What is javascript?
javascript is a client-side scripting language that is executed by a web browser while loading an webpage into the browser. to build dynamic web pages in HTML we generally use Javascript as part of web pages.
using java script we can change
1. contents of an HTML element
2. change styles of an HTML element
3. we show and can hide HTML elements within an web page.

from the above we can understand using javascript we can change the page elements of a HTML element which makes the elements of the page and webpage dynamic.

How to write javascript in a webpage?
In general there are many browser-based | client-side scripting languages are available.
1. javascript
2. vbscript
3. typescript
which are executed by the web browser. now such scripting language instructions has to be written in special tag to distinghuish HTML from script instructions using <script> tag.

We can write javascript code with in HTML page either in head or body sections of the HTML. while writing javascript code we need surround the code inside the <script type="text/javascript"></script>.

```
<html>
  <head>
    <script type="text/javascript">
      // write javascript code here
    </script>
  </head>
</html>
```

In the above code type="" attribute in script is not mandatory, because javascript has been announced as a the default scripting language by the web browsers.

We can write javascript at 2 places either within <head> section or within <body> section. it is always recommended to write javascript in body section of the page at the end of the body.
If we write javascript at the end of the body we have several advantages:-
1. the page loading will be very fast, the entire HTML

elements will be rendered quickly with no wait of executing the javascript.
2. if we write javascript at the <head> level before the browser constructs the page elements of your webpage javascript might begin execution referring the page elements which are not yet constructor these results in script errors. to avoid write javascript at the end of the body tag, so that by that time the web browser loads and renders all the page elements so that you javascript can refer all of the elements of the page without errors.

```
<html>
  <head>
  </head>
  <body>
    <!-- html elements -->

    <script>
      // script logic
    </script>
  </body>
</html>
```

javascript can be written in 2 ways.
1. embedded javascript = we directly write the javascript instructions within the web page itself under script tag directly.

2. external javascript = the javascript code is written external in a file with extension ".js" and will be imported whereever we want to use it.

page.js

```
alert("page loaded");
```

```
<html>
  <head>
  </head>
  <body>
    <script type="text/javascript" src="page.js"></script>
  </body>
</html>
```

if we want to reuse the javascript across multiple pages of your application, it is recommended to go for external javascript only.
you can write any number of <script> tags within a page or any no of external js imports in a page.

What are the display possibilities of javascript?
There are 4 ways in which a javascript program can write the output of its execution to a browser/user.

1. few are used for writing output directly on to the browser window
2. few are used for collecting user inputs from the browser like a diaglog or alert box
3. few are used for debugging the javascript code execution

There are predefined objects in javascript language representing a web browser few of them are
#document = browser object, representing the renderable area of your web browser, where content is render.
#window = represents the entire browser window, using this object we can interact with web browser like

1. close()
2. maximize
3. minimize
4. change address bar location

#console = is another predefined object which can be used for writing log messages of the javascript execution. that helps javascript developers to debug the code easily

1.document.write() =
is used for directly writing the output to an web page. it writes the output content based on the current page flow of your web page.

2.element.innerHTML =
document.write() writes the content directly on to the webpage, instead we want to write some content into an exiting element of the page, we can use innerHTML on that element as shown below.

```
<script>
  divelement = document.getElementById("topsection");
  divelement.innerHTML = "Hurray! written by javascript";
</script>

<div id="topsection">
  Hurray! written by javascript
</div>
```

innerHTML is an property that is by default available for all the container elements like

1. div
2. p
3. span

4. anchor
5. header
6. select
7. table
8. ul .... etc

3.window.alert()
alert() is function in window object which can be used for drawing the user attention towards the application in displaying critical information. alerts are display as dialogues unless user interacts the alerts will not dis-appear and page will not continue execution

4.console.log()
for writing log messages into the web browser for debugging javascript.

Javascript statements and variables

1. javascript statements
Every javascript statement should end with (;) at the end of the statement. You can write javascript statements with any amount of whitespaces and new lines as well, still it executes.

