JavaScript

1.1 Interview Questions

1) **What is JavaScript?**
JavaScript is a scripting language.
It is a case sensitive language.
It is an object based language.
It is a loosely typed checking language.
It was developed by Brendan Eich in 1995.
The original name of JavaScript is LiveScript.

2) **Advantages of JavaScript?**
Speed/Faster
Simplicity
Interoperability
Versatility
Rich interfaces
Reduce Server Load
No compiler and No interpreter
Weakly typed language
Platform independent
Client Side validation

3) **Disadvantages of JavaScript?**
Client-Side Security
Browser Support
Stop Rendering
Slow Bitwise Operation
Single Inheritance

4) **Difference between Java and JavaScript?**

Java	JavaScript
It is a non-scripting language.	It is a scripting language.
Java can run individually.	JavaScript can't run individually.
Java does not required browser window for execution	JavaScript requires browser window for execution.
It is an object oriented programming language.	It is an object based programming language.
It is a strongly typed checking language.	It is a loosely typed checking language
It is complex language.	It is easy language.

5) **What is JavaScript Engine?**

JavaScript Engine is software or a program which converts user understandable scripting language to machine understandable scripting language.
By default, every browser contains JavaScript engine.
The following lists of JavaScript engine present in a browser are.
Ex:

Browser	JavaScript Engine
1) Chrome	V8 Engine
2) Firefox	Spidermonkey
3) Safari	javascriptcore
4) Edge	Chakra

6) **Types of JavaScript?**

We have two type of JavaScript.

- **Internal JavaScript / Embedded JavaScript**
  It contains html code and JavaScript code inside a single file with ".html" extension.
- **External JavaScript / External JavaScript**
  It contains html code in ".html" file and JavaScript code in ".js" file.

7) **Types of functions in JavaScript?**

We have two types of functions.

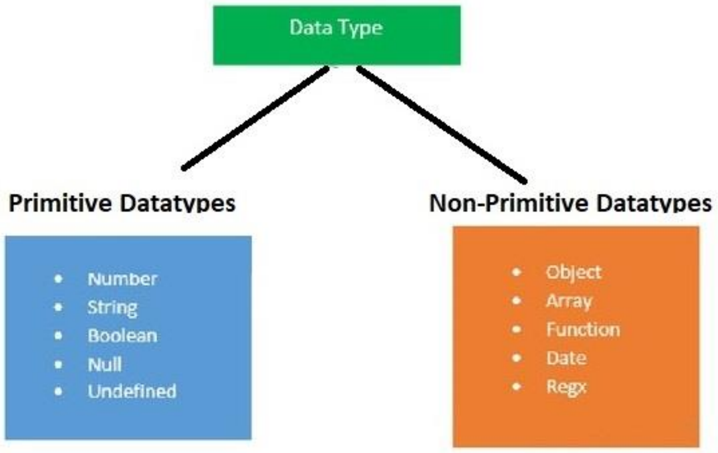
- **Named Functions**
  These types of functions contains name at the time of definition.
  Ex:
```
function f1()
{
        document.writeln("hello world");
}
f1();
```
- **Anonymous Functions**
  These types of functions don't contain any name.
  They are declared dynamically at runtime.
  Ex:
```
var f1=function(){
        document.writeln('hello world');
}
f1();
```

8) **What is JavaScript Closure?**

A closure is the combination of a function bundled together along lexical scope.
In other words, a closure gives you access to an outer function's scope from an inner function.
In JavaScript, closures are created every time when function is created.

Ex:
```
var a=10;
function f1()
{
    var b=20;
    function f2()
    {
        var c=30;
        document.writeln(a+b+c);
    }
    f2();
}
f1();
```

9) Explain JavaScript Data Types?



10) Difference between null and undefined?

In JavaScript, undefined is a type, whereas null an object.

- **undefined**

  It means a variable declared, but no value has been assigned a value.

  Ex:
  ```
  <script>
      var x;
      document.writeln(x); //undefined
  </script>
  ```
- **null**

  A null in JavaScript is an assignment value.

