1. **JAVASCRIPT**

**VARIABLE, OPERATOR, CONDITIONAL AND LOOPING STATEMENTS, FUNCTIONS**

Programming language of the web

Applications:

Create front end frameworks Angular,React

Back end piece of web application

Desktop application

Mobile apps

Ai and machine learning

JavaScript was invented by Brendan Eich in 1995.

It was developed for Netscape 2, and became the ECMA-262 standard in 1997.

After Netscape handed JavaScript over to ECMA, the Mozilla foundation continued to develop JavaScript for the Firefox browser.

**Variables:**

container for storing data

var x=5;

var y=6;

var z=x+y;

var x;//variable declaration

var x=2;//variable initialization

to declare a variable we can use let, var and const.

**const** ( will not able to change a value)

const pi=3.14;

**var** (can able to redeclare a variable)

**let** (cannot redeclare a variable)

**Data types:**

* String
* Number
* Boolean
* undefined
* null
* array
* object

**operators:**

There are different types of JavaScript operators:

Arithmetic Operators [+, -, \*, \*\*, /, %, ++, --]de­crement

Assignment Operators [=, +=, -=, \*=, /=, %=, \*\*=]

Comparison Operators [==, !=, ===, !==, <, >, <=, >=, ?]

Logical Operators [&&, ||, !]

Type Operators [typeof, instanceof]

**conditional and looping statements:**

Conditional statements are used to perform different actions based on different conditions.

if statement:

syntax:

**if(condition)**

**{//code to be executed**

**}**

if else statement:

**if(condition)**

**{//code**

**}**

**else**

**{//code**

**}**

else if statement:

**if(condition)**

**{//code**

**}**

**else if(condition)**

**{//code**

**}**

**else{**

**//code**

**}**

SWITCH statement:

The switch statement is used to perform different actions based on different conditions.

syntax:

switch(expression) {

case x:

// code block

break;

case y:

// code block

break;

default:

// code block

}

* The switch expression is evaluated once.
* The value of the expression is compared with the values of each case.
* If there is a match, the associated block of code is executed.
* If there is no match, the default code block is executed.

**Looping: (for, while, do-while)**

for loop:

for (expression 1; expression 2; expression 3) {

// code block to be executed

}

while loop:

The while loop loops through a block of code as long as a specified condition is true.

while (condition) {

// code block to be executed

}

do- while loop:

The do while loop is a variant of the while loop. This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.

do {

// code block to be executed

}

while (condition);

Strings:

// Using double quotes:

var carName1 = "Volvo XC60";

// Using single quotes:

var carName2 = 'Volvo XC60';

Numbers:

// With decimals:

var x1 = 34.00;

// Without decimals:

var x2 = 34;

Boolean:

var a=true;

var b=false;

Arrays:

var name=[“anu”,”banu”,”charu”];

objects:

var person={firstname=”john”, lastname=”elice” ,age=50, color=”white”};

Undefined:

var x;//undefined

var x=5;//number

var x=”name”;//string

typeof:

typeof 50;//number

typeof “name”;//string

typeof “”//string

Function:

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).

* A JavaScript function is defined with the function keyword, followed by a name, followed by parentheses ().
* Function names can contain letters, digits, underscores, and dollar signs (same rules as variables).
* The parentheses may include parameter names separated by commas:
* (parameter1, parameter2, ...)
* The code to be executed, by the function, is placed inside curly brackets: {}

syntax:

function name(parameter1, parameter2, parameter3) {

// code to be executed

}

functionname(value1,value2,value3);

Example:

<!DOCTYPE html>

<html>

<body>

<h2>JavaScript Functions</h2>

<p>This example calls a function which performs a calculation, and returns the result:</p>

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

<script>

**function myFunction(p1, p2) {**

**return p1 \* p2;**

**}**

document.getElementById("demo").innerHTML = **myFunction(4, 3);**

</script>

</body>

</html>

**Examples:**

**roll,name,3 marks,average.**

**eligible for vote?**

**2 number equal?**

**to print natural numbers**

**to print even numbers**

**to fine cube of each number**

**to find sum of given natural number**

**calculator for function**

1. **JS ESSENTIALS, VAR, LET, CONST, ARROW FUNCTION, DEFAULT ARGUMENTS, TEMPLATE STRINGS, STRING METHODS**

To declare javascript :

var x=9;

let y=2;

const pi=3.14;

**Arrow functions:**

**let x;**

x=(val)=>val+”world”;

x(“hello”); //function call

**Default Arguments:**

If y is not passed or undefined, then y = 10.

function myFunction(x, **y = 10**) {

return x + y;

}

myFunction(5);

**Template strings:**

let string1=”hello world”;

let string2=’hello world’;

let string3=` he’s often called “johny” `;

let firstname=”john”;

let lastname=”marry”;

let name=`welcome ${firstname} ${lastname}`;

**String methods:**

**String length**

The length property returns the length of a string:

let text = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

let length = text.length;

**There are 3 methods for extracting a part of a string:**

* **slice(start, end)**
* **substring(start, end)**
* **substr(start, length)**

**String slice()**

slice() extracts a part of a string and returns the extracted part in a new string.

