   </head>
7)   <body>
8)     <h1>Event Hanling by using DOM</h1>
9)   </body>
10)  <script type="text/javascript" src="demo.js"></script>
11) </html>
```



## demo.js

```
1) var myh1=document.querySelector('h1')
2) function getRandomColor(){
3)   var letters='0123456789ABCDEF';
4)   var color='#';
5)   for (var i = 0; i < 6; i++) {
6)     var r=Math.floor(Math.random()*16);
7)     color=color+letters[r]
8)   }
9)   return color
10) }
11)
12) function getRandomName(){
13)   var names=['SUNNY LEONE','MALLIKA','VENNA','KATRINA','PRIYANKA','DEEPIKA','KAREENA'];
14)   var r=Math.floor(Math.random()*7);
15)   return names[r];
16) }
17)
18) myh1.addEventListener('mouseover',function(){
19)   myh1.textContent=getRandomName();
20)   myh1.style.color=getRandomColor();
21) });
22) myh1.addEventListener('mouseout',function(){
23)   myh1.textContent=getRandomName();
24)   myh1.style.color=getRandomColor();
25) });
```

**Q) Names should be displayed with the specified color only like Katrina always with red color etc**

## TIC TAC TOE Implementation:

- 1) Create HTML, CSS, JavaScript Files
- 2) Add CSS, Bootstrap links to HTML inside Head Part
- 3) Add JavaScript File to HTML at Bottom of Body Tag
- 4) Add Jumbotron to the HTML

```
1) <div class="container">
2)   <div class="jumbotron">
3)     <h1>Welcome to DURGASOFT TIC TAC TOE Game!!!</h1>
4)     <p>Be Ready To play to improve logical thinking...</p>
5)     <button id="b" type="button" class="btn btn-primary btn-lg"
       name="button">Restart Game!!!</button>
```



```
6) </div>
7) </div>
```

## 5. Define Jumbotron related styles in CSS File

```
1) .container .jumbotron{
2) background: green;
3) color:white;
```

## 6. Add table after jumbotron inside container

```
1) <table>
2) <tr>
3) <td></td>
4) <td></td>
5) <td></td>
6) </tr>
7) <tr>
8) <td></td>
9) <td></td>
10) <td></td>
11) </tr>
12) <tr>
13) <td></td>
14) <td></td>
15) <td></td>
16) </tr>
17) </table>
```

## 7. Define Table related styles in css file

```
1) td{
2) height: 150px;
3) width: 150px;
4) border: 5px solid blue;
5) text-align: center;
6) font-size: 100px;
7) }
```

## 8. Implementing Restart button functionality inside js file

```
1) var restartb=document.querySelector('#b')
2) var cells=document.querySelectorAll('td')
3) function clearAllCells(){
4) for (var i = 0; i < cells.length; i++) {
```





```
5) cells[i].textContent=""
6) }
7) }
8) restartb.addEventListener('click',clearAllCells)
```

## 9. Add content change functionality of the cells inside javascript file

```
1) function changeContent(){
2)   if (this.textContent=== "") {
3)     this.textContent='X'
4)   }
5)   else if (this.textContent=='X') {
6)     this.textContent='O'
7)   }
8)   else {
9)     this.textContent=""
10)  }
11) }
12) for (var i = 0; i < cells.length; i++) {
13)   cells[i].addEventListener('click',changeContent)
14) }
```

## ttt.html

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3)   <head>
4)     <link rel="stylesheet" href="ttt.css">
5)     <!-- Latest compiled and minified CSS -->
6)     <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity = "sha384-BVYiISiFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">
7)     <meta charset="utf-8">
8)     <title>Tic Tac Toe Game</title>
9)   </head>
10)  <body>
11)    <div class="container">
12)      <div class="jumbotron">
13)        <h1>Welcome to DURGASOFT TIC TAC TOE Game!!!</h1>
14)        <p>Be Ready To play to improve logical thinking...</p>
15)        <button id="b" type="button" class="btn btn-primary btn-lg" name="button">Restart Game!!!</button>
16)      </div>
17)    <table>
```



```
18) <tr>
19) <td></td>
20) <td></td>
21) <td></td>
22) </tr>
23) <tr>
24) <td></td>
25) <td></td>
26) <td></td>
27) </tr>
28) <tr>
29) <td></td>
30) <td></td>
31) <td></td>
32) </tr>
33) </table>
34)
35) </div>
36) <script src="ttd.js"></script>
37) </body>
38) </html>
```