  Ex:
  ```
  <script>
      var x=null;
      document.writeln(x); //null
  </script>
  ```

11) Difference between == and ===?

- **==:**
  It is used to check only equality
  Ex:

```
<script>
        document.writeln ((10==10) +"<br>");//true
        document.writeln ((10=="10") +"<br>");//true
        document.writeln ((false==false)+"<br>");//true
        document.writeln ((0==false)+"<br>");//true
        document.writeln ((true==true)+"<br>");//true
        document.writeln ((1==true)+"<br>");//true
</script>
```

- **===:**
  It is used to check equality and datatype.
  Ex:

```
<script>
        document.writeln((10===10) +"<br>");//true
        document.writeln((10==="10") +"<br>");//false
        document.writeln((false===false)+"<br>");//true
        document.writeln((0===false)+"<br>");//false
        document.writeln((true===true)+"<br>");//true
        document.writeln((1===true)+"<br>");//false
</script>
```

12) Difference between innerText and innerHTML?

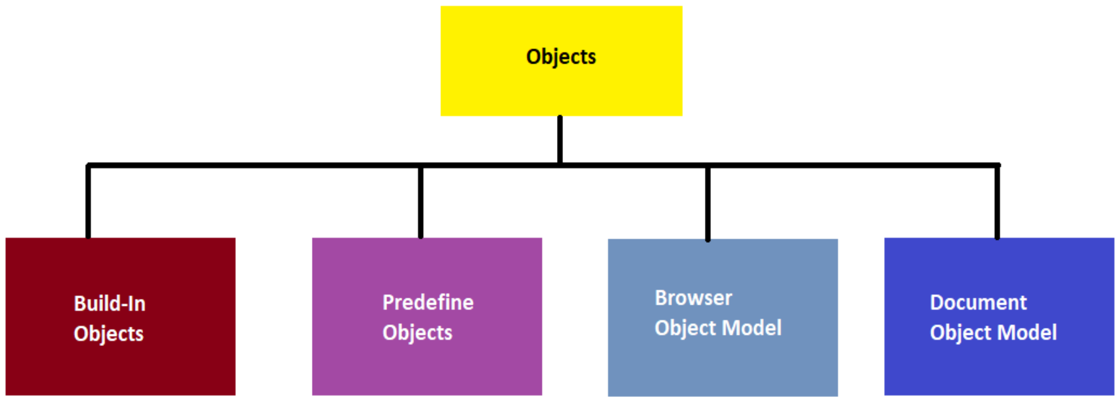
- **innerText:**
  The innerText property is used to write the simple text using JavaScript dynamically.
  Ex:

```
<div id="result"></div>
<script>
document.getElementById ('result').innerText="Simple Text";
</script>
```

- **innerHTML:**
  The innerHTML property is used to write the HTML code using JavaScript dynamically.
  Ex:

```
<div id="result"></div>
<script>
document.getElementById ('result').innerHTML="<h1>HTML Text</h1>";
</script>
```

13) Types of objects in JavaScript?



❖ **Build-In Objects:**

Built-in objects are provided by JavaScript.
Built-in objects are not related to any Window or DOM.
These objects are used for simple data processing in the JavaScript.
Ex:

String
Number
Boolean
Math
Array

❖ **Predefine Objects:**

Objects which are created by the user based on the requirement are called custom objects.
JavaScript provides built-in object called Object.
We can use "Object" to create custom objects in JavaScript.
Ex:

```
<script>
        var emp={id:101,name:"IHUB TALENT",salary:40000}
        document.writeln (emp.id+" "+emp.name+" "+emp.salary);
</script>
```

❖ **Browser Object Model:**

The Browser Object Model is used to interact with browser.
The default object for a browser is window object.
We can call all the functions/methods by using window or directly.
Ex:

```
window.alert("Welcome to JavaScript");
or
alert("Welcome to JavaScript");
```

❖ **<u>Document Object Model:</u>**

When a web page is loaded, the browser creates a Document Object Model of the page.
Using document object, Our HTML document is represented in a tree node hierarchy.
A document object is a root node for HTML document tree node structure/hierarchy.
DOM always looks for three nodes.

