JAVASCRIPT Interview Questions:-

1.What is V8 JavaScript engine

V8 is an open source high-performance JavaScript engine used by the Google Chrome browser, written in C++. It is also being used in the node.js project. It implements ECMAScript and WebAssembly, and runs on Windows 7 or later.

Note: It can run standalone, or can be embedded into any C++ application.

2.What is jQuery

jQuery is a popular cross-browser JavaScript library that provides Document Object Model (DOM) traversal, event handling, animations and AJAX interactions by minimizing the discrepancies across browsers. It is widely famous with its philosophy of “Write less, do more”. For example, you can display welcome message on the page load using jQuery as below,

```
$(document).ready(function () {
  // It selects the document and apply the function on page load
  alert("Welcome to jQuery world");
});
```

Note: You can download it from jquery's official site or install it from CDNs, like google.

3.What are the different methods to find HTML elements in DOM

If you want to access any element in an HTML page, you need to start with accessing the document object. Later you can use any of the below methods to find the HTML element,

i. document.getElementById(id): It finds an element by Id
ii. document.getElementsByTagName(name): It finds an element by tag name
iii. document.getElementsByClassName(name): It finds an element by class name

4.How do you load CSS and JS files dynamically

You can create both link and script elements in the DOM and append them as child to head tag

function loadCSS(url) {

const link = document.createElement('link');

link.rel = 'stylesheet';

link.href = url;

document.head.appendChild(link);

}

// Usage:

loadCSS('path/to/styles.css');

5.How do get the timezone offset from date

You can use the `getTimezoneOffset` method of the date object. This method returns the time zone difference, in minutes, from current locale (host system settings) to UTC

```
var offset = new Date().getTimezoneOffset();
console.log(offset); // -480.
```

6.How do you declare namespace?

In JavaScript, there isn't a direct built-in concept of namespaces like in some other programming languages. However, you can simulate namespaces by organizing your code using objects to create a similar structure.

Here's an example of how you can create namespaces in JavaScript:

// Creating a namespace using an object literal const MyNamespace = { someFunction: function() { console.log('This is a function inside MyNamespace'); }, someVariable: 'This is a variable inside MyNamespace' }; // Accessing elements within the namespace

MyNamespace.someFunction(); // Calling the function
console.log(MyNamespace.someVariable); // Accessing the variable

In this example, **`MyNamespace`** is an object that acts as a namespace. Functions, variables, or other objects can be added as properties within this namespace. Access to these properties is achieved by using dot notation (**`MyNamespace.someFunction()`** or **`MyNamespace.someVariable`**).

Another approach involves using closures to create namespaces: const MyNamespace = (function() { const privateVariable = 'This is private'; return { publicFunction: function() { console.log('This is a public function'); }, publicVariable: 'This is a public variable' }; })(); // Accessing elements within the namespace MyNamespace.publicFunction(); // Calling the function console.log(MyNamespace.publicVariable); // Accessing the variable

7.Does JavaScript supports namespace

JavaScript doesn’t support namespace by default. So if you create any element(function, method, object, variable) then it becomes global and pollutes the global namespace. Let's take an example of defining two functions without any namespace,

```
function func1() {
  console.log("This is a first definition");
}
function func1() {
  console.log("This is a second definition");
}
func1(); // This is a second definition
```

It always calls the second function definition. In this case, namespace will solve the name collision problem.

8.What is the difference between java and javascript

Both are totally unrelated programming languages and no relation between them. Java is statically typed, compiled, runs on its own VM. Whereas Javascript is dynamically typed, interpreted, and runs in a browser and nodejs environments. Let's see the major differences in a tabular format,

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

9.How do you print numbers with commas as thousand separators

You can use the `Number.prototype.toLocaleString()` method which returns a string with a language-sensitive representation such as thousand separator,currency etc of this number.

```
function convertToThousandFormat(x) {
  return x.toLocaleString(); // 12,345.679
}

console.log(convertToThousandFormat(12345.6789));
```

10.How to get the value from get parameters?

In JavaScript, you can retrieve the values from the URL query parameters (also known as "GET parameters") using the **`URLSearchParams`** interface, which provides methods to work with the query string of a URL.

Here's an example of how to get the value from GET parameters using **`URLSearchParams`**:

// Assuming the URL is something like:
https://example.com/page?param1=value1&param2=value2

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
const urlParams = new URLSearchParams(window.location.search);

// Retrieving specific parameter values
const param1Value = urlParams.get('param1');
const param2Value = urlParams.get('param2');

console.log(param1Value); // Outputs: value1
console.log(param2Value); // Outputs: value2
```

11.How do you check whether an array includes a particular value or not

The `Array#includes()` method is used to determine whether an array includes a particular value among its entries by returning either true or false. Let's see an example to find an element(numeric and string) within an array.