## ttd.css

```
1) .container .jumbotron{
2) background: green;
3) color:white;
4) }
5) td{
6) height: 150px;
7) width: 150px;
8) border: 5px solid blue;
9) text-align: center;
10) font-size: 100px;
11) }
```

## ttd.js

```
1) var restartb=document.querySelector('#b')
2) var cells=document.querySelectorAll('td')
3) function clearAllCells(){
4) for (var i = 0; i < cells.length; i++) {
5) cells[i].textContent=""
6) }
7) }
```



```
8) restartb.addEventListener('click',clearAllCells)
9) function changeContent(){
10) if (this.textContent==="") {
11)   this.textContent='X'
12) }
13) else if (this.textContent=='X') {
14)   this.textContent='O'
15) }
16) else {
17)   this.textContent=""
18) }
19) }
20) for (var i = 0; i < cells.length; i++) {
21)   cells[i].addEventListener('click',changeContent)
22) }
```



***jQuery***

# Study Material



# jQuery

Official website: [jquery.com](http://jquery.com)

jQuery is a fast, small, and rich JavaScript library.

We can use jQuery Library to grab and manipulate HTML elements, to perform event handling and Ajax.

jQuery supports multiple browsers, i.e. it provides support for cross-browser compatibility.

The main advantage of jQuery is it provides several methods and objects in the form of JavaScript file, so that we can use these directly and the developer's life gets simplified.

**Note:** In Plain old JavaScript (also known as Vanilla JavaScript), we have to write everything manually. But if we use jQuery, we are not required to write much code and we can use its library directly.

## Sample Code to Change Color of every h1 Tag:

### Vanilla JS code:

```
1) var myh1=document.querySelectorAll('h1')
2) for( h1 of myh1){
3) h1.style.color='red';
4) }
```

### jQuery Code:

```
$('h1').css('color','red')
```

## Advantages of jQuery:

- 1) It provides several built-in methods and objects. We can use these directly so that development will become very easy.
- 2) Clear and Shorter code
- 3) Ease of use
- 4) Cross-Browser support
- 5) AJAX support



## Limitations of jQuery:

- 1) What ever jQuery doing, we can implement everything without jQuery also. ie jQuery won't do any things extra.
- 2) The total library should be loaded compulsory, which creates performance problems.

**Note:** Because of above limitations, some part of the developer's community won't recommend jQuery usage.

**Eg:** youmightnotneedjquery.com

## How to connect with jQuery:

We can make jQuery library available to our application in the following 2 ways

- 1) By Locally
- 2) By CDN

### 1) By Locally:

- Download jQuery.js file from jquery.com → <https://code.jquery.com/jquery-3.3.1.js>
- Download and place in application folder.
- Inside head of html we have to write <script> tag as follows  
<script type="text/javascript" src="jquery.js"></script>

### 2) By using CDN:

[www.code.jquery.com](http://www.code.jquery.com)

```
1) <script
2)   src="http://code.jquery.com/jquery-3.3.1.js"
3)   integrity="sha256-2Kok7MbOyxpqUVvAk/HJ2jigOSYS2auK4Pfzbm7uH60="
4)   crossorigin="anonymous"></script>
```