1) Element node
2) Attribute node
3) Text node

Using document object we can add dynamic content to the web page.
A document object is a property of window.
It means we can call document object directory or by using window.
Ex:

```
window.document
or
document
```

14) What is JavaScript hoisting?

Hoisting is the default behavior of JavaScript where all the variable and function declarations are moved on top.
This means that irrespective of where the variables and functions are declared, they are moved on top of the scope.
The scope can be both local and global.
Ex1:

```
hoistedVariable = 3;                  var hoistedVariable;
console.log(hoistedVariable);   ==>   console.log(hoistedVariable);
var hoistedVariable;                  var hoistedVariable;
```

Ex2:

```
hoistedFunction();  //calling         function hoistedFunction()
function hoistedFunction()            {
{                               ==>   console.log(" Hello world! ");
console.log(" Hello world! ");        }
}                                     hoistedFunction();  //calling
```

15) What is window object in JavaScript?

It is used to create a window on a browser.
A window object is created atomically by the browser.
A "window" is object of browser but not JavaScript.
A "window" object is used to write programming related to browser.
With the help of window object we can perform following activities very easily.

- ✓ It display dialog boxes and pop boxes.
- ✓ We can find width and height of a browser.
- ✓ We can move or resize the browser.
- ✓ Scroll to the browser.
- ✓ Get URL, hostname, protocol and etc. of a browser.
- ✓ We can get JavaScript history.

16) List of some methods present in window object?

We have following methods in window object.

Method	**Description**
alert()	It will display alert box.
confirm()	It will display confirm dialog box.
prompt()	It will display prompt dialog box.
open()	Open a new window
close()	Close the current window
moveTo()	Move the current window
moveBy()	Move the current window
resizeTo()	Resize the current window

17) List of some methods present in document object?

We have following methods in document object.

Method	**Description**
getElementById (id)	Find an element by element id
getElementsByTagName (name)	Find elements by tag name
getElementsByClassName (class)	Find elements by class name
createElement(element)	Create an HTML element
removeChild(element)	Remove an HTML element
appendChild(element)	Add an HTML element
replaceChild(new, old)	Replace an HTML element
write(text)	Write into the HTML output stream

18) Explain High order functions in JavaScript?

Functions that operate on other functions, either by taking them as arguments or by returning them, are called higher-order functions.

Ex:

```
<script>
        function higherOrder (fn) {
        fn();
        }
        higherOrder(function() {document.writeln("Hello world") });
</script>
```

Ex:
```
<script>
        function higherOrder2()
        {
                return function() {
                         return "Do something";
                }
        }
        var x = higherOrder2();
        x();
</script>
```

19) What is currying in JavaScript?

Currying simply means evaluating functions with multiple arguments and decomposing them into a sequence of functions with a single argument.
By using the currying technique, we do not change the functionality of a function, we just change the way it is invoked.
Ex:
```
<script>
        function add (a) {
                return function(b){
                        return a + b;
                }
        }
        add(3)(4) ;
</script>
```
Ex:
```
<script>
        function multiply(a,b){
         return a*b;
        }

        function currying(fn){
                return function(a){
                        return function(b){
                                return fn(a,b);
                                }
                        }
                }
        multiply(4, 3); // Returns 12
        var curriedMultiply = currying(multiply);
        curriedMultiply(4)(3); // Also returns 12
</script>
```

20) What is the difference between exec () and test () methods in JavaScript?

- ✓ test () and exec () are RegExp expression methods used in JavaScript.
- ✓ We'll use exec () to search a string for a specific pattern, and if it finds then it will return the pattern directly otherwise t will return an 'empty' result.
- ✓ We will use a test () to find a string for a specific pattern. It will return the Boolean value 'true' on finding the given text otherwise, it will return 'false'.

21) What are the types of errors in JavaScript?

There are two types of errors in JavaScript.