The method takes 2 parameters: start position, and end position (end not included).

let text = "Apple, Banana, Kiwi";

let part = text.slice(7, 13); //Banana

let text = "Apple, Banana, Kiwi";

let part = text.slice(-12);//not work

**String substring()**

substring() is similar to slice().

The difference is that start and end values less than 0 are treated as 0 in substring().

let str = "Apple, Banana, Kiwi";

let part = str.substring(7, 13);

**String substr()**

substr() is similar to slice().

The difference is that the second parameter specifies the length of the extracted part.

let str = "Apple, Banana, Kiwi";

let part = str.substr(7, 6);

**String replace()**

The replace() method replaces a specified value with another value in a string:

let text = "Please visit Microsoft!";

let newText = text.replace("Microsoft", "W3Schools");

**String replaceAll()**

let text= "I love cats. Cats are very easy to love. Cats are very popular."

text = text.replaceAll("Cats","Dogs");

text = text.replaceAll("cats","dogs");

**String toUpperCase()**

let text1 = "Hello World!";

let text2 = text1.toUpperCase();

**String toLowerCase()**

let text1 = "Hello World!"; // String

let text2 = text1.toLowerCase();

**String concat()**

concat() joins two or more strings:

let text1 = "Hello";

let text2 = "World";

let text3 = text1.concat(" ", text2);

**String trim()**

The trim() method removes whitespace from both sides of a string:

Example

let text1 = " Hello World! ";

let text2 = text1.trim();

**String trimStart()**

The trimStart() method works like trim(), but removes whitespace only from the start of a string.

Example

let text1 = " Hello World! ";

let text2 = text1.trimStart();

**String trimEnd()**

The trimEnd() method works like trim(), but removes whitespace only from the end of a string.

Example

let text1 = " Hello World! ";

let text2 = text1.trimEnd();

**String charAt()**

The charAt() method returns the character at a specified index (position) in a string:

Example

let text = "HELLO WORLD";

let char = text.charAt(0);

**String charCodeAt()**

The charCodeAt() method returns the unicode of the character at a specified index in a string:

The method returns a UTF-16 code (an integer between 0 and 65535).

Example

let text = "HELLO WORLD";

let char = text.charCodeAt(0);

**String split()**

A string can be converted to an array with the split() method:

Example

text.split(",") // Split on commas

text.split(" ") // Split on spaces

text.split("|") // Split on pipe

let text = "a,b,c,d,e,f";

const myArray = text.split(",");

**String indexOf()**

The indexOf() method returns the index of (position of) the first occurrence of a string in a string:

Example

let str = "Please locate where 'locate' occurs!";

str.indexOf("locate");

**String lastIndexOf()**

The lastIndexOf() method returns the index of the last occurrence of a specified text in a string:

Example

let text = "Please locate where 'locate' occurs!";

text.lastIndexOf("locate");

Both indexOf(), and lastIndexOf() return -1 if the text is not found:

Both methods accept a second parameter as the starting position for the search:

Example

let text = "Please locate where 'locate' occurs!";

text.indexOf("locate", 15);

**String search()**

The search() method searches a string for a string (or a regular expression) and returns the position of the match:

Examples

let str = "Please locate where 'locate' occurs!";

str.search("locate");

The two methods, indexOf() and search(), are equal?

They accept the same arguments (parameters), and return the same value?

The two methods are NOT equal. These are the differences:

The search() method cannot take a second start position argument.

The indexOf() method cannot take powerful search values (regular expressions).

**String match()**

The match() method returns an array containing the results of matching a string against a string (or a regular expression).

Examples

Perform a search for "ain":

let text = "The rain in SPAIN stays mainly in the plain";

text.match("ain"); //1

Perform a global, case-insensitive search for "ain":

let text = "The rain in SPAIN stays mainly in the plain";

text.match(/ain/gi); // 4

If a regular expression does not include the g modifier (global search), match() will return only the first match in the string.

**String matchAll()**

**String includes()**

The includes() method returns true if a string contains a specified value.

Otherwise it returns false.

Examples

Check if a string includes "world":

let text = "Hello world, welcome to the universe.";

text.includes("world");//true

Check if a string includes "world". Start at position 12:

let text = "Hello world, welcome to the universe.";

text.includes("world", 12);//false

**String startsWith()**

The startsWith() method returns true if a string begins with a specified value.

Otherwise it returns false:

Examples

Returns true:

let text = "Hello world, welcome to the universe.";

text.startsWith("Hello");//true

A start position for the search can be specified:

Returns false:

let text = "Hello world, welcome to the universe.";

text.startsWith("world", 6)//true

**String endsWith()**

The endsWith() method returns true if a string ends with a specified value.

Otherwise it returns false:

Examples

Check if a string ends with "Doe":

let text = "John Doe";

text.endsWith("Doe");

Check if the 11 first characters of a string ends with "world":

let text = "Hello world, welcome to the universe.";

text.endsWith("world", 11);