```
var numericArray = [1, 2, 3, 4];
console.log(numericArray.includes(3)); // true

var stringArray = ["green", "yellow", "blue"];
console.log(stringArray.includes("blue")); //true
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

12.How do I modify the url without reloading the page?

In JavaScript, you can modify the URL without reloading the page using the **`history.pushState()`** method. This method is part of the History Web API, which allows you to manipulate the browser history and change the URL without triggering a full page reload.

```
// Replace 'newpath' and 'newTitle' with your desired path and title
const newpath = '/new-path'; // New path or URL
const newTitle = 'New Page Title'; // New page title (optional)

// Change the URL without reloading the page
history.pushState(null, newTitle, newpath);
```

13.How do you list all properties of an object

You can use the `Object.getOwnPropertyNames()` method which returns an array of all properties found directly in a given object. Let's the usage of it in an example,

```
const newObject = {
  a: 1,
  b: 2,
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
  c: 3,
};

console.log(Object.getOwnPropertyNames(newObject));
["a", "b", "c"];
```

14.Is enums feature available in javascript

No, javascript does not natively support enums. But there are different kinds of solutions to simulate them even though they may not provide exact equivalents. For example, you can use freeze or seal on object,

```
var DaysEnum = Object.freeze({"monday":1, "tuesday":2,
"wednesday":3, ...})

15.What is an enum
```

An enum is a type restricting variables to one value from a predefined set of constants. JavaScript has no enums but typescript provides built-in enum support.

```
enum Color {
 RED, GREEN, BLUE
}
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

16.What is an Unary operator

The unary(+) operator is used to convert a variable to a number.If the variable cannot be converted, it will still become a number but with the value NaN. Let's see this behavior in an action.

```
var x = "100";
var y = +x;
console.log(typeof x, typeof y); // string, number

var a = "Hello";
var b = +a;
console.log(typeof a, typeof b, b); // string,
number, NaN
```

17.How do you sort elements in an array

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

The sort() method is used to sort the elements of an array in place and returns the sorted array. The example usage would be as below,

```
var months = ["Aug", "Sep", "Jan", "June"];
months.sort();
console.log(months); //  ["Aug", "Jan", "June",
"Sep"]
```

18.What is the purpose of compareFunction while sorting arrays

1. The compareFunction is used to define the sort order. If omitted, the array elements are converted to strings, then sorted according to each character's Unicode code point value. Let's take an example to see the usage of compareFunction,

```
let numbers = [1, 2, 5, 3, 4];
numbers.sort((a, b) => b - a);
console.log(numbers); // [5, 4, 3, 2, 1]
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

19.How do you reversing an array

2. You can use the reverse() method to reverse the elements in an array. This method is useful to sort an array in descending order. Let's see the usage of reverse() method in an example,

```
let numbers = [1, 2, 5, 3, 4];
numbers.sort((a, b) => b - a);
numbers.reverse();
console.log(numbers); // [1, 2, 3, 4 ,5]
```

20.How do you find min and max value in an array

You can use `Math.min` and `Math.max` methods on array variables to find the minimum and maximum elements within an array. Let's create two functions to find the min and max value with in an array,

```
var marks = [50, 20, 70, 60, 45, 30];
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
function findMin(arr) {
  return Math.min.apply(null, arr);
}
function findMax(arr) {
  return Math.max.apply(null, arr);
}

console.log(findMin(marks));
console.log(findMax(marks));
```

1. 21.How do you find min and max values without Math functions

You can write functions which loop through an array comparing each value with the lowest value or highest value to find the min and max values. Let's create those functions to find min and max values,

```
var marks = [50, 20, 70, 60, 45, 30];
function findMin(arr) {
  var length = arr.length;
  var min = Infinity;
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
  while (length--) {
    if (arr[length] < min) {
      min = arr[length];
    }
  }
  return min;
}

function findMax(arr) {
  var length = arr.length;
  var max = -Infinity;
  while (length--) {
    if (arr[length] > max) {
      max = arr[length];
    }
  }
  return max;
}

console.log(findMin(marks));
console.log(findMax(marks));
```

22.What is an event queue

n JavaScript, the event queue is an essential part of the event-driven architecture that manages the execution of asynchronous and non-blocking code. It is closely related

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

to the event loop, which is a crucial component of how JavaScript handles asynchronous operations.