---

2. javascript variables
How to declare variables in javascript.
There are 2 ways we can declare variable in javascript
1. var keyword
2. let (not supported by many browsers)

```
var i = 10;
var s = "Good Morning";
```

The variables can be combined and declared in one single line itself.

```
var i = 10, greet = "Good Morning";
```

---

the variable names that we are declaring should follow naming conventions

1. a variable name can container letters, digits, underscore and dollar.
2. it should start with a letter only or can have an _ or a $ instead of a letter
3. usually follows camel case only while declaring variables.

```
    4. keywords of javascript cannot be used as variable
    names

we can use an variable without declaration as well,
variable declaration is not mandatory in javascript.
a = 10;
b = 20;
-------------------------------------------------------
-----------------------------------------------
Javascripts supports the following data types
1. Number
2. String
while declaring a variable we dont need to decare type of
the variable, javascript supports dynamic typing which
indicates the datatype of the variable will be decided
based on the value we assigned to the variable

var a = 10; // number type
var person = "alex"; // string type variable
a string can assigned in 2 ways either using singlequote
or double quote allowed
var person = 'alex';

divcontents.innerHTML="<input type='' name='' />"; - here
we can write another string inside a string using
singlequote, we can escape the strings

-------------------------------------------------------
-----------------------------------------------
A variable in javascript will be initialized only once.
a = 10;
var a = 20; // we cannot re-declare a variable,as a is
already defined the var statement will be ignored
a = 30;

document.write("a : " +a);
-------------------------------------------------------
-----------------------------------------------
when we declare a variable without an value, by default
the variable is initialized to undefined in javascript.
undefined is a special type that is used for indicating
the type is not know.

var a; // undefined type
undefined - is a special type in javascript to let us
understand the variable is not assigned with value to
indicates its data type

-------------------------------------------------------
-----------------------------------------------
```

```
when we declare a variable with var keyword, that variable acts as a global variable within the document, so that we can access that variable in any <script></script> sections of the document
<head>
  <script>
    var a = 10;
  </script>
</head>
<body>
  <script>
    document.write("a : " +a);
  </script>
</body>
-------------------------------------------------------------------------------------------
in javascript we can assign a variable to a null value, null is a special type of object indicating the variable doesnt hold any value.
  var a; // undefined
  var b = null; // is assgined with null value, null is an object type in javascript
  var s = ""; // emtpy value

a == b = returns true
a === b = to distingush between null and undefined we need to use ====

-------------------------------------------------------------------------------------------

We can find the datatype of a variable in javascript using typeof(variable) in javascript
  a = 10;
  typeof(a) = NUMBER
  s = 'good morning'
  typeof(s) = STRING
  var x;
  typeof(x) = UNDEFINED
  var y = null;
  typeof(y) = OBJECT
-------------------------------------------------------------------------------------------
let is a keyword that can be used for declaring a variable that scopes to a block level, think of its like a local variable, so that the value of the variable will be defined to the scope of the block in which it is assigned.

<script>
```

```
    var a = 10;
    {
        let a = 20; // we assigned value 20 to the
        variable a within the block scope, once the block
        of execution completed the a variable regains its
        original value
        document.write("in block a : "+ a); // 20
    }
    document.write("out block a : " + a); // 10 here back
</script>
```

Javascript functions

functions are the named block code that can be executed by passing paramters and returns the return value up on executing the block of code written inside it.

functions are the means of achieving reusablity in a program. if have some repeatedly lines of code that has to be executed for several times at different places in your program, instead of duplicating the code we use functions in javascript.

syntax:-

```
<script>
function funcName() {

}

(or)
var func = function() {
}
</script>
```

a function can even take parameters as part of it just declare the names of the parameters into which you want to recieve the values, we dont need to declare the variable by prefixing var keyword.

```
<script>
  function funcName(a, b) {

  }
</script>
```

and even a function can return a return value using the return statement.

```
<script>
  function funcName(a, b) {
    // logic
```

```
    return returnValue;
  }
</script>
```

functions can be written in any place within the html webpage and to have them more reusable it is often recommended to declare them in external js file and import and use it in all the web page of your application.

functions by themselve will not be executed unless those are being called by others. There are many ways a function can be called or executed.

