JAVASCRIPT BASIC

Comments

Single Line Comments

```
//
```

Multi-line Comments

```
 /*                                          */
```

Variables

Variables are used to store data, like string of text, numbers, etc. The data or value stored in the variables can be set, updated, and retrieved whenever needed. In general, variables are symbolic names for values.

```
<script>

    // Creating variables

    var name = "Ram";
    var age = 5;

    // Printing variable values
    document.write(name + "<br>");
    document.write(age + "<br>");
    </script>
```

let and const Keywords

let and const for declaring variables.

```
<script>
    // Declaring variables
    let name = "One";
    let age = 54;

    document.write(name + "<br>");
    document.write(age + "<br>");

    const PI = 3.14;

    document.write(PI); // 3.14
    </script>
```

rules for naming a JavaScript variable:

- A variable name must start with a letter, underscore (_), or dollar sign (`$`).
- A variable name cannot start with a number.
- A variable name can only contain alpha-numeric characters (`A-z`, `0-9`) and underscores.
- A variable name cannot contain spaces

Alert :

```
 <script>
    alert("Hello World!");
    var x = 10;
    var y = 20;
    var sum = x + y;
    alert(sum);
    </script>
```

Comparisons(Boolean Data Type)

```
<script>
    var a = 2, b = 5, c = 10;
    document.write(b > a)
    document.write("<br>");
    document.write(b > c)
    </script>
```

Array Data Type

An array is a type of object used for storing multiple values in single variable. Each value (also called an element) in an array has a numeric position,

```
 <script>
    var colors = ["Red", "Yellow", "Green", "Orange"];
    var cities = ["4", "2", "5"];
    document.write(colors[0] + "<br>");
    document.write(cities[2]);
    </script>
```

Length of a String

```
<script>
    var str1 = "This is a paragraph of text.";
    document.write(str1.length + "<br>");

    var str2 = "This is a \n paragraph of text.";
    document.write(str2.length);
    </script>
(2)
<script>
    var myString = 'Hello World!';
    var myString = "Hello World!";
    document.write(myString + "<br>");
    document.write(myString);
    </script>
```

Function Data Type

Functions can be stored in variables, objects, and arrays. Functions can be passed as arguments to other functions, and functions can be returned from functions.

```
<script>
    var greeting = function(){
        return "Hello World!";
    }

    document.write(typeof greeting)
    document.write("<br>");
    document.write(greeting());
    </script>
```

Function

```
 <script>
    function createGreeting(name){
        return "Hello, " + name;
    }
    function displayGreeting(greetingFunction, userName){
        return greetingFunction(userName);
    }
    var result = displayGreeting(createGreeting, "Peter");
    document.write(result);
    </script>
```

typeof Operator

```
<script>
    document.write(typeof 15 + "<br>");
    document.write(typeof 42.7 + "<br>");
    document.write(typeof '' + "<br>");
    document.write(typeof 'hello' + "<br>");
    document.write(typeof '12' + "<br>");
    document.write(typeof true + "<br>");
    document.write(typeof false + "<br>");
    document.write(typeof {name: "John", age: 18} + "<br>");
    document.write(typeof [1, 2, 4] + "<br>");
    document.write(typeof function(){});
    </script>
```

Operators

Arithmetic Operators

```
    <script>
    var x = 10;
    var y = 4;
    document.write(x + y);
    document.write("<br>");

    document.write(x - y);
    document.write("<br>");

    document.write(x * y);
    document.write("<br>");

    document.write(x / y);
    document.write("<br>");

    document.write(x % y);
    </script>
```

Assignment Operators

```
<script>
    var x;

    x = 10;
    document.write(x + "<br>");

    x = 20;
    x += 30;
    document.write(x + "<br>");

    x = 50;
    x -= 20;
    document.write(x + "<br>");

    x = 5;
    x *= 25;
    document.write(x + "<br>");

    x = 50;
    x /= 10;
    document.write(x + "<br>");

    x = 100;
    x %= 15;
    document.write(x);
    </script>
```

