**JavaScript**

**JAVASCRIPT LANGUAGE**

JavaScript (js) is a light-weight object-oriented programming language which is used by several websites for scripting the webpages. It is an interpreted, full-fledged programming language that enables dynamic interactivity on websites when applied to an HTML document. It was introduced in the year 1995 for adding programs to the webpages in the Netscape Navigator browser. Since then, it has been adopted by all other graphical web browsers. With JavaScript, users can build modern web applications to interact directly without reloading the page every time. The traditional website uses js to provide several forms of interactivity and simplicity.

**Features of JavaScript:**

1. All popular web browsers support JavaScript as they provide built-in execution environments.
2. JavaScript follows the syntax and structure of the C programming language. Thus, it is a structured programming language.
3. JavaScript is a weakly typed language, where certain types are implicitly cast (depending on the operation).
4. JavaScript is an object-oriented programming language that uses prototypes rather than using classes for inheritance.
5. It is a light-weighted and interpreted language.
6. It is a case-sensitive language.
7. JavaScript is supportable in several operating systems including, Windows, macOS, etc.
8. It provides good control to the users over the web browsers.

**Application of JavaScript**

* Client-side validation,
* Dynamic drop-down menus,
* Displaying date and time,
* Displaying pop-up windows and dialog boxes (like an alert dialog box, confirm dialog box and prompt dialog box),
* Displaying clocks etc.

**Example:**

**<script>**

document.write("Hello JavaScript by JavaScript");

**</script>**

**JavaScript Data Types:**

JavaScript provides different data types to hold different types of values. There are two types of data types in JavaScript.

1. Primitive data type
2. Non-primitive (reference) data type

**primitive data types**

|  |  |
| --- | --- |
| Data Type | Description |
| String | represents sequence of characters e.g. "hello" |
| Number | represents numeric values e.g. 100 |
| Boolean | represents boolean value either false or true |
| Undefined | represents undefined value |
| Null | represents null i.e. no value at all |

non-primitive data types

|  |  |
| --- | --- |
| Data Type | Description |
| Object | represents instance through which we can access members |
| Array | represents group of similar values |
| RegExp | represents regular expression |

**JavaScript Operators**:

JavaScript operators are symbols that are used to perform operations on operands.

1. ithmetic Operators
2. Comparison (Relational) Operators
3. Bitwise Operators
4. Logical Operators
5. Assignment Operators
6. Special Operators

**JavaScript Arithmetic Operators**

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Example** |
| + | Addition | 10+20 = 30 |
| - | Subtraction | 20-10 = 10 |
| \* | Multiplication | 10\*20 = 200 |
| / | Division | 20/10 = 2 |
| % | Modulus (Remainder) | 20%10 = 0 |
| ++ | Increment | var a=10; a++; Now a = 11 |
| -- | Decrement | var a=10; a--; Now a = 9 |

**If-else:**

The JavaScript if-else statement is used to execute the code whether condition is true or false. There are three forms of if statement in JavaScript.

1. If Statement
2. If else statement
3. if else if statement

**If statement**

It evaluates the content only if expression is true.

**Syntax**:

if(expression){

//content to be evaluated

}

**Example:**

**<script>**

var a=20;

if(a**>**10){

document.write("value of a is greater than 10");

}

**</script>**

**If...else Statement**

It evaluates the content whether condition is true of false.

Syntax:

if(expression){

//content to be evaluated if condition is true

}

else{

//content to be evaluated if condition is false

}

**Example**:

**<script>**

var a=20;

if(a%2==0){

document.write("a is even number");

}else{

document.write("a is odd number");

}

**</script>**

**If...else if statement**

It evaluates the content only if expression is true from several expressions.

Syntax:

if(expression1){

//content to be evaluated if expression1 is true

}  else if(expression2){

//content to be evaluated if expression2 is true

}  else if(expression3){

//content to be evaluated if expression3 is true

}else{

//content to be evaluated if no expression is true

}

Example:

**<script>**

var a=20;

if(a==10){

document.write("a is equal to 10");

}

else if(a==15){

document.write("a is equal to 15");

}

else if(a==20){

document.write("a is equal to 20");

}

else{

document.write("a is not equal to 10, 15 or 20");

}

**</script>**

**JavaScript Loops:**

The JavaScript loops are used to iterate the piece of code using for, while, do while or for-in loops.