1. the calling program or a main script can call the javascript function
2. function can get triggered based on an event, so here function acts as a event handler
   for eg.. click on button, can call a function or closing a window, can execute a function etc
3. function calling by itself for eg. timer function to set where the function will be called repeatedly.

scope of variables in function

---

variables declared inside function using var keyword will be local to function and are called function variables

```
<script>
  function add() {
    var sum = 0;
  }
  add();
  alert("sum : " + sum); // produces error sum is not defined
</script>
```

if you declare a variable inside function without var keyword, then the variable becomes global variable

```
<script>
  function add() {
    sum = 0;
  }
  add();
  alert("sum : " + sum); //will works because sum is global variable
</script>
```

we can define functions and can assign to variables, those variable are function pointers can call the functions using the variableName(); function calling operator.

```
<script>
  var sum = function() {
    return 10;
  }
  sum();
</script>
```

---

javascript objects

---

in javascript there is no concept of class declaration and creating objects of the class. Directly we define object with attributes and functions as part of it.

An object contains variable declarations and functions binded to the object using which we perform some operation.

syntax:-

```
var person = {
  firstName: "joe",
  lastName: "smith",
  age: 23,
  gender: "male"
};
```

we defined an object called person above. now we can access the attributes of the object using object.propertyName

```
person.firstName
person.lastName
```

an object can not only have variable declarations it have even functions as well.

```
var person = {
  firstName: "joe",
  lastName: "smith",

  fullname = function() {
    return this.firstName + "-" + this.lastName;
  };
};

alert(person.fullname());
```

we can change the value of a property in an object using assignment operator as object.propertyName=value

```
person.firstName = "rock";
alert(person.fullname());
```

```
312 js strings
313
314 set of characters combined together is called a string.
    in javascript we can create strings in multiple ways.
315 1. var s = "Good Morning";
316 2. var s = 'Good Morning!';
317 3. var s = new String("Good Morning"); // here the String
    is an object type in javascript
318
319 in the first 2 cases the string is a string type variable
    but in the last case the String type is returned as
    object. It is always recommended to create a string using
    double/single quote, dont use String constructor to
    create a string because we cannot differentiate the
    Object Type and String Type.
320
321 var s1 = "javascript!";
322 var s2 = new String("javascript!");
323
324 if we apply a == operator on s1 and s2, they return false
    even though the contents of both the strings are same.
325 s1 == s2 = false = because s1 is string type and s2 is
    object type, as their fundamental types of different they
    return false.
326
327 A string object has few attributes and methods available.
328 var s = "string attributes";
329 s.length = length is an attribute in String class that
    gives the length of the given string.
330
331 ------------------------------------------------------
    ------------------------------------------------
332 Methods of String
333 ------------------
334 var message = "Good Morning have a nice day!";
335 message.indexof("Morning"); = return the position of the
    string where it was found in the given string
336
337 var message = "Good Morning! hope all things goes good";
338 message.lastIndexOf("Good") = will returns the last
    occurence position of good in given string
339
340 both these methods returns -1 if the string was not found.
341
342 another form of indexOf(String, startPosition) and
    lastIndexOf(String, position); tells from the given
    position in the string the string has to be searched.
343
344 search = used for searching a substring in a given string
    and once found it returns index position of the string
```