- **Syntax error:**
  Syntax errors are mistakes or spelling problems in the code that cause the program to not execute at all or to stop running halfway through.
- **Logical error:**
  Reasoning mistakes occur when the syntax is proper but the logic or program is incorrect. The application executes without problems in this case. However, the output findings are inaccurate.

22) What is Recursion in JavaScript?

A function which calls itself for many numbers of times is called recursion.

Ex:

```
<script>
        function add(number) {
                if (number <= 0) {
                        return 0;
                } else {
                        return number + add(number - 1);
                }
        }
        add(3)
</script>
```

Explanation:

```
=> 3 + add(2) => 3 + 2 + add(1) => 3 + 2 + 1 + add(0) =>  3 + 2 + 1 + 0 = 6
```

23) What is rest parameter in JavaScript?

It provides an improved way of handling the parameters of a function.
Any number of arguments will be converted into an array using the rest parameter.
Rest parameters can be used by applying three dots (...) before the parameters.

Ex:

```
<script>
        function f1(...args) { document.writeln("hello");}
        f1(10);
        f1(10,20);
        f1(10,20,30);
</script>
```

24) What is spread operator in JavaScript?

The spread operator is used to spreading an array.

Ex:

```
<script>
        function addFourNumbers(num1,num2,num3,num4){
                document.writeln (num1+" "+num2+" "+num3+" "+num4);
        }
        let fourNumbers = [5, 6, 7, 8];
        addFourNumbers(...fourNumbers);
</script>
```

25) What is Object Destructuring?

Object destructuring is a new way to extract elements from an object or an array.

Ex:

```
<script >
        const classDetails = {
                        no: 101,
                        name: "Alan",
                        add:"hyd"
        }
        const {no:classNo, name:className,add:classAdd} = classDetails;
        document.writeln(classNo);//101
        document.writeln(className);//Alan
        document.writeln(classAdd);//hyd
</script>
```

Ex:

```
<script>
        const arr = [1, 2, 3, 4];
        const [first,second,third,fourth] = arr;
        document.writeln(first); // Outputs 1
        document.writeln(second); // Outputs 2
        document.writeln(third); // Outputs 3
        document.writeln(fourth); // Outputs 4
 </script>
```

26) Difference between localStorage and sessionStorage?

- **<u>localStorage:</u>**

  A localStorage property allows us to save key/value pair in a browser window.
  A localStorage will store the data with no expiry date. Our data will not be removed if we close the browser window or tab. It will be present in next days.

  Ex:

  ```
  localStorage.setItem("name","Alan");
  var val1=localStorage.getItem("name");
  ```

- **<u>sessionStorage:</u>**
  A sessionStorage property allows us to save key/value pair in a browser window. A sessionStorage store the values based on one session. Our data will be deleted once we close the browser or tab.
  Ex:
  ```
              sessionStorage.setItem("name","Alan");
              var val1=sessionStorage.getItem("name");
  ```

27) Differences between var, let and const in JavaScript?

var	let	const
It is a functional scope	It is a block scope	It is a block scope
It can be declared without initialization.	It can be declared without initialization.	It can't be declared without initialization.
It can be updated.	It can be updated.	It can't be updated.
It can be re-declared.	It can't be re-declared.	It can't be re-declared.
It can be access without initialization as it default value is undefined.	It can be access without initialization as it default value is undefined.	It can't be access without initialization.

28) Explain debugger keyword?

The debugger keyword stops the execution of JavaScript, and calls (if available) the debugging function.
This has the same function as setting a breakpoint in the debugger.
If no debugging is available, the debugger statement has no effect.
Ex:
```
<p id="demo"></p>
<p>With the debugger turned on, the code below should stop executing
        before it executes the third line.</p>
<script>
        let x = 15 * 5;
        debugger;
        document.getElementById("demo").innerHTML = x;
</script>
```

29) What is form validation?

The process of checking format and pattern of form data is called form validation.
There are two ways to perform form validation.

- **<u>Client Side Form Validation</u>**
  Form validation which is performed at client side is called client side form validation.
  To perform client side form validation we need to use JavaScript.

