javascript

What is JavaScript?

JavaScript is a high-level programming language primarily used for web development. It was created to add interactivity and dynamic behavior to websites. JavaScript is supported by all modern web browsers and allows developers to manipulate webpage content, respond to user actions, and create interactive web applications.

```
<script>

 JavaScript code

</script>
```

1. **Client-Side Scripting:** JavaScript is primarily executed on the client-side, meaning it runs within the user's web browser. It enables developers to modify and control the behavior of webpages dynamically, without requiring server interaction.
2. **Cross-Platform Compatibility:** JavaScript is supported by all major web browsers, including Chrome, Firefox, Safari, and Edge. This compatibility allows developers to create web applications that work seamlessly across different platforms and devices.
3. **Object-Oriented Programming:** JavaScript supports object-oriented programming (OOP) paradigms, allowing developers to organize their code into reusable objects and classes. This approach promotes code modularity and reusability.
4. **Event-Driven Programming:** JavaScript is event-driven, meaning it can respond to user interactions, such as mouse clicks, keyboard input, and form submissions. Developers can define event handlers to execute specific code when events occur.
5. **DOM Manipulation:** JavaScript interacts with the Document Object Model (DOM) of a webpage, which represents the structure and content of the HTML document. It enables developers to manipulate elements, change styles, modify content, and create dynamic effects on webpages.
6. **Third-Party Libraries and Frameworks:** JavaScript has a vast ecosystem of libraries and frameworks that extend its capabilities and simplify web development. Popular libraries like React, Angular, and Vue.js enable developers to build complex and interactive web applications more efficiently.
7. **Server-Side Development:** While JavaScript is primarily used on the client-side, it can also be used on the server-side through platforms like Node.js. This allows developers to build server applications using JavaScript, providing a unified language for both client and server development.

Overall, JavaScript is a versatile programming language that empowers developers to create rich and interactive web

Script enclose before body close

```
<html>
<body>
<script language="javascript" type="text/javascript">
<!--
 document.write ("Hello World!")
//-->
</script>
</body>
</html>
```

Case sensitive

Case Sensitivity JavaScript is a case-sensitive language. This means that the language keywords, variables, function names, and any other identifiers must always be typed with a consistent capitalization of letters. So the identifiers Time and TIME will convey different meanings in JavaScript

comments

```
<script language="javascript" type="text/javascript">
<!-- This is a comment –>
// This is a comment. It is similar to comments in C++
/*
This is a multiline comment in JavaScript
It is very similar to comments in C Programming
*/

</script>
```

variables

- Using var (var a=10; var b=”anu”)
- Using let (let a=10,let b=”anu”)
- Using const

  (Variables defined with `const` cannot be Redeclared.

  Variables defined with `const` cannot be Reassigned)

- Using nothing

Javascript output

JavaScript can "display" data in different ways:

- **Writing into an HTML element, using `innerHTML`.**
- **Writing into an Html element using innertext,**
- **Writing into the HTML output using `document.write()`.**
- **Writing into an alert box, using `window.alert()`.**
- **Writing into the browser console, using `console.log()`.**

DOM

1. **Accessing Elements: You can use the DOM to select and manipulate elements on a web page. For example, you can retrieve elements by their ID, class name, or tag name using methods like `getElementById`, `getElementsByClassName`, or `getElementsByTagName`.**
2. **Modifying Content: You can change the content of HTML elements using the DOM. For instance, you can update the text inside a `<p>` element by modifying its `textContent` or `innerHTML` properties.**
3. **Changing Styles: The DOM allows you to modify the style properties of elements, such as their colors, sizes, positions, etc. You can access and modify an element's style using its `style` property.**
4. **Adding and Removing Elements: You can create new elements and insert them into the document using DOM manipulation. For example, you can use methods like `createElement` and `appendChild` to add new elements to the page. Conversely, you can remove existing elements using methods like `removeChild` or `remove`.**
5. **Responding to Events: With the DOM, you can register event handlers to respond to user interactions, such as clicks, mouse movements, or form submissions. You can use methods like `addEventListener` to attach event handlers to elements and execute specific actions when events occur.**

Inner html

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My First Paragraph</p>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = 5 + 6;
</script>

</body>
</html>
```

Document.write

```
<!DOCTYPE html>
<html>
<body>
<script>
document.write("hello <br> jfghjj");
document.write("This is javascript program");
document.write(55+120);
</script>

</body>
</html>
```

Window.alert

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
window.alert(5 + 6);
</script>

</body>
</html>
```

console.log

```
<!DOCTYPE html>
<html>
<body>

<script>
console.log(5 + 6);
</script>

</body>
</html>
```

Events

- onclick :The user clicks an HTML element
- onchange : An HTML element has to be changed
- onmouseover :The user moves the mouse over an
HTML Element
- onmouseout :The user moves the mouse away from an
- HTML element
onkeyup :The user start to type

Javascript events

https://www.edureka.co/blog/javascript-events/#onclick

```
<!DOCTYPE html>
<html>
  <head>
    <title>My HTML Page</title>
  </head>
  <body>
    <h1>Hello, world!</h1>
    <button onclick="myFunction()">Click me!</button>
    <script>
      function myFunction() {
        alert("Button clicked!");
      }
    </script>
  </body>
</html>
```

Javascript OOPS concept

Class

Classes are blueprints for creating objects with shared properties and methods. In JavaScript, classes can be defined using the class keyword.

```
Class className

{

Properties;

methods;

}
```

Object

Objects: In JavaScript, an object is a collection of properties and methods that can be accessed and manipulated using dot notation or square bracket notation.

constructor and this keyword

In JavaScript, a constructor is a special method that is called when you create a new instance of a class using the `new` keyword. The purpose of a constructor is to initialize the object's properties and set up any necessary state.

```
class Person {
  constructor(name, age) {
    this.name = name;
    this.age = age;
  }
}
```

In JavaScript, `this` is a special keyword that refers to the current object.When used inside a method of an object, `this` refers to the object that the method belongs to.

Inheritance

Inheritance is the process of creating new classes that inherit properties and methods from existing classes.