When asynchronous tasks or events occur in JavaScript (such as DOM events, setTimeout callbacks, promises, etc.), they are not immediately executed. Instead, they are placed into a queue called the "event queue" to be processed later. The event queue operates in conjunction with the call stack and the event loop.

22.What is call stack

Call Stack is a data structure for javascript interpreters to keep track of function calls(creates execution context) in the program. It has two major actions,

i. Whenever you call a function for its execution, you are pushing it to the stack.
ii. Whenever the execution is completed, the function is popped out of the stack.

Let's take an example and it's state representation in a diagram format

```
function hungry() {
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

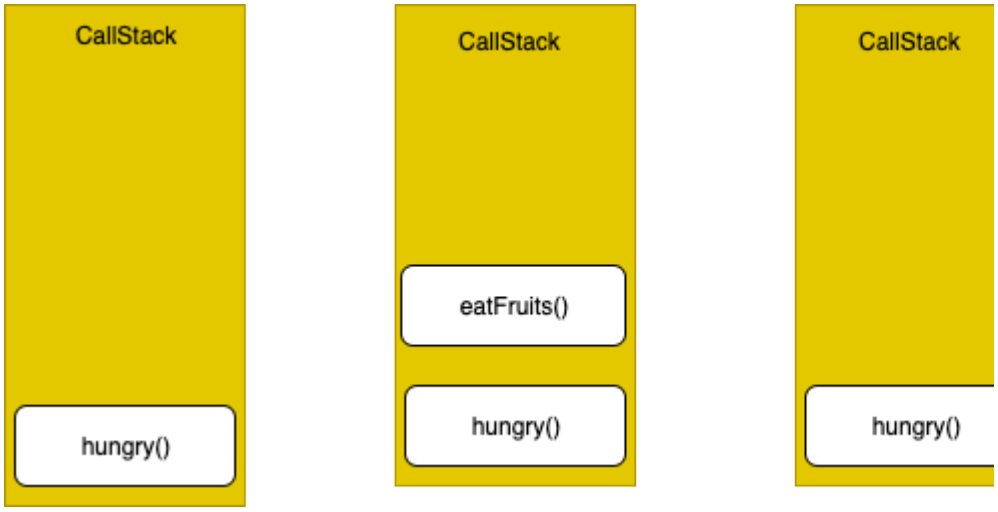
```
  eatFruits();
}
function eatFruits() {
  return "I'm eating fruits";
}

// Invoke the `hungry` function
hungry();
```

The above code processed in a call stack as below,

i. Add the `hungry()` function to the call stack list and execute the code.
ii. Add the `eatFruits()` function to the call stack list and execute the code.
iii. Delete the `eatFruits()` function from our call stack list.
iv. Delete the `hungry()` function from the call stack list since there are no items anymore.

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages



23.What is an event loop

The event loop is a process that continuously monitors both the call stack and the event queue and checks whether or not the call stack is empty. If the call stack is empty and there are pending events in the event queue, the event loop dequeues the event from the event queue and pushes it to the call stack. The call stack executes the event, and any additional events generated during the execution are added to the end of the event queue.

Note: The event loop allows Node.js to perform non-blocking I/O operations, even though JavaScript is single-

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

threaded, by offloading operations to the system kernel whenever possible. Since most modern kernels are multi-threaded, they can handle multiple operations executing in the background.

24.,How does synchronous iteration works

Synchronous iteration was introduced in ES6 and it works with below set of components,

Iterable: It is an object which can be iterated over via a method whose key is Symbol.iterator. Iterator: It is an object returned by invoking `[Symbol.iterator]()` on an iterable. This iterator object wraps each iterated element in an object and returns it via `next()` method one by one. IteratorResult: It is an object returned by `next()` method. The object contains two properties; the `value` property contains an iterated element and the `done` property determines whether the element is the last element or not.

Let's demonstrate synchronous iteration with an array as below,

```
const iterable = ["one", "two", "three"];
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
const iterator = iterable[Symbol.iterator]();
console.log(iterator.next()); // { value: 'one',
done: false }
console.log(iterator.next()); // { value: 'two',
done: false }
console.log(iterator.next()); // { value: 'three',
done: false }
console.log(iterator.next()); // { value:
'undefined, done: true }
```

25.What is an Iterator

An iterator is an object which defines a sequence and a return value upon its termination. It implements the Iterator protocol with a `next()` method which returns an object with two properties: `value` (the next value in the sequence) and `done` (which is true if the last value in the sequence has been consumed).

26.What is an Intl object

The Intl object is the namespace for the ECMAScript Internationalization API, which provides language sensitive string comparison, number formatting, and

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

date and time formatting. It provides access to several constructors and language sensitive functions.

1.🔗27.How do you perform language specific date and time formatting

You can use the `Intl.DateTimeFormat` object which is a constructor for objects that enable language-sensitive date and time formatting. Let's see this behavior with an example,