- **<u>Server Side Form Validation</u>**
  Form validation which is performed at server side is called server side form validation.
  To perform server side form validation we will use php, .net, java, python & etc.

30) What is Array in JavaScript?

A JavaScript array is an object which is collection of same or similar elements.
There are three ways to create JavaScript arrays.

- **<u>By using Array literal</u>**
  Ex:
  ```
  var arr=[10,20,30,40];
  var arr=["one","two","three","four"];
  ```

- **<u>By creating instance of an Array</u>**
  Ex:
  ```
  var arr=new Array();
  arr[0]=10;
  arr[1]=20;
  arr[2]=30;
  arr[3]=40;
  ```

- **<u>By creating Array constructor</u>**
  Ex:
  ```
  var arr=new Array(10,20,30,40);
  ```

31) What is String in JavaScript?

A JavaScript String is an object which is collection of characters.
There are two ways to create JavaScript string.

- **<u>By using String literal</u>**
  Ex:
  ```
  var str="This is IHUB Talent";
  ```

- **<u>By create instance of a String</u>**
  Ex:
  ```
  var str=new String("IHUB Talent");
  ```

32) What is RegEx?

In JavaScript, regular expressions are also objects.
A regular expression is a sequence of characters that forms a search pattern in text.
Usually such patterns are used by string-searching algorithms for "find" or "find and replace" operations on strings, or for input validation.
Ex:
```
[A-Za-z. ]{6,20}$
```

33) What is JavaScript Event?

- ✓ The change in the state of an object is known as an Event.
- ✓ In html, there are various events which represents that some activity is performed by the user or by the browser.
- ✓ When JavaScript code is included in HTML, JavaScript react over these events and allow the execution. This process of reacting over the events is called Event Handling.
- ✓ JavaScript handles the HTML events via Event Handlers.
- ✓ We have different types of events.

Mouse Events:

Event Performed	Event Handler	Description
Click	Onclick	When mouse click on an element
mouseover	onmouseover	When the cursor of the mouse comes over the element
mouseout	onmouseout	When the cursor of the mouse leaves an element
mousedown	onmousedown	When the mouse button is pressed over the element
mouseup	onmouseup	When the mouse button is released over the element
mousemove	onmousemove	When the mouse movement takes place.

Form Events:

Event Performed	Event Handler	Description
Focus	Onfocus	When the user focuses on an element
submit	Onsubmit	When the user submits the form
Blur	Onblur	When the focus is away from a form element

Keyboard Events:

Event Performed	Event Handler	Description
Keydown & Keyup	onkeydown & onkeyup	When the user press and then release the key

Document/Window Event:

Event Performed	Event Handler	Description
Load	Onload	When the browser finishes the loading of the page
Unload	Onunload	When the visitor leaves the current webpage, the browser unloads it
Resize	Onresize	When the visitor resizes the window of the browser

34) What is Synchronous JavaScript?

As the name suggests synchronous means to be in a sequence, i.e. every statement of the code gets executed one by one. So, basically a statement has to wait for the earlier statement to get executed.

Ex:

```
<script>
        document.write("Hi"); // First
        document.write("<br>");
        document.write("Mayukh") ;// Second
        document.write("<br>");
        document.write("How are you"); // Third
</script>
```

35) What is Asynchronous JavaScript?

Asynchronous code allows the program to be executed immediately where the synchronous code will block further execution of the remaining code until it finishes the current one. This may not look like a big problem but when you see it in a bigger picture you realize that it may lead to delaying the User Interface.

Ex:

```
<script>
        document.write("Hi");
        document.write("<br>");
```

```
            setTimeout(() => {
                document.write("Let us see what happens");
            }, 2000);
            document.write("<br>");
            document.write("End");
            document.write("<br>");
        </script>
```

36) Explain JavaScript promises?