We can get several jquery CDNs from the google

**Eg:** <script type="text/javascript"  
src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

## How to Check whether jQuery available or not in our Application:

From the console just type

> jQuery or \$

We should not get any error

## jQuery Selectors:

In Vanilla Javascript we have several methods to select/grab html elements like

getElementById()

getElementsByClassName()



getElementsByTagName()  
querySelector()  
querySelectorAll()  
etc

But in jQuery we have only one way to select html elements.i.e to use \$ symbol

In jQuery \$ symbol is equivalent to querySelectorAll() in Vanilla Javascript

## Demo Application

### demo.html

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <script
5) src="http://code.jquery.com/jquery-3.3.1.js"
6) integrity="sha256-2Kok7MbOyxpgUVvAk/HJ2jigOSYS2auK4Pfzbm7uH60="
7) crossorigin="anonymous"></script>
8) <meta charset="utf-8">
9) <title></title>
10) </head>
11) <body>
12) <h1>Hello First H1</h1>
13) <h2>Favourite Drink:</h2>
14) <ul>
15) <li>KingFisher</li>
16) <li>KnockOut</li>
17) <li>Foster</li>
18) <li id='special'>HumanBlood</li>
19) </ul>
20) <a href="https://google.com">Click Here to go to Google</a>
21) </body>
22) </html>
```

### jQuery code:

\$('#h1')  
\$('#li')  
\$('#special')  
\$('#body')  
etc...



## Manipulating HTML Elements:

Once we grab elements by using \$ symbol, we can manipulate by using css() method.

**Syntax:** \$(selector).css(property,value)

### **Examples:**

```
$('#h1').css('color','white');  
$('#h1').css('background','red');  
$('#h1').css('border','5px solid green');
```

We can save selected html element by using variable

```
var x = $('#h1')  
x.css('color','white');  
x.css('background','red');  
x.css('border','5px solid green');
```

Instead of passing parameters one by one, we can create Object and pass that object directly.

```
1) var x = $('#h1')  
2) var myCSS={  
3)   color:'white',  
4)   background:'green',  
5)   border:'red 5px solid'  
6) }  
7) x.css(myCSS);
```

**Note:** We can use \$ Symbol to select and css() Method to manipulate HTML Elements.

## How to Select a particular HTML Element instead of all matched Elements:

By default \$('element') selects all matched html elements.

But by using the following ways we can get particular matched element.

```
$('#element:first')  
$('#element:last')  
$('#element:first-of-type')  
$('#element:nth-of-type(n)')  
etc...
```





**Eg:**

- 1) \$('h1') → Selects all h1 tags
- 2) \$('h1:first') → Selects only first h1 tag
- 3) \$('h1').first() → Selects only first h1 tag
- 4) \$('h1:first-of-type') → Selects only first h1 tag
- 5) \$('h1:nth-of-type(2)') → Selects second h1 tag
- 6) \$('h1:last') → Selects only last h1 tag
- 7) \$('h1').last() → Selects only last h1 tag

**Q1) Write Vanilla JavaScript and jQuery Codes to Change all h1 Tags Text Color as White and Background as Red**

**Vanilla JavaScript Code:**

```
1) var allh1s=document.querySelectorAll('h1');
2) for(h1 of allh1s){
3)   h1.style.color='white';
4)   h1.style.background='red';
5) }
```

**jQuery Code:**

```
1) var mystyles={
2)   color:'white',
3)   background:'red'
4) };
5) $('h1').css(mystyles)
6)
7) Instead of this we can write directly as follows
8)
9) $('h1').css({
10)   color:'white',
11)   background:'red'
12) });
```

**Q2) Write Vanilla JavaScript and jQuery Codes to set all li Tags Font Size as 20px**

**Vanilla JavaScript Code:**

```
1) var x=document.querySelectorAll('li');
2) for(li of x){
3)   li.style.fontSize='20px';
4) }
```



## jQuery Code:

```
$('li').css('fontSize','20px')
```

**Note:** The biggest advantage of jQuery is we can do more things with less Code (Write less, do More)

## Demo Application

### demo.html

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3)   <head>
4)     <script
5)       src="http://code.jquery.com/jquery-2.2.4.js"
6)       integrity="sha256-iT6Q9iMjYuQiMWNd9IDyBUStIq/8PuOW33aOqmvFpql="
7)       crossorigin="anonymous"></script>
8)     <meta charset="utf-8">
9)     <title></title>
10)  </head>
11)  <body>
12)    <p>This is First Paragraph</p>
13)    <p id="second">This is Second Paragraph</p>
14)    <p class="remaining">This is Third Paragraph</p>
15)    <p class="remaining">This is Fourth Paragraph</p>
16)  </body>
17) </html>
```