that was found.
345 In case of search, there is no second parameter like position within the string where it has begin search and search allows the search string as regular expression where as indexOf and lastIndexOf doesnt allows regular expression based searching.
346
347 var message = "Good that we are all together";
348 message.search("are"); = returns the index position of "are" where it was found in the given string
349
350 working with substrings there are 2 methods are available.
351 1. slice(start, end)
352 2. substring(start, end)
353
354 slice() method is used for extracting a substring from a given string based on start and end position specified. this method can take negative numbers as well, given a negative number it calculates the position from at of the string in reverse order.
355
356 substring() method is simlar to slice() method only but the only difference is substring doesnt allow negative numbers.
357 The substring has another form also substring(beginIndex) which returns the substring from the begin index till the last.
358
359 var line = "Once upon a time, there lived a shepherd boy";
360 var sline = line.substring(10); // once upon (result)
361
362 replace() method is used for replacing a given string with new string. the replace method will not modify the original string rather it creates a new tring by replacing.
363 var line = "Once upon a time, there lived a shepherd boy";
364 var rline = line.replace("boy", "girl");
365 replace method only replaces the first match of the string, but not all.
366
367 toUpperCase() and toLowerCase() = to convert a given string into upper case or lower case letters we use these methods.
368 var line = "The sheep are being chased by the wolf!";
369 var upperLine = line.toUpperCase();
370
371 concat() method to concat any given 2 strings.
372 var s1 = "Good";
373 var s2 = "Afternoon";

```
374
375 var greetings = s1.concat(" ", s2); // result in Good
    Afternoon
376 all the string functions returns a new string after
    performing the operation they will not modify the
    existing string
377
378 trim() = is used for removing leading and tailing space
    of a given string
379 var s = "remove space after me        ";
380 var s1 = s.trim();
381
382
383 charAt(position) = is used for extracting a character in
    the given string under specified location
384   var s = "find charachter in string";
385   var ch = s.charAt(2);// i
386 charCodeAt(position) = returns utf8/16 charset encoding
    value it returns
387
388 the other way we can access the characters in a string is
    using property operator
389   s[2] = will returns i
390 ------------------------------------------------------
    ------------------------------------------------
    javascript numbers
391
392 var a = 10; // a number
393 var b = 10.2; // decimal number
394 a number in javascript doesnt have different types like
    int, float, double  everything is a number of size 64-bit
    length
395
396 all the arthematic and relational operators can be
    applied on numbers.
397   var a = 10;
398   var b = 20;
399   var sum = a + b; // arthematic operation
400
401 as there are no datatypes in javascript when apply +
    operator between strings it acts as concatnation
    operator. but when applied on numbers arthematic operator.
402   if we use both in combination then expression would be
      evaluated from left to the right and based that outcome
      will be derived.
403   For e.g.
404   var a = "Good Morning";
405   var b = 10;
406   var c = 20;
407
```

```
  var s = a + b + c; // the result would be Good Morning1020
  var s1 = b + c + a; //30Good Morning



var a = "bye";
var b = 20;
applying any other arthematic operators apart from + results in NaN = not a number
  var c = a / b; // c is NaN

var a = "Good Day";
var n = isNaN(a); = true if the given string is container an integer value otherwise returns false
var b = "10";
isNaN(b) = false

NaN = is a predefined Type which is of Number
Infinity

Number even has functions
  - parseInt("10"); // converts a string number into number
  - parseFloat("10.2"); // converts floating value in number
  - var a = 93.983;
    a.toFixed(2); // returns 93.98 fixing decimal positions to 2
  - valueOf(), toString()

There are constants in Number class
  - Number.MAX_VALUE
  - NUMBER.MIN_VALUE

important:- NaN, isNaN(), parseInt() and parseFloat()