1. r loop
2. while loop
3. do-while loop
4. for-in loo

**For loop:**

*The****JavaScript for loop***iterates the elements for the fixed number of times*.*

Syntax:

for (initialization; condition; increment)

{

    code to be executed

}

Example:

<script>

for (i=1; i<=5; i++)

{

document.write(i + "<br/>")

}

</script>

**while loop:**

The**JavaScript while loop**iterates the elements for the infinite number of times.

Syntax:

while (condition)

{

    code to be executed  }

Example:

**<script>**

var i=11;

while (i**<**=15)

{

document.write(i + "**<br/>**");

i++;

}

**</script>**

**Functions**:

JavaScript functions are used to perform operations.

There are mainly two advantages of JavaScript functions.

* 1. Code reusability: We can call a function several times so it save coding.
  2. Less coding: It makes our program compact. We don’t need to write many lines of code each time to perform a common task.

Syntax:

function functionName([arg1, arg2, ...argN]){

 //code to be executed

}

Example:

**<script>**

function msg(){

alert("hello! this is message");

}

**</script>**

**<input** type="button" onclick="msg()" value="call function"**/>**

**JavaScript Object**

A javaScript object is an entity having state and behavior (properties and method). For example: car, pen, bike, chair, glass, keyboard, monitor etc.

JavaScript is an object-based language. Everything is an object in JavaScript.

JavaScript is template based not class based. Here, we don't create class to get the object. But, we direct create objects.

**Creating Object in JavaScript:**

There are Three type of object.

1. By object literal
2. By creating instance of Object directly (using new keyword)
3. By using an object constructor (using new keyword)
4. **JavaScript Object by Object literal:**

The syntax of creating object using object literal is given below:

**Object = {‘key1’:’value1’,’key2’:value2,’key3’:’value3’, …………}**

As you can see, property and value is separated by: (colon).

**Example:**

**<script>**

emp={id:102,name:"Shyam Kumar",salary:40000}

document.write(emp.id+" "+emp.name+" "+emp.salary);

**</script>**

1. **By creating instance of Object:**

The syntax of creating object directly is given below:

**Var objectname=new object()**

Here, **new keyword** is used to create object.

**Example:**

**<script>**

var emp=new Object();

emp.id=101;

emp.name="GNDU";

emp.salary=50000;

document.write(emp.id+" "+emp.name+" "+emp.salary);

**</script>**

## By using an Object constructor:

## you need to create function with arguments. Each argument value can be assigned in the current object by using this keyword.

## The this keyword refers to the current object

## Example:

**<script>**

function emp(id,name,salary){

this.id=id;

this.name=name;

this.salary=salary;

}

e=new emp(103,"GNDU ",30000);

document.write(e.id+" "+e.name+" "+e.salary);

**</script>**

**Host Object**

Host objects are the objects which are created and provided to JavaScript by the environment. which hosts the code. In Node environment or in browser, the JavaScript code will behave differently when executed in different JavaScript engines. Since JavaScript always works in the context of the hosting environment, the result of executed code may differ depending on the current host.

One famous host object which developers interact regularly is - **console** object. It has several methods like .**log(), .error(), .table(), dir(), info(), profile()** etc. console object is provided by hosting environment so that JavaScript code can interact with the host for development related job. It may perform differently depending on the hosting environment which supplies the console object.

**Example:**

<script>

var a=True;

if (a){

console.log(‘GNDU’);

}else{

console.log(“GNDU”)

}

</script>

**JavaScript in your browser**

JavaScript is enabled in your web browser.

**Google Chrome:**

1. on browser menu click on the "Customize and control Google Chrome" and select "Settings".
2. In the "Settings" section click on the "Show advanced settings..."
3. Under the "Privacy" click on the "Content settings...".
4. When the dialog window opens, look for the "JavaScript" section and select "Allow all sites to run JavaScript (recommended)".
5. Click on the "OK" button to close it.
6. Close the "Settings" tab.
7. Click on the "Reload this page" button of the web browser to refresh the page.