**Case-1:** Select all <p> tags and set background as green

```
$('p').css('background','green')
```

**Case-2:** Select all <p> tags with class 'remaining' and make them 200px wide(width)

```
$('.remaining').css('width','200px')
```

**Case-3:** Select all <p> tags with id 'second' and give red solid 10px border.

```
$('#second').css('border','10px solid red')
```

**case-4:** Select only first <p> tag and change text color as white

```
$('p:first').css('color','white')
```

**case-4:** Select only third <p> tag and change font-size as 30px

```
$('p:nth-of-type(3)').css('fontSize','30px')
```



## The Most commonly used jQuery Methods:

The following are the most commonly used jQuery methods

- 1) text()
- 2) html()
- 3) attr()
- 4) val()
- 5) addClass()
- 6) removeClass()
- 7) toggleClass()

### 1) text():

- We can use this method to get and set text of the matched elements. i.e this method acts as both getter and setter method.
- text() → To get text of all matched elements including child tags. In this case this method acts as getter method
- text(content) → To set provided text content for every matched element. In this case this method acts as setter method.

**Note:** If any method developed to acts as both getter and setter method then it is said to be this method follows getter and setter paradigm.

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <script
5) src="http://code.jquery.com/jquery-2.2.4.js"
6) integrity="sha256-iT6Q9iMjYuQiMWNd9IDyBUSTlq/8PuOW33aOqmvFpql="
7) crossorigin="anonymous"></script>
8) <meta charset="utf-8">
9) <title></title>
10) </head>
11) <body>
12) <h1>This is H1 Data</h1>
13) <h2>Choose Your Favourite Subject:</h2>
14) <ul>
15) <li>HTML</li>
16) <li>JavaScript</li>
17) <li>CSS</li>
18) <li>jQuery</li>
19) </ul>
20) </body>
21) </html>
```



1) To get Text Content of h1:

`$('#h1').text()`

2) To get Text of all li Tags

`$('#li').text()` OR `$('#ul').text()`

3) To get Text of only First li

`$('#li').first().text()`

4) To Set h1 Text Content as : DURGA SOFTWARE SOLUTIONS

`$('#h1').text('DURGA SOFTWARE SOLUTIONS')`

5) To Set Every li Tag Text Content as : KINGFISHER

`$('#li').text('KINGFISHER')`

2) html():

- We can use this method to get and set html content.
- If we are not passing any argument then this method acts as getter method, which returns HTML contents of the first matched element.
- Eg: `$('#h1').html()`
- If we are passing argument then this method acts as setter method, which sets the HTML contents of every matched element.
- Eg: `$('#li').html('<a href="https://amazon.com">AMAZON</a>')`

3) attr():

- We can use this method to get and set attribute values.
- `attr(attributename)` → To get the value of specified attribute of the first matched element.
- `attr(attributename, attributevalue)` → If the specified attribute already there then old value replaced with new value. If the specified attribute not already available then a new attribute will be added for every matched tag.

## Demo for getting and setting src Attribute of img Tag:

``

``

``



## 1) To Set Width and Height of every Image Properly

```
$('#img').css({width:'150px',height:'150px'});
```

## 2) To get src Attribute Value of the First Image

```
$('#img').attr('src')
```

## 3) To Set src Attribute of all Images with Our New Image

```
$('#img').attr('src',"https://images.unsplash.com/photo-1484406566174-9da000fda645?ixlib=rb-0.3.5&ixid=eyJhcnBfaWQiOjEyMDd9 &s=a439fe04a06e1457297643e219add900&auto=format&fit=crop&w=400&q=60");
```

## 4) To Change only First Image src Attribute Value