```
var date = new Date(Date.UTC(2019, 07, 07,
3, 0, 0));
```

28.What is nodejs

Node.js is a server-side platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. It is an event-based, non-blocking, asynchronous I/O runtime that uses

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

Google's V8 JavaScript engine and libuv library.

```
console.log(new Intl.DateTimeFormat("en-GB").format(date)); // 07/08/2019
console.log(new Intl.DateTimeFormat("en-AU").format(date)); // 07/08/2019
```

29.What are the two types of loops in javascript

1.

i. Entry Controlled loops: In this kind of loop type, the test condition is tested before entering the loop body. For example, For Loop and While Loop comes under this category.
ii. Exit Controlled Loops: In this kind of loop type, the test condition is tested or evaluated at the end of the loop body. i.e, the loop body will execute at least once irrespective of test condition true

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

or false. For example, do-while loop comes under this category.

30.What are the various statements in error handling

Below are the list of statements used in an error handling,

i. try: This statement is used to test a block of code for errors
ii. catch: This statement is used to handle the error
iii. throw: This statement is used to create custom errors.
iv. finally: This statement is used to execute code after try and catch regardless of the result.

30.What is an error object

An error object is a built in error object that provides error information when an error occurs. It has two

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

properties: name and message. For example, the below function logs error details,

```
try {
  greeting("Welcome");
} catch (err) {
  console.log(err.name + "<br>" + err.message);
}
```

31.What are the function parameter rules

JavaScript functions follow below rules for parameters,

i. The function definitions do not specify data types for parameters.
ii. Do not perform type checking on the passed arguments.
iii. Do not check the number of arguments received. i.e, The below function follows the above rules,

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
function functionName(parameter1,
parameter2, parameter3) {
  console.log(parameter1); // 1
}
functionName(1);
```

32.What are the different ways to access object properties

There are 3 possible ways for accessing the property of an object.

i. Dot notation: It uses dot for accessing the properties

```
objectName.property;
```

i. Square brackets notation: It uses square brackets for property access

```
objectName["property"];
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

i. Expression notation: It uses expression in the square brackets

```
objectName[expression];
```

33.What are primitive data types

A primitive data type is data that has a primitive value (which has no properties or methods). There are 7 types of primitive data types.

i. string
ii. number
iii. boolean
iv. null
v. undefined
vi. bigint
vii. symbol

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

34.How do you print the contents of web page

The window object provided a print() method which is used to print the contents of the current window. It opens a Print dialog box which lets you choose between various printing options. Let's see the usage of print method in an example,

```
<input type="button" value="Print"
onclick="window.print()" />
```

Note: In most browsers, it will block while the print dialog is open.

35.How do you decode an URL

The decodeURI() function is used to decode a Uniform Resource Identifier (URI) previously created by encodeURI().

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
var uri =
"https://mozilla.org/?x=шеллы";
var encoded = encodeURI(uri);
console.log(encoded); //
https://mozilla.org/?x=%D1%88%D0%B5
%D0%BB%D0%BB%D1%8B
try {
  console.log(decodeURI(encoded));
// "https://mozilla.org/?x=шеллы"
} catch (e) {
  // catches a malformed URI
  console.error(e);
}
```

36.How do you encode an URL

The encodeURI() function is used to encode complete URI which has special characters except (, / ? : @ & = + $ #) characters.

```
  var uri =
  "https://mozilla.org/?x=шеллы"
  ;
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
var encoded = encodeURI(uri);
console.log(encoded); //
https://mozilla.org/?x=%D1%88%
D0%B5%D0%BB%D0%BB%D1%8B
```

37.What is a WeakMap

The WeakMap object is a collection of key/value pairs in which the keys are weakly referenced. In this case, keys must be objects and the values can be arbitrary values. The syntax is looking like as below,

```
new WeakMap([iterable]);
```

Let's see the below example to explain it's behavior,

```
var ws = new WeakMap();
var user = {};
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
ws.set(user);
ws.has(user); // true
ws.delete(user); //
removes user from the map
ws.has(user); // false,
user has been removed
```

38.What are the differences between WeakMap and Map

The main difference is that references to key objects in Map are strong while references to key objects in WeakMap are weak. i.e, A key object in WeakMap can be garbage collected if there is no other reference to it. Other differences are,

i. Maps can store any key type Whereas WeakMaps can store

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

only collections of key objects

ii. WeakMap does not have size property unlike Map
iii. WeakMap does not have methods such as clear, keys, values, entries, forEach.
iv. WeakMap is not iterable.
v.

