Welcome Geeks

Introduction to JavaScript

- JavaScript is the world's most popular programming language.

- JavaScript can be used for Client-side developments as well as Server-side developments

https://survey.stackoverflow.co/2024/

Including JavaScript in an HTML Page

```
<script type="text/javascript">

//JS code goes here

</script>
```

Call an External JavaScript File

```
<script src="myscript.js"></script>
```

Including Comments

Single-line comments — To include a comment that is limited to a single line, precede it with //

Multi-line comments — In case you want to write longer comments between several lines, wrap it in /* and */ to avoid it from being executed

Console

DataTypes in JavaScript

Data Types: Every Variable has a data type that tells what kind of data is being stored in a variable. There are two types of data types in JavaScript namely Primitive data types and Non-primitive data types.

Primitive data types: The predefined data types provided by JavaScript language are known as primitive data types. Primitive data types are also known as in-built data types.

Non-primitive data types: The data types that are derived from primitive data types of the JavaScript language are known as non-primitive data types. It is also known as derived data types or reference data types.

Primitive DataTypes

1. Number: Number data type in javascript can be used to hold decimal values as well as values without decimals.

```
<script>
    let x = 250;
    let y = 40.5;
    console.log("Value of x=" + x);
    console.log("Value of y=" + y);
</script>
```

Primitive DataTypes

2. String: The string data type in javascript represents a sequence of characters that are surrounded by single or double quotes.

```
<script>
    let str = 'Hello All';
    let str1 = "Welcome to my new house";
    console.log("Value of str=" + str);
    console.log("Value of str1=" + str1);
</script>
```

Primitive DataTypes

3. Undefined: The meaning of undefined is ‘value is not assigned’.

```
<script>
    console.log("Value of x=" + x);
</script>
```

Primitive DataTypes

4. Boolean: The boolean data type can accept only two values i.e. true and false.

```
<script>
    console.log("value of bool=" + bool);
</script>
```

Primitive DataTypes

5. Null: This data type can hold only one possible value that is null.

```
<script>
    let x = null;
    console.log("Value of x=" + x);
</script>
```

Non Primitive DataTypes

1. Object: Object in Javascript is an entity having properties and methods. Everything is an object in javascript.

How to create an object in javascript:

Using Constructor Function to define an object:

```
// Create an empty generic object
var obj = new Object();

// Create a user defined object
var mycar = new Car();
```

Using Literal notations to define an object:

```
// An empty object
var square = {};

// Here a and b are keys and
// 20 and 30 are values
var circle = {a: 20, b: 30};
```

Non Primitive DataTypes

2. Array: With the help of an array, we can store more than one element under a single name.

Ways to declare a single dimensional array:

```
// Call it with no arguments
var a = new Array();

// Call it with single numeric argument
var b = new Array(10);

// Explicitly specify two or
// more array elements
var d = new Array(1, 2, 3, "Hello");
```

Scopes

Scope determines the accessibility of variables and functions in different parts of your code.
Types of Scopes:

Global Scope: Variables declared outside of any function or block. Accessible throughout the code.

Function Scope: Variables declared within a function. Only accessible within that function.

Block Scope: Variables declared within a block ({}). Only accessible within that block.

```
var a = "Global Scope";

function testScope() {
  var b = "Function Scope";
  if (true) {
    let c = "Block Scope";
    console.log(c); // Block Scope
  }
  console.log(b); // Function Scope
}

console.log(a); // Global Scope
testScope();
```

Functions

A JavaScript function is a block of code designed to perform a particular task.

A JavaScript function is executed when "something" invokes it (calls it).

```
function functionName(Parameter1, Parameter2, ..)
{
    // Function body
}
```

Hoisting

Hoisting is JavaScript's default behavior of moving variable and function declarations to the top of their containing scope before code execution.

Variable Hoisting:
var: Hoisted and initialized as undefined.
let and const: Hoisted but not initialized, leading to a "temporal dead zone."

Function Hoisting:
Function Declarations: Fully hoisted, meaning you can call the function before it's defined.
Function Expressions: Not hoisted; treated like variables.

```
console.log(x); // undefined (due to hoisting)
var x = 5;

console.log(y); // ReferenceError (due to temporal dead zone)
let y = 10;

testHoisting(); // Works due to hoisting
function testHoisting() {
  console.log("Hoisted Function");
}

testExpression(); // ReferenceError
var testExpression = function () {
  console.log("Not Hoisted");
};
```

var vs let vs const

var:

Hoisting: Variables declared with var are hoisted to the top of their scope and initialized as undefined.
Re-declaration: Can be re-declared and updated

let:

Hoisting: Variables declared with let are hoisted but not initialized, resulting in a "temporal dead zone" until the declaration is encountered.
Re-declaration: Cannot be re-declared within the same scope but can be updated.

var vs let vs const

const:

Hoisting: Variables declared with const are hoisted but not initialized, similar to let.
Re-declaration/Update: Cannot be re-declared or updated. Must be initialized during declaration.