Promises are used to handle asynchronous operations in JavaScript.
They can handle multiple asynchronous operations easily and provide better error handling than callbacks and events.
A Promise has four states:
fulfilled: Action related to the promise succeeded
rejected: Action related to the promise failed
pending: Promise is still pending i.e. not fulfilled or rejected yet
settled: Promise has fulfilled or rejected
Ex:

```
    <script>
        var promise = new Promise(function(resolve, reject) {
        const x = "ihubtalent";
        const y = "ihubtalent1";
        if(x === y) {
            resolve();
        } else {
            reject();
        }
        });
        promise.
            then(function () {
                console.log('Success, You are a GEEK');
            }).
            catch(function () {
                console.log('Some error has occurred');
            });
</script>
```

37) What is JavaScript Math object?

The JavaScript Math object allows you to perform mathematical tasks on numbers.
Ex:

Math.round(x)	Returns x rounded to its nearest integer
Math.ceil(x)	Returns x rounded up to its nearest integer
Math.floor(x)	Returns x rounded down to its nearest integer
Math.trunc(x)	Returns the integer part of x (new in ES6)

38) What is class in JavaScript?

A JavaScript class is not an object.
It is a template for JavaScript objects.
Use the class keyword to create a class.
A class keyword is used to declare a class with any particular name.
According to JavaScript naming conventions, the name of the class always starts with an uppercase letter.
Ex:

```
<script>
        class Example
        {
                -
                -//code to be declare
                -
        }
</script>
```

39) What is Constructor in JavaScript?

A JavaScript constructor method is a special type of method which is used to initialize and create an object.
It is called when memory is allocated for an object.
The constructor keyword is used to declare a constructor method.
The class can contain one constructor method only.
JavaScript allows us to use parent class constructor through super keyword.
Ex:

```
class Example
{
        constructor()
        {
                -
                -// code to be declare
                -
        }
 }
```

40) What is object in JavaScript?

A JavaScript object is an entity having state and behavior (properties and method).
Syntax:

```
var objectname =new Object();
```

41) What is Encapsulation in JavaScript?

The process of wrapping property and function within a single unit is known as encapsulation.
To achieve an encapsulation in JavaScript we need to do following things.

- ✓ Use var keyword to make data members private.
- ✓ Use setter methods to set the data and getter methods to get that data.

` Ex:

```
class person{
         constructor(name,id){
                  this.name = name;
                  this.id = id;
         }
         add_Address(add){
                  this.add = add;
         }
         getDetails(){
         console.log(`Name is ${this.name},Address is: ${this.add}`);
         }
}
let person1 = new person('Mukul',21);
person1.add_Address('Delhi');
person1.getDetails();
```

42) What is Inheritance in JavaScript?

The JavaScript inheritance is a mechanism that allows us to create new classes on the basis of already existing classes.

It provides flexibility to the child class to reuse the methods and variables of a parent class.

The JavaScript extends keyword is used to create a child class on the basis of a parent class.

Ex:

```
<script>
         class Moment extends Date {
                  constructor() {
                  super();
         }}
         var m=new Moment();
         document.writeln("Current date:")
 document.writeln(m.getDate()+"-"+(m.getMonth()+1)+"-"+m.getFullYear());
</script>
```

43) What is Abstract in JavaScript?

Hiding internal implementation and highlighting the set of services is called Abstraction.

The best example of Abstraction is GUI(Graphical User Interface) ATM machine where bank people will hide internal implementation and highlights the set of services like banking, withdrawal, mini statement, balance enquiry and etc.

44) What is polymorphism in JavaScript?

The process of encapsulating data members and it's behaviors in a single entity is called polymorphism.

Ex:

```
class A
{
        display()
        {
                document.writeln("A is invoked<br>");
        }
}
 class B extends A
{
        display()
        {
                document.writeln("B is invoked");
        }
}
A a=new A();
a.display(); // A is invoked
B b=new B();
b.display(); // B is invoked
```

45) What is addEventListener in JavaScript?

The addEventListener() method attaches an event handler to the specified element.
We can add many event handlers to one element.
We can easily remove an event listener by using the removeEventListener() method.
Ex:

```
p.addEventListener("click", function(){ alert("Hello World!"); });
```