-------------------------------------------------------------------------------
js date
Date is an object type in javascript and we can use constructor of Date object to creates in javascript.

var d = new Date(); // this creates a date with current date and time
var d = new Date(2021, 11, 20); year, month , date
var d = new Date(2021); only year
var d = new Date(2011, 01); //only year and month
var d = new Date(2011, 01, 20, 11, 54); // year, month, day, hour, minutes
var d = new Date(2011, 01, 20, 11, 54, 23); // year, month, day, hour, minutes and seconds
```

```
448
449   d.toString() == returns string representation of data
      object
450   d.getMonth() == returns month of given date
451   d.getFullYear()
452   d.getDay()
453   ------------------------------------------------------------
      ----------------------------------------------
454   js arrays
455
456   What are arrays what is the purpose of arrays?
457   Arrays are used for storing collection or group of values.
458
459   var marks = [10, 8, 7, 6, 5];  // this is an array
460   var fruits = ["apple", "banana", "orange"];
461
462   we can create array using Array Object.
463   var fruits = new Array("apple", "banana", "orange"); //
      instead of using Object notation recommended to use []
      square bracket notation to create an array.
464
465   how to access elements from array, using index position
466     fruits[0] = returns apple
467   how to modify the element of an array
468     fruits[0] = "pineapple";
469
470   In javascript arrays are Object type, if we use
      typeof(array) it returns object type only. In this case
      how to find a given object is an array type or not in
      javascript.
471
472   Array.isArray() = we can find whether the given object is
      array or not
473
474   var fruits = ["apple", "banana"];
475
476   fruits.constructor.toString(); // Array = its an array
477   var arrayType = fruits.constructor.toString();
478   var pos = arrayType.indexOf("Array");
479     if(pos != -1) {
480       // its an array
481     }
482
483   var a = new Number(10); // object
484   a.constructor.toString(); // Number
485
486   var d = new Date(); // object
487
488   we can find the length of an array using fruits.length
      attribute
```

```
489   push() = a method to add an element into the array
490
491   var fruits = ["apple", "banana"];
492   fruits.push("grapes"); // adds the grapes to the end of
      array
493
494   instead of using push function we can use index also
495   var fruits = ["apple", "banana"];
496   fruits[2] = "grapes"; // this will extend the array
      automatically
497   fruits[10] = "papaya"; //this leaves empty 8 locations
      between 2 and 10
498
499   Array functions
500     sort = to sort the elements of an array alphabetic or
        numbers based on its contents
501     var a = [8, 92, 2, 89];
502     a.sort(); // sort the elements
503
504     reverse = reverses the contents of an array
505     var products = ["fridge", "tv"];
506     var rproducts = products.reverse();
507
508     shift = removes the first element from the array
509     var b = [10, 239, 29, 40];
510     b.shift(); // removes 10
511
512     unshift = adds an new element at the begining of the
        array
513     var b = [298, 03, 02];
514     b.unshift(1); = adds 1 to the begining of the array
515
516     pop() = for removing an element from the last
517     delete fruits[10]; // deletes the element mark it as
        undefined
518
519     splice() = to add new elements in to array we use splice
520     var products = ["tv", "fridge", "mixer"];
521     array.splice(startPosition, remove, "elements to be
        added");
522
523     products.splice(1, 1, "air conditioner", "washing
        machine"); // tv, air conditioner, washing machine
524     products.splice(1, 1); == removes 1 element from 1
        position
525
526     concat() = concats given 2 arrays
527     var i = [10, 20];
528     var j = [29, 39];
529     var bigN = i.concat(j); // concats both the arrays
```

```
array attributes:-
  length

array functions
  - sort
  - push
  - pop
  - shift
  - unshift
  - splice
  - concat
  - reverse
  - join*
-------------------------------------------------------------
-----------------------------------------------
Array Iterations [streaming api in java]
-----------------------------------------
```

