What is EcmaScript?

It is an Scripting language standard that define the syntax, keyword, operator and other element of programming language. It is best known as standard for javaScript.

ECMA: Euopean Computer Manufacturing Association

ECMA does: Ensure the webPages work acrose different brouser

ES6 Features?

1. **Let keyword:** let varaibale is mutable, it scope is block level, redeclaration in same block in not allowed
2. **Const :** declare varable with constant, it scope is block level, redeclaration in same block in not allowed
3. **String template literals: `Hello ${variable}`**
4. **Arrow Function:** it provide us concise syntax for writing function

**Features:** **1)** No this binding: it does not have their own this. They inherit this form surrounding scope

**2)** No argument Object: they do not have their own argument object

function traditional() { console.log(arguments); // Arguments object is available }

traditional(1, 2, 3); // Output: [1, 2, 3]

const arrow = () => console.log(arguments);

arrow(1, 2, 3); // Error: arguments is not defined

**3)**  Cannot be used with Constructor: cannot be used with new Keyword

**5) Destructuring Assignment:** it allows us to extract value form array or properties form object into variables const numbers = [1, 2, 3, 4, 5]; const [a, b] = numbers; console.log(a); // Output: 1 console.log(b); // Output: 2

**6) Spread …** **Operator:** it allows us to expand element of an (iterable) array, string, object into individual element **Spread Operator (...):** Expands elements or properties. **Rest Operator (...):** Collects elements or properties into an array or object.

**7) for...of loop :The for...of loop in JavaScript is used to iterate over the values of iterable objects such as arrays, strings, Maps, Sets, and more. It provides a clean and concise way to loop through elements, making it easier to work with iterables.**

**8) Maps and Sets:**

A Map is a collection of **key-value pairs**, where keys can be of any data type (including objects, functions, and primitives). It maintains the insertion order of its elements.

**Key Features of Map:**

* Keys can be any data type.
* Maintains the order of insertion.
* Provides a better alternative to plain objects for key-value pairs when keys are not just strings.

| * **Method** | **Description** | **Example** |
| --- | --- | --- |

|  |  |  |
| --- | --- | --- |
| set(key, value) | Adds a key-value pair to the map. | map.set('name', 'Ganesh'); |

|  |  |  |
| --- | --- | --- |
| get(key) | Returns the value associated with the key. | map.get('name'); // Output: Ganesh |

|  |  |  |
| --- | --- | --- |
| has(key) | Checks if a key exists in the map. | map.has('name'); // Output: true |

|  |  |  |
| --- | --- | --- |
| delete(key) | Removes a key-value pair by key. | map.delete('name'); |

|  |  |  |
| --- | --- | --- |
| size | Returns the number of elements in the map. | map.size; // Output: 1 |

|  |  |  |
| --- | --- | --- |
| clear() | Removes all elements from the map. | map.clear(); |
|  |  |  |

**Set**

A Set is a collection of **unique values**. It does not allow duplicate elements and is often used when you need to store a list of items where uniqueness is required.

**Key Features of Set:**

* Only stores unique values.
* Can store any data type (primitives or objects).
* Does not maintain an index like arrays.
* **Methods of Set:**

| **Method** | **Description** | **Example** |
| --- | --- | --- |
| add(value) | Adds a value to the set. | set.add(10); |
| has(value) | Checks if a value exists in the set. | set.has(10); // Output: true |
| delete(value) | Removes a value from the set. | set.delete(10); |
| size | Returns the number of elements in the set. | set.size; // Output: 1 |
| clear() | Removes all elements from the set. | set.clear(); |

**8. Classes:** Classes in javascript can be used to create new Objects with the help of a constructor, each class can only have one constructor inside it.

**9) Promises:** Promises are the way to handle asynchronous operations and Manage the eventual completion or rejection of those operation

**Key States of a Promise:**

1. **Pending**: The initial state of the Promise. The operation is still in progress.
2. **Fulfilled**: The operation has completed successfully, and the Promise has a result value.
3. **Rejected**: The operation failed, and the Promise has an error or reason for the failure.