39.List down the collection of methods available on WeakMap

Below are the list of methods available on WeakMap,

i. set(key, value): Sets the value for the key in the WeakMap

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

object. Returns the WeakMap object.

ii. delete(key): Removes any value associated to the key.

iii. has(key): Returns a Boolean asserting whether a value has been associated to the key in the WeakMap object or not.

iv. get(key): Returns the value associated to the key, or undefined if there is none. Let's see the functionality of all the above methods in an example,

```
var weakMapObject =
new WeakMap();
var firstObject = {};
var secondObject = {};
// set(key, value)
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
weakMapObject.set(firs
tObject, "John");
weakMapObject.set(seco
ndObject, 100);
console.log(weakMapObj
ect.has(firstObject));
//true
console.log(weakMapObj
ect.get(firstObject));
// John
weakMapObject.delete(s
econdObject);
```

40.What is a WeakSet

WeakSet is used to store a collection of weakly(weak references) held objects. The syntax would be as follows,

```
new
WeakSet([iterable
]);
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

Let's see the below example to explain it's behavior,

```
var ws = new
WeakSet();
var user = {};
ws.add(user);
ws.has(user); //
true
ws.delete(user);
// removes user
from the set
ws.has(user); //
false, user has
been removed
```

1.41.What are the

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

differences between WeakSet and Set

The main difference is that references to objects in Set are strong while references to objects in WeakSet are weak. i.e, An object in WeakSet can be garbage collected if there is no other reference to it. Other differences are,

i. Sets can store any

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

value
Whereas WeakSets can store only collections of objects

ii. WeakSet does not have size property unlike Set
iii. WeakSet does not have methods such as clear, keys, values, entries, forEach.
iv. WeakSet is not iterable.

42.List down the collection

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

of methods available on WeakSet

Below are the list of methods available on WeakSet,

i. add(value): A new object is appended with the given value to the weakset
ii. delete(value): Deletes the value from the WeakSet collection.
iii. has(value): It returns true if the value is present in the WeakSet Collection, otherwise it returns false.

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

Let's see the functionality of all the above methods in an example,

```
var weakSetObject
= new WeakSet();
var firstObject =
{};
var secondObject
= {};
// add(value)
weakSetObject.add
(firstObject);
weakSetObject.add
(secondObject);
console.log(weakS
etObject.has(firs
tObject)); //true
weakSetObject.del
ete(secondObject)
;
```

43.How do you create an

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

object with prototype

The Object.create() method is used to create a new object with the specified prototype object and properties.
i.e, It uses an existing object as the prototype of the newly created object. It returns a new object with the specified prototype object and properties.

```
const user =
{
  name:
"John",
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
  printInfo:
function () {

console.log(`
My name is
${this.name}.
`);
  },
};

const admin =
Object.create
(user);

admin.name =
"Nick"; //
Remember that
"name" is a
property set
on "admin"
but not on
"user" object

admin.printIn
fo(); // My
name is Nick
```

44.How can you get the

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
list of keys
of any object
```

You can use the `Object.keys()` method which is used to return an array of a given object's own property names, in the same order as we get with a normal loop. For example, you can get the keys of a user object,

```
const
user = {
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
  name:
"John",

gender:
"male",
  age:
40,
};

console.
log(Obje
ct.keys(
user));
//['name
',
'gender'
, 'age']
```

45.What is the main difference betwe

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

en Object .values and Object .entries metho d

The Object.values() method's behavior is similar to Object.entries() method but it returns

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

an array of values instead [key,value] pairs.

```
const object = {

  a: "Good morning",

  b: 100,

};
```

```
for (let value of Object.values(object)) {
```

Feature	Java	JavaScript
Typed	It's a strongly typed language	It's a dynamic typed language
Paradigm	Object oriented programming	Prototype based programming
Scoping	Block scoped	Function-scoped
Concurrency	Thread based	event based
Memory	Uses more memory	Uses less memory. Hence it will be used for web pages

```
  console
.log(`${va
lue}`); //
'Good
morning'



}
```