JavaScript Strings

JavaScript strings are for storing and manipulating text.

JavaScript Numbers

Numbers can be written with or without decimals.

JavaScript Number Methods

- toString() (numbers to strings)
- parseInt() (variables to numbers)

JavaScript Loops

Loops can execute a block of code a number of times.

```
for (let i = 0; i < 5; i++) {
  console.log("The number is " + i);
}
```

JavaScript Arrays

An array is a special variable, which can hold more than one value:

JavaScript Objects

Objects are variables too. But objects can contain many values.

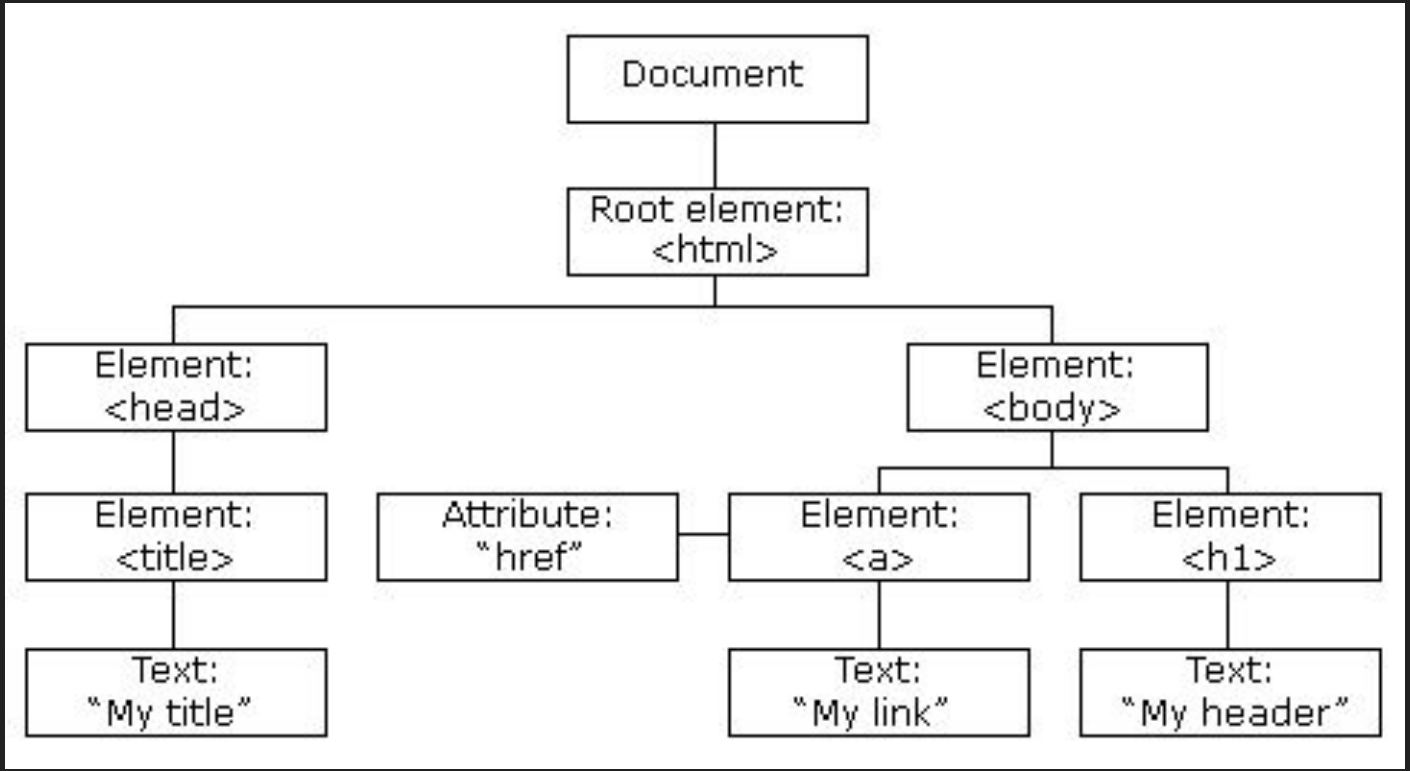
JavaScript

https://www.learn-js.org/

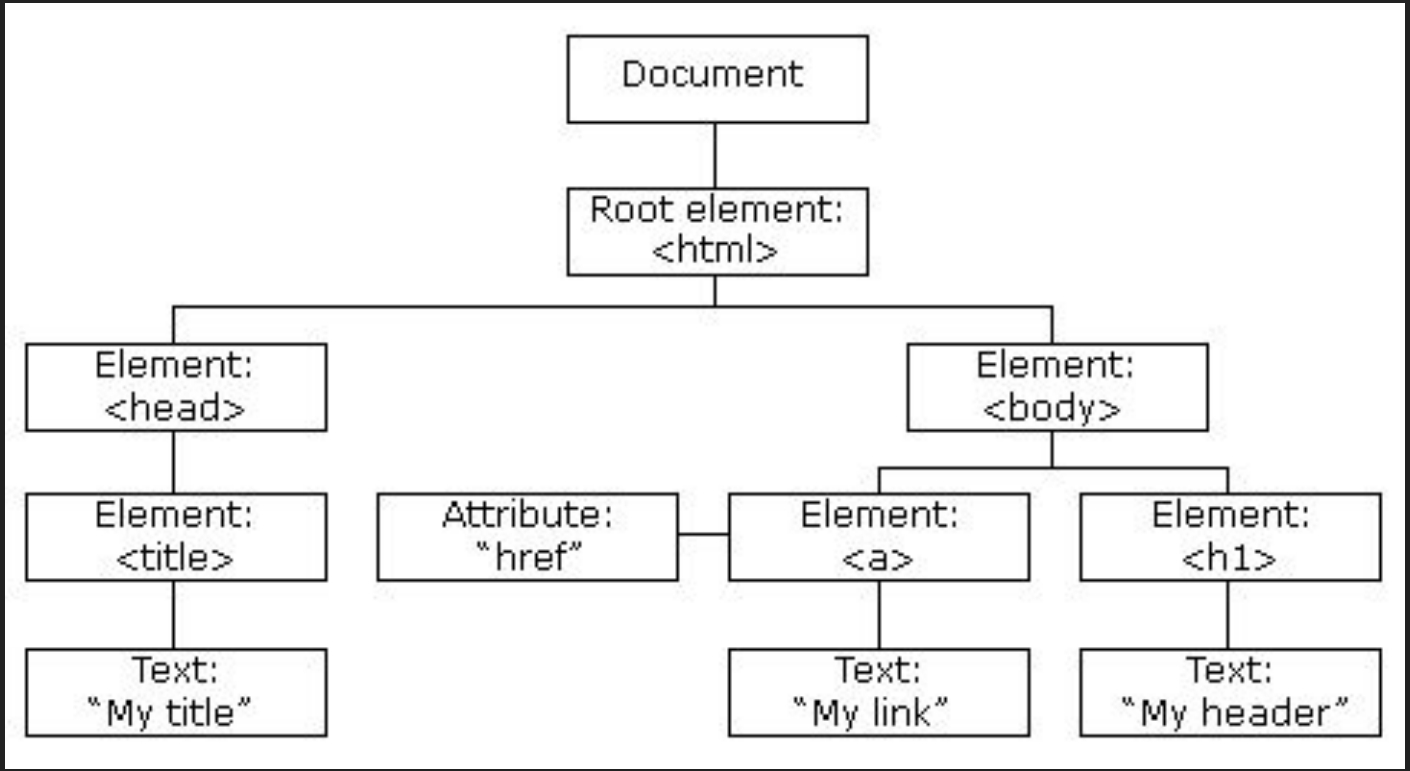
DOM

- DOCUMENT OBJECT MODEL
- Structural representation of the HTML Document
- JavaScript can be used to read/write/manipulate the DOM

DOM


Document
Root element:
<html>
Element:
<head>
Element:
<body>
Element:
<title>
Attribute:
"href"
Element:
<a>
Element:
<h1>
Text:
"My title"
Text:
"My link"
Text:
"My header"

DOM Manipulation????



Document Object

When an HTML document is loaded into a web browser, it becomes a document object.

The document object is the root node of the HTML document.

The document object is a property of the window object.

Single Element Selectors

document.getElementById();
document.querySelector();

Multi Element Selectors

document.getElementsByClassName();
document.querySelectorAll();

Events

HTML DOM allows JavaScript to react to HTML events.

A JavaScript can be executed when an event occurs, like when a user clicks on an HTML element.

Number Guessing Game using JS