10) **Default Parameters:** Allows functions to have default values for parameters.

function greet(name = "Guest") {

return `Hello, ${name}!`;

}

console.log(greet());

1. **How many data types in JS?**

**Two types : 1) Primitive and Non Primitive**

**Primitive: Number, BigInt, Boolean, Undified, NAN, Symbol, String**

**Non Primitive: Array, Function, Object**

**6 Hosting**

**Hosting is a behaviour where variable and function declaration are moved to the top of container scope during the Compile phase but initialization is occurred .**

**7) Arrray Method**

|  |  |
| --- | --- |
| push() | Adds one or more elements to the end of the array. |

|  |  |
| --- | --- |
| pop() | Removes the last element of the array. |

|  |  |
| --- | --- |
| shift() | Removes the first element of the array. |

|  |  |
| --- | --- |
| unshift() | Adds one or more elements to the beginning of the array. |

|  |  |
| --- | --- |
| splice() | Adds/removes elements at a specified index. |

|  |  |
| --- | --- |
| sort() | Sorts the elements of the array in place. |

|  |  |
| --- | --- |
| reverse() | Reverses the order of the elements in the array. |

|  |  |
| --- | --- |
| fill() | Fills all elements of the array with a static value from start to end indices. |

|  |  |  |  |
| --- | --- | --- | --- |
| copyWithin() | | Copies part of the array to another location in the same array. | |
|  | |  | |
|  | |  | |
| **Method** | **For Iterating over array:** | |

|  |  |
| --- | --- |
| **for** | When you need full control over the iteration. |

|  |  |
| --- | --- |
| **for...of** | Simple iteration over values. |

|  |  |
| --- | --- |
| **forEach()** | When you need a clean and functional approach. |

|  |  |
| --- | --- |
| **map(**) | When you need a new array based on transformations. |

|  |  |
| --- | --- |
| **reduce(**) | When reducing the array to a single value. |

|  |  |
| --- | --- |
| **while/do...while** | When conditions govern iteration. |

|  |  |
| --- | --- |
| **entries/keys/values** | For advanced key-value or index-value iteration. |

**Sorting and Searchin methods:**

|  |  |
| --- | --- |
| indexOf() | Finds the first index of a value. |

|  |  |
| --- | --- |
| lastIndexOf() | Finds the last index of a value. |

|  |  |
| --- | --- |
| includes() | Checks if an array contains a value. |

|  |  |
| --- | --- |
| find() | Returns the first element satisfying a condition. |

|  |  |
| --- | --- |
| findIndex() | Returns the index of the first element satisfying a condition. |

|  |  |
| --- | --- |
| some() | Checks if at least one element satisfies a condition. |

|  |  |
| --- | --- |
| every() | Checks if all elements satisfy a condition. |

|  |  |
| --- | --- |
| sort() | Sorts elements (default: as strings). |

|  |  |
| --- | --- |
| reverse() | Reverses the array. |

|  |  |
| --- | --- |
| flat() | Flattens nested arrays. |

**ReactJS**

**Q. What is React?**

React is an open JavaScript library that is used for build user interface (UI) for website and web application. Basically React is used for develop the single page application with multipage user interface

**Q. Why use React?**

**1. Reusable Component:** react provide facility to us to create multiple UI components separately and integrate them with each other.

**2. Virtual DOM:** React Virtual DOM improved the performance by updating components that have changed rather than refreshing entire DOM this leads to smooth user experience

**3. SEO friendly:** React JS can help website rank higher on search engine reduce the page load time.

**4. Strong community support:** React js has large active developer community who share their knowledge and expertise

**5. Ease to learn:** React JS use the all syntax of JavaScript & ES6

**6. Power of React Native:** react native is library which help us to develop the cross platform mobile application

**Q. When to use react?**

**1. Complex user interface:** if we have web application with dynamic and interactive elements such as real time update and data visualization etc in that case we can use react

**2. High performance application:** React efficient rendering system and optimized virtual DOM allow it to handle large amount of data and complex user interface without scarifying performance

**3. Cross platform application:** using react we can develop the web application/mobile application/desktop application

1. React JS – web application

2. React Native: mobile application