```
$('#img').first().attr('src',"https://images.unsplash.com/photo-1504350440606-81847d413a13?ixlib=rb-0.3.5&ixid=eyJhcnBfaWQiOjEyMDd9&s=bdf251e852bd65abcc522fc4e903da02&auto=format&fit=crop&w=400&q=60")
```

## Demo for getting and setting Type Attribute of Input Tag:

User Input: `<input type="text" name="" value="">`

### 1) To get the Value of Type Attribute of Input Tag

```
$('#input').attr('type')
```

### 2) To Set the Value of Type Attribute with 'color'

```
$('#input').attr('type','color')
```

### 3) To Set the Value of Type Attribute with 'checkbox'

```
$('#input').attr('type','checkbox')
```

### 4) val() Method:

We can use val() method to get value of the first matched element.

Eg: value entered in the text box

which radio button selected

which value selected from dropdown box

We can also use val() method to set value for every matched element.

val() → Getter Method

val(text) → Setter Method

### Eg 1: with input tag

Enter Name:<input type="text" name="user1" value="" placeholder="Your Name">

Enter Name:<input type="text" name="user2" value="" placeholder="Your Name">

\$('#input').val() → It returns the value entered by the end user in the first text box.

\$('#input').val('durga') → To set value durga for every text box.



**Usecase:** While implementing reset button functionality to clear all input fields this method is helpful.

**Eg 1:** with dropdown menu

Choose Your Required Course:

```
1) <select class="" name="">
2) <option value="CorePython">Core Python</option>
3) <option value="AdvPython">Adv Python</option>
4) <option value="DJango">DJango</option>
5) <option value="Flask">Flask</option>
6) </select>
```

`$('select').val()` → Returns Selected Value

`$('select').val('DJango')` → To set value as DJango

5) **addClass()**: We can use this method to add specified class/classes to the set of matched elements.

6) **removeClass()**: Remove a single class/classes from each matched element.

7) **toggleClass()**: We can use this method to add and remove classes from the matched elements.

If the specified class already set then it will be removed. If it is not already set then the class will be added for every matched element.

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3) <head>
4) <script
5) src="http://code.jquery.com/jquery-2.2.4.js"
6) integrity="sha256-iT6Q9iMjYuQiMWNd9IDyBUSTlq/8PuOW33aOqmvFpqI="
7) crossorigin="anonymous"></script>
8) <meta charset="utf-8">
9) <title></title>
10) <style >
11) .high{
12)   color:white;
13)   background:red;
14) }
15) .low{
16)   color:white;
```



```
17) background: blue;
18) }
19) .completed{
20) text-decoration: line-through;
21) }
22) </style>
23) </head>
24) <body>
25) <h1>List of Activities: </h1>
26) <ul>
27) <li>Meeting Sunny</li>
28) <li>Reading Django</li>
29) <li>Taking Classes</li>
30) <li>Visiting US</li>
31) <li>Watching Movie</li>
32) <li>Playing Cricket</li>
33) <li>Swimming </li>
34) </ul>
35) </body>
36) </html>
```

1) `$('li').addClass('high')`

It will add 'high' class to every li tag

2) `$('li:nth-of-type(even)').removeClass('high')`

It will remove 'high' class from every even numbered li tag.

3) `$('li:nth-of-type(2)').addClass('completed')`

It will add 'completed' class only for 2nd li

4) `$('li').toggleClass('low')`

It will add 'low' class for every li tag(because this class is not already set)

5) `$('li').toggleClass('low')`

It will remove 'low' class from every li tag(because this class is already set)

## Event Handling by using jQuery:

- We can implement event handling to make our html elements interactive.
- jQuery defines several methods for event handling. We can get complete jquery event handling related methods from below link: <https://api.jquery.com/category/events/>



## The Top 3 Most Commonly used jQuery Event Methods:

- 1) click()
- 2) keypress()
- 3) on()

### 1) click():

jQuery click() method can be used to add a click listener to the elements.