String Operators

```
<script>
    var str1 = "Hello";
    var str2 = " World!";

    document.write(str1 + str2 + "<br>");

    str1 += str2;
    document.write(str1);
    </script>
```

Incrementing and Decrementing Operators

Operator	Name	Effect
`++x`	Pre-increment	Increments x by one, then returns x
`x++`	Post-increment	Returns x, then increments x by one
`--x`	Pre-decrement	Decrements x by one, then returns x
`x--`	Post-decrement	Returns x, then decrements x by one

```
<script>

    var x;

    x = 10;

    document.write(++x);

    x = 10;

    document.write(x++);

    x = 10;

    document.write(--x);

    x = 10;

    document.write(x--);

    </script>
```

Logical Operators

The logical operators are typically used to combine conditional statements.

Operator	Name	Example	Result
`&&`	And	`x && y`	True if both x and y are true
`||`	Or	`x || y`	True if either x or y is true
`!`	Not	`!x`	True if x is not true

```
<script>

    var year = 2018;

    // Leap years are divisible by 400 or by 4 but not 100

    if((year % 400 == 0) || ((year % 100 != 0) && (year % 4 ==
0))){

        document.write(year + " is a leap year.");
    } else{
        document.write(year + " is not a leap year.");
    }
    </script>
```

Comparison Operators

Operator	Name	Example	Result
`==`	Equal	`x == y`	True if x is equal to y
`===`	Identical	`x === y`	True if x is equal to y, and they are of the same type
`!=`	Not equal	`x != y`	True if x is not equal to y
`!==`	Not identical	`x !== y`	True if x is not equal to y, or they are not of the same type
`<`	Less than	`x < y`	True if x is less than y
`>`	Greater than	`x > y`	True if x is greater than y
`>=`	Greater than or equal to	`x >= y`	True if x is greater than or equal to y
`<=`	Less than or equal to	`x <= y`	True if x is less than or equal to y

```
<script>
    var x = 25;
    var y = 35;
    var z = "25";

    document.write(x == z);
    document.write("<br>");

    document.write(x === z);
    document.write("<br>");

    document.write(x != y);
    document.write("<br>");

    document.write(x !== z);
    document.write("<br>");
    document.write(x < y);
    document.write("<br>");

    document.write(x > y);
    document.write("<br>");

    document.write(x <= y);
    document.write("<br>");

    document.write(x >= y);
    </script>
```

Events

event is something that happens when user interact with the web page, such as when he clicked a link or button, entered text into an input box or textarea, made selection in a select box, pressed key on the keyboard, moved the mouse pointer, submits a form, etc. In some cases, the Browser itself can trigger the events, such as the page load and unload events.the names for event handlers always begin with the word "on", so an event handler for the click event is called **onclick**, similarly an event handler for the load event is called **onload**, event handler for the blur event is called **onblur.**

```
 <button type="button" onclick="alert('Hello World!')">Click
Me</button>
```

(2)

```
 <button type="button" id="myBtn">Click Me</button>
    <script>
        function sayHello(){
            alert('Hello World!');
        }
        document.getElementById("myBtn").onclick = sayHello;
    </script>
```

Mouse Events

The Click Event (onclick)

```
<button type="button" onclick="alert('You have clicked a button!');">Click Me</button>
    <a href="#" onclick="alert('You have clicked a link!');">Click Me</a>
```

The Mouseover Event (onmouseover)

```
<button type="button" onmouseover="alert('You have placed mouse pointer over a button!');">Place Mouse Over Me</button>
    <a href="#" onmouseover="alert('You have placed mouse pointer over a link!');">Place Mouse Over Me</a>
```

The Keydown Event (onkeydown)

```
<input type="text" onkeydown="alert('You have pressed a key inside text input!')">
    <hr>
    <textarea cols="30" onkeydown="alert('You have pressed a key inside textarea!')"></textarea>
```

The Focus Event (onfocus)

```
    <script>
        function highlightInput(elm){
            elm.style.background = "yellow";
        }
    </script>
    <input type="text" onfocus="highlightInput(this)">
    <button type="button">Button</button>
```