**Internet Explorer:**

1. In Internet Explorer menu click "Tools" icon and select "Internet Options".
2. In the "Internet Options" window select the "Security" tab.
3. On the "Security" tab click on the "Custom level..." button.
4. When the "Security Settings - Internet Zone" dialog window opens, look for the "Scripting" section.
5. In the "Active Scripting" item select "Enable".
6. When the "Warning!" window pops out asking "Are you sure you want to change the settings for this zone?" select "Yes".
7. In the "Internet Options" window click on the "OK" button to close it.
8. Click on the "Refresh" button of the web browser to refresh the page.

**Mozilla Firefox:**

1. the address bar, type about: config and press Enter.
2. Click "I'll be careful, I promise" if a warning message appears.
3. In the search box, search for JavaScript enabled
4. Toggle the "JavaScript enabled" preference (right-click and select "Toggle" or double-click the preference) to change the value from "false" to "true".
5. Click on the "Reload current page" button of the web browser to refresh the page.

**Document Object Model**

The document object represents the whole html document. When html document is loaded in the browser, it becomes a document object. It is the root element that represents the html document. It has properties and methods. By the help of document object, we can add dynamic content to our web page.

The HTML DOM can be accessed with JavaScript (and with other programming languages).

Example:

<html>  
<body>

<p id="demo"></p>  
  
<script>

document.getElementById("demo").innerHTML = "Hello World!";  
</script>  
</body>  
</html>

**getElementById:**

The most common way to access an HTML element is to use the id of the element.

In the example above the getElementById method used id="demo" to find the element.

**Finding HTML Elements:**

with JavaScript, you want to manipulate HTML elements. There are several ways to do this:

1. Finding HTML elements by id
2. Finding HTML elements by tag name
3. Finding HTML elements by class name
4. Finding HTML elements by CSS selectors
5. **By Id:**

The easiest way to find an HTML element in the DOM, is by using the element id. This example finds the element with id="intro":

**Example:**

<!DOCTYPE html>

<html>

<body>

<h2>Finding HTML Elements by Id</h2>

<p id="intro">Hello World!</p>

<p>This example demonstrates the <b>getElementsById</b> method.</p>

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

<script>

var myElement = document.getElementById("intro");

document.getElementById("demo").innerHTML = "The text from the intro paragraph is " + myElement.innerHTML;

</script>

</body>

</html>

1. **Tag Name:**

<!DOCTYPE html>

<html>

<body>

<h2>Finding HTML Elements by Tag Name</h2>

<p>Hello World!</p>

<p>This example demonstrates the <b>getElementsByTagName</b> method.</p>

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

<script>

var x = document.getElementsByTagName("p");

document.getElementById("demo").innerHTML = 'The text in first paragraph (index 0) is: ' + x[0].innerHTML;

</script>

</body>

</html>

1. **by Class Name:**

If you want to find all HTML elements with the same class name, use getElementsByClassName().

**Example:**

<!DOCTYPE html>

<html>

<body>

<h2>Finding HTML Elements by Class Name</h2>

<p>Hello World!</p>

<p class="intro">The DOM is very useful.</p>

<p class="intro">This example demonstrates the <b>getElementsByClassName</b> method.</p>

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

<script>

var x = document.getElementsByClassName("intro");

document.getElementById("demo").innerHTML = 'The first paragraph (index 0) with class="intro": ' + x[0].innerHTML;

</script>

</body>

</html>

1. **by CSS Selectors:**

If you want to find all HTML elements that match a specified CSS selector (id, class names, types, attributes, values of attributes, etc), use the querySelectorAll() method

**Example:**

<!DOCTYPE html>

<html>

<body>

<h2>Finding HTML Elements by Query Selector</h2>

<p>Hello World!</p>

<p class="intro">The DOM is very useful.</p>

<p class="intro">This example demonstrates the <b>querySelectorAll</b> method.</p>

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

<script>

var x = document.querySelectorAll("p.intro");

document.getElementById("demo").innerHTML = 'The first paragraph (index 0) with class="intro": ' + x[0].innerHTML;

</script>

</body>

</html>