**Eg 1:** To raise alert message when ever we are clicking h1 tag.

```
1) $('h1').click(function(){  
2)   alert('h1 tag got clicked');  
3) });
```

**Eg 2:** To raise alert message and to change background color of last button on click event

```
1) $('button:last').click(function(){  
2)   alert('Hello Stupid Dont Sleep I will Kill You!!!');  
3)   $(this).css('background','red')  
4) });
```

**Note:** In Vanilla Java Script, 'this' always represent current element. But in jQuery we have to use \$(this)

## Demo Application:

```
1) <!DOCTYPE html>  
2) <html lang="en" dir="ltr">  
3)   <head>  
4)     <script  
5)       src="http://code.jquery.com/jquery-2.2.4.js"  
6)       integrity="sha256-iT6Q9iMJYuQiMWNd9IDyBUStIq/8PuOW33aOqmvFpql="  
7)       crossorigin="anonymous"></script>  
8)     <meta charset="utf-8">  
9)     <title></title>  
10)   </head>  
11)   <body>  
12)     <h1>jQuery Event Handler Demo </h1>  
13)     <button type="button" name="button">Dont Sleep First Warning</button>  
14)     <button type="button" name="button">Dont Sleep Second Warning</button>  
15)     <button type="button" name="button">Dont Sleep Third Warning</button>  
16)     <script type="text/javascript" src="demo.js">  
17)   </script>  
18)   </body>
```





```
19) </html>
```

## demo.js:

```
1) $('h1').click(function(){
2)   alert('h1 tag clicked')
3) })
4) $('button:first').click(function(){
5)   alert('Hello Dont Sleep');
6)   $(this).css('background','yellow')
7) });
8) $('button:nth-of-type(2)').click(function(){
9)   alert('Dont Sleep I will beat you');
10)  $(this).css('background','orange')
11) });
12) $('button:last').click(function(){
13)  alert('Hello Stupid Dont Sleep I will Kill You!!!');
14)  $(this).css('background','red')
15) });
```

## 2) keypress():

- We can use this method to add keypress listener to elements.
- i.e whenever we are pressing the key if we want to do any activity automatically then we should go for keypress() method.
- Eg: To raise alert message for every character typed in text box  
Enter Name:

## jQuery Code:

```
1) $('input').keypress(function(){
2)   alert('Inserted one character!!!!');
3) })
```

## To Know the pressed Character:

- In the case of keypress event, the total information is available in the event object. For every character the corresponding key-code is available.
- char-code, unicode,ascii code, which all represents same key-code
- The complete key-code information is available in the following link  
<https://www.cambiaresearch.com/articles/15/javascript-char-codes-key-codes>
- If we use event.which, then we will get the corresponding key-code



## To raise Alert Message whenever we are pressing x OR X:

```
1) $('input').keypress(function(event){  
2)   if (event.which==88 || event.which==120)  
3)   {  
4)     alert('Hello You are pressing x or X. You are under monitoring !!!!')  
5)   }  
6) }}
```

**Note:** The key-code for 'x' is: 120 where as for 'X' is: 88

## To raise Alert Message with typed Content whenever we are pressing Enter Key:

```
1) $('input').keypress(function(event){  
2)   if (event.which==13)  
3)   {  
4)     alert('Hello just pressed enter key and your current content is:'+$({this}).val())  
5)   }  
6) }}
```

## Difference between keypress and keydown, keyup Events:

keydown and keyup provides a code indicating which key is pressed, where as keypress indicates that which character was entered.

**Eg:** For shift+a

- 1) keydown for 'shift'
- 2) keydown for 'a'
- 3) keyup for 'a'
- 4) keyup for 'shift'
- 5) key press for 'A'

### 3) on():

- on() is the most commonly used method to perform event handling in jQuery.
- It is similar to Vanilla Java Script addEventListener() method.
  
- click() → applicable only for click event
- keypress() → applicable only for keypress event
- on() → applicable for all events including click and keypress.