The Change Event (onchange)

```
    <select onchange="alert('You have changed the selection!');">
        <option>Select</option>
        <option>Male</option>
        <option>Female</option>
    </select>
```

The Submit Event (onsubmit)

```
<form action="" method="post" onsubmit="alert('Form data will be
submitted to the server!');">
        <label>First Name:</label>
        <input type="text" name="first-name" required>
        <input type="submit" value="Submit">
    </form>
```

Document/Window Events

The Load Event (onload)

```
<body onload="window.alert('Page is loaded successfully!');">
    <h1>This is a heading</h1>
```

Finding a String Inside Another String

You can use the `indexOf()` method to find a substring or string within another string. This method returns the index or position of the first occurrence of a specified string within a string.

```
<script>

    var str = "If the facts don't fit the theory, change the
facts.";

    var pos = str.indexOf("facts");

    document.write(pos);

    </script>
```

you can use the lastIndexOf() method to get the index or position of the last occurrence of the specified string within a string,

```
 <script>

    var str = "If the facts don't fit the theory, change the
facts.";

    var pos = str.lastIndexOf("facts");

    document.write(pos); // Outputs: 46

    </script>
```

Extracting a Substring from a String

You can use the `slice()` method to extract a part or substring from a string.

This method takes 2 parameters: *start index* (index at which to begin extraction), and an optional *end index* (index before which to end extraction), like `str.slice(startIndex, endIndex)`.

```
<script>

    var str = "The quick brown fox jumps over the lazy dog.";

    var subStr = str.slice(4, 15);

    document.write(subStr);

    </script>
```

(2)

```
 <script>
    var str = "The quick brown fox jumps over the lazy dog.";
    document.write(str.slice(-28, -19) + "<br>"); // Prints: fox
jumps
    document.write(str.slice(31)); // Prints: the lazy dog.
    </script>
```

Replacing the Contents of a String

```
  <script>
    var str = "Color red looks brighter than color blue.";
    var result = str.replace("color", "paint");
    document.write(result);
    </script>
```

Converting a String to Uppercase or Lowercase

You can concatenate or combine two or more strings using the + and += assignment operators.

```
<script>
    var str = "Hello World!";
    var result = str.toUpperCase();
    document.write(result); // Prints: HELLO WORLD!
    </script>
```

```
  <script>
    var str = "Hello World!";
    var result = str.toLowerCase();
    document.write(result); // Prints: hello world!
    </script>
```

Concatenating Two or More Strings

You can concatenate or combine two or more strings using the + and += assignment operators.

```
  <script>
    var hello = "Hello";
    var world = "World";
    var greet = hello + " " + world;
    document.write(greet + "<br>"); // Prints: Hello World

    var wish  = "Happy";
        wish += " New Year";
    document.write(wish); // Prints: Happy New Year
    </script>
```

Accessing Individual Characters from a String

You can use the `charAt()` method to access individual character from a string, like `str.charAt(index)`. The `index` specified should be an integer between `0` and `str.length - 1`. If no index is provided the first character in the string is returned, since the default is `0`.

```
    <script>
    var str = "Hello World!";
    document.write(str.charAt() + "<br>");
    document.write(str.charAt(6) + "<br>");
    document.write(str.charAt(30) + "<br>");
    document.write(str.charAt(str.length - 1));
    </script>
```

Splitting a String into an Array

The `split()` method can be used to splits a string into an array of strings, using the syntax `str.split(separator, limit)`. The `seperator` argument specifies the string at which each split should occur, whereas the `limit` arguments specifies the maximum length of the array.

```
<script>
    var fruitsStr = "Apple, Banana, Mango, Orange, Papaya";
    var fruitsArr = fruitsStr.split(", ");
    document.write(fruitsArr[0] + "<br>");
    document.write(fruitsArr[2] + "<br>");
    document.write(fruitsArr[fruitsArr.length - 1]);
    document.write("<hr>");

    // Loop through all the elements of the fruits array
    for(var i in fruitsArr) {
        document.write("<p>" + fruitsArr[i] + "</p>");
    }
    </script>
```