As array in javascript is object type we can use various different functions to iterate and process the elements of the array. there are plenty of functions to stream the elements and processes them in javascript.

#1 foreach

```
<script>
  var marks = [80, 76, 89, 45, 30];

  for(var i=0;i<marks.length;i++) { // legacy
    // logic
  }

  marks.forEach(print);
  function print(value) {
    document.write("value : " + value);
  }
  marks.forEach(function(n) {
    document.write("n : " + n);
  });

  marks.forEach(function(n, index, array) { // classic for loop because we have index, value and array
    document.write("n : "+ n + " indexOf(n) : "+ index + " from array : "+ array);
  });
</script>
```

#2 map

map is used for mapping each element of the array into different value

```
var marks = [80, 75, 90, 91, 86];

var marks25 = marks.map(function(n, idx, array) {
  return n/4;
});

#3 filter()
based on a matching condition we want return sub group of values from the original array.
var names = ["paul", "james", "jack", "adam", "samuel", "steve", "joseph"];
var shortNames = names.filter(function(name) {
  if(name.length <= 4) {
    return true;
  }
  return false;
});

#4 reduce()
its a function that is used for applying a formula/operation and accumulate the value into single outcome
var numbers = [0, 1, 2, 3, 7, 4 12, 18, 15];
var evenNumbers = 0;
for(var i=0;i<numbers.length;i++) {
  if(numbers[i] % 2 == 0) {
    evenNumbers++;
  }
}

var nevens = numbers.reduce(function(total, n, idx, array) {
  if(n % 2 == 0) {
    total = total+1;
  }
  return total;
});

Note:- total will be assigned to the first element of the array for the first iteration.

#5 every()
For every element in the array meets the criteria then return true otherwise false.

var prime = [7, 3, 11, 15];
var isPrimeArray = prime.every(function(n) {
  var f = true;
  for(var i=2;i<n/2;i++) {
    if(n % i == 0) {
```

```
      f = false;
    }
  }
  return f;
});

#6 some()
if pass a function to some in which we write a conditional expression, the function will be called on each value of the array.
  - If atleast one of the value in the array pasess throught the conditional expression it returns true otherwise it returns false

has negative numbers?
var numbers = [10 , 20, 11 ,23, -10];
var hasNegativeNumbers = numbers.some(function(n) {
  if(n < 0) {
    return true;
  }
  return false;
});

#7 indexOf()
is a function used for finding the index position of a value.
var fruits = ["banana","apple","orange"];
fruits.indexOf("apple");

#8 lastIndexOf()
finds the last index position of an element within the array
var fruits = ["banana","apple","orange","apple"];
var lindex = fruits.lastIndexOf("apple"); // returns 3

#9 find() = used for searching for an element based on criteria and returning it
var numbers = [10, 20, -10, 9, 1, 6];

var neNumbers = numbers.find(function(n) {
  if(n < 0) {
    return true;
  }
  return false;
});

#10 findIndex() = returns the index position of the element where the number was found
-------------------------------------------------------------------------------
```

```
Control Statements
  2 Types are there
    1) Conditional control statements
      if
      if-else
      if-else-if
      switch
    2) Loop control statements
    for
    for in loop
    while

How to access html elements of a page in javascript?
- we can uniquely access an html element on a web page in javascript using element id. so every element should be binded with unique id to make it accessible directly in javascript
<input type="text" name="name" id="name" value=""/>
<script>
  var nameInpt = document.getElementById("name"); // input[type=text] object will get
  var name = nameInpt.value;
</script>

The Math functions available in javascript are
Math
-----
random()
floor()
round()
ciel
min()
max()
------------------------------------------------------------------------------------------
Javascript Objects:
-------------------
In javascript everything is almost an object
  1. Number
  2. String
  3. Date
  4. Function
  5. Array
  6. Object
Every object has attributes and methods as part of them
There are many ways we can create Objects in javascript.
  #1 using curly brackets

  var permanentAddress = {
    streetAddress: "2nd lane, beside park",
```

```
    city: "hyderabad",
    state: "TS",
    zip: 87877
  }

  var temporaryAddress = {
    streetAddress: "telephone exchange steet",
    city:"hyderabad",
    state: "ts",
    zip: 93833
  }


  #2 using Object notation we can create javascript object

  var address = new Object();
  address.streetAddress = "2nd lane, beside park";
  address.city = "hyderabad";
  address.state= "TS";

  #3 javascript function constructor
  var address = function(streetAddress, city, state) {
    this.streetAddress = streetAddress;
    this.city = city;
    this.state = state;
  }
  var streetAddress = address("2nd lane","hyderabad",
  "ts");


  #4 using class declaration
    class FullAddress {
      // mandatory the name should be constructor only
      constructor(addressLine1, addressLine2, city,
      state, zip, country) {
          this.addressLine1 = addressLine1;
          this.addressLine2 = addressLine2;
          this.city = city;
          this.state = state;
          this.zip = zip;
          this.country = country;
      }
      toString() {
          return JSON.stringify(this);
      }
    }
  ------------------------------------------------------
  ------------------------------------------------
  Object Prototype in javascript
  Every object in javascript has its own prototype, a
```

```
prototype refers a skeleton structure of an object. we
can imagine this as a reflection package in java

Date object has its prototype as Date.prototype
Array object has its prototype as Array.prototype

var Passport = function(passportNo, passportHolderName,
age) {
  this.passportNo = passportNo;
  this.passportHolderName = passportHolderName;
  this.age = age;
}
Passport.protoype.gender = "Male";
Passport.prototype.isValid = function() {
    return true;
}

var passport1 = new Passport('pa9292', 'james',23);
passport1.gender = "Female";
passport1.isValid();
------------------------------------------------------------
-----------------------------------------------
HTML DOM
DOM Stands document object model, its a programming model
that can be used for expressing HTML Page elements in
terms of Objects, so that we can access these objects and
we can modify.

using the DOM Programming model we can perform the
following this:
  1. We can access all the elements of our page using DOM
  api
  2. We can change attributes and the values of the
  objects
  3. We can modify the styles of the HTML Elements
  4. We can bind even listeners using the HTMLDOM Elements
thus making your HTML Page dynamic

Browser while loading the HTML Page we provided, it
constructs an in-memory dom model of the HTML Page
elements and places in the memory as a Tree structure, so
that programmers through javascript can access these
objects and can manipulate.

Everything by default in DOM is Node, a Node holds
data/contents of tag and relationship with other Nodes.
Browser while loading the HTML Page representing each Tag
in constructs Node in DOM Tree with relationship of other
Nodes as Parent and Child.
```