**Eg 1:** Whenever mouseover event happend, the text of h1 tag should be changed to 'Bangalore City' with red background and white text.

Whenever mouseout event happend, the text of h1 tag should be changed to 'Hyderabad City' with green background and white text.

```
1) $('h1').on('mouseover',function(){
2)   $(this).text('BANGALORE CITY')
3)   $(this).css({background:'red',color:'white'});
4) })
5) $('h1').on('mouseout',function(){
6)   $(this).text('HYDERABAD CITY')
7)   $(this).css({background:'green',color:'white'});
8) })
```

**Eg 2:** For button

**Single Click:** alert message as Hello Stupid dont click

**Double Click:** alert message as Hello Animal I will kill you

## html

```
1) <button type="button" name="button">Click Here to get Greeting Message</button><br><br>
2) <button type="button" name="button">Double Click Here to get Greeting Message</button>
```

## jQuery

```
1) $('button:first').on('click',function() {
2)   alert('Hello Stupid Dont Click!!!')
3) })
4)
5) $('button:last').on('dblclick',function() {
6)   alert('Hello Animal I Will Kill You!!!')
7) })
```

## jQuery Effects:

jQuery provides several in-built effects. But main important effects are:

- 1) Fading Effects
- 2) Sliding Effects

<https://api.jquery.com/category/effects/>



## I) Fading Effects:

jQuery defines the following methods for fading purposes

### 1) fadeOut():

Hide the matched elements by fading them to transparent.

### 2) fadeIn():

Display the matched elements which are fadeout

### 3) fadeToggle():

Display or hide the matched elements

If already fadeOut then fadeIn will be performed.

If already fadeIn then fadeOut will be performed.

## Various Sample Codes:

```
1) $('div').fadeOut();
2) $('div').fadeOut(2000);
3) $('div').fadeOut(2000,function() {
4)     console.log('Fadeout of the element completed');
5) });
```

2000 is the time in milliseconds

Similarly we can use these combinations for fadeIn() and fadeToggle() also.

## Demo Application

### demo.html

```
1) <!DOCTYPE html>
2) <html lang="en" dir="ltr">
3)   <head>
4)     <style>
5)       div{
6)         height: 100px;
7)         width: 100px;
8)         background: red;
9)         color:white;
10)        margin:20px;
11)        text-align: center;
12)        float: left;
13)      }
14)    </style>
```



```
15) <script src="http://code.jquery.com/jquery-2.2.4.js"
16) integrity="sha256-iT6Q9iMJYuQiMWNd9IDyBUSTlq/8PuOW33aOqmvFpql="
17) crossorigin="anonymous"></script>
18) <meta charset="utf-8">
19) <title></title>
20) </head>
21) <body>
22) <button type="button" name="button">Click Here</button>
23) <div>First Div</div>
24) <div>Second Div</div>
25) <div>Third Div</div>
26) <div>Fourth Div</div>
27) </body>
28) <script type="text/javascript" src="demo.js">
29) </script>
30) </html>
```

## demo.js

```
1) $('button').on('click',function(){
2)   $('div').fadeToggle(2000);
3) })
```

**Note:** Whenever we perform fadeout just elements will be hidden but won't be removed. But based on requirement we can remove the matched elements also.

```
1) $('button').on('click',function(){
2)   $('div').fadeToggle(2000,function(){
3)     $(this).remove();
4)   });
5) })
```

## II) Sliding Effects:

jQuery defines the following methods for sliding effects purposes.

### 1) slideUp():

Hide the matched elements with a sliding motion.

### 2) slideDown():

Display the matched elements with a sliding motion.

### 3) slideToggle():

Display or hide the matched elements with a sliding motion.



**Eg:**

```
$('#button').on('click',function(){  
    $('#div').slideToggle(2000);  
})
```

**Note:** passing function as argument, remove matched elements are exactly same as fading effects.

## **Assignments:**

- 1) Connect4 Game
- 2) Todo List Application
- 3) Implement atleast 2 case studies from w3schools how to  
<https://www.w3schools.com/howto/>