There are different types of Nodes are there based on type of the elements your page contains

- Anchor
- Input
- Select
- Button etc

Representing the type of the tag it constructs respective node type in DOM Tree

---

The Top-level object of DOM is document representing your HTML document. The First-Level object of your page is HTML. The secondlevel object is head, body

They can be multiple objects of different types at 3rd level like

- anchor, buttons, input, select, div, p, span

How to navigate between the elements of the DOM Tree?

There are traversal methods are there using which we can access the elements through relationship.

1. firstChild
2. childNodes
3. nextSibling
4. prevSibling
5. parent

For e.g..

```
<html>
  <head>
    <title>DOM Tree</title>
  </head>
  <body>
    <a href="#">None</a>
    <script>
      var htmlNode = document.firstNode;
      alert(htmlNode);
    </script>
  </body>

</html>
```

The document object has provided convinients method to access the page elements instead of using relationship methods above

1. getElementsById() = to access a html dom object of a HTML element using id
2. getElementsByTagName("div") = we can elements based on tag names
3. getElementsByClassName(cssClass) = we can access all the elements based on css classes

We can change the HTML Elements
1. element.innerHTML
2. element.attribute = value
3. element.style.background = newStyle

We can create elements, add and remove the elements in HTML page using document
1. document.createElement(element)
2. document.removeElement(element)
3. document.appendChild(element)
4. document.replaceChild(newElement, existingElement)
5. document.write(textcontent)

We can bind events with functions
element.onclick = function() {}

We can find all the elements of an html document using predefined attributes types in document
  1. document.forms = returns all form objects on the page
  2. document.body = body object
  3. document.head = head object
  4. document.scripts = return all script elements
  5. document.anchors = returns all a tags
  6. document.documentURI = returns URI of the document
  7. document.domain = returns host
  8. document.URL = gets the complete url of the browser

We can execute a javascript handler function upon clicking on a button by writing onclick attribute on the button tag. attaching a javascript handler to an event of an object is called "event binding technic"
There are 2 ways we can do event binding
1. static binding = at the time of declaring the HTML element we bind that to javascript handler which is called static binding

2. dynamic binding = at runtime based on conditions we are going to attach event handlers to the componets/page elements called "Dynamic Binding"

<button onclick="place();">place</button> = this is called static binding