V-Ex Tech

V-EX TECH

Web Development

Java / Node.js / PHP / .Net / Python

Certification Course

Assured Placement Program

With International Certificate

About V-Ex Tech….

V-Ex Tech is an elevated education platform providing rigorous industry-relevant programs Designed and delivered on collaboration with industry professionals. It has been constantly Into process of creating an immersive learning experience binding latest technologies, pedagogy and services with enormous job placement opportunities too.

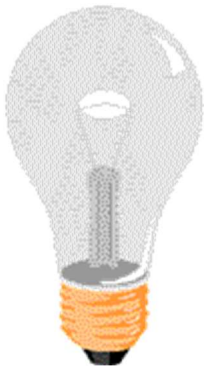
V-Ex Tech

JavaScript Introduction

JavaScript Can Change HTML Content

One of many JavaScript HTML methods is `getElementById()`.

```
document.getElementById("demo").innerHTML = "Hello JavaScript";
```



Turn on the light Turn off the light

JavaScript Can Change HTML Styles (CSS)

```
document.getElementById("demo").style.fontSize = "35px";
```

External JavaScript

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

Internal JavaScript

Inline JavaScript

JavaScript Display Possibilities

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

Using document.write()

```
<script>
document.write(5 + 6);
</script>
```

Using window.alert()

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

Using console.log()

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

JavaScript Statements

```
<body>



        <p id="demo1"></p>

        <script>
        let a, b, c;
        a = 5; b = 6; c = a + b;
        document.getElementById("demo1").innerHTML = c;
        </script>

        </body>
```

Function

```
<body>


        <button type="button" onclick="myFunction()">Click Me!</button>

        <p id="demo1"></p>
        <p id="demo2"></p>

        <script>
        function myFunction() {
          document.getElementById("demo1").innerHTML = "Hello Dolly!";
          document.getElementById("demo2").innerHTML = "How are you?";
        }
        </script>

        </body>
```

String concate

```
<body>

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

        <script>
        document.getElementById("demo").innerHTML = "John" + " "  + "Doe";
        </script>

        </body>
```

JavaScript Variables

- Automatically

- Using `var`
- Using `let`
- Using `const`

```
<body>


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

        <script>
        x = 5;
        y = 6;
        z = x + y;
        document.getElementById("demo").innerHTML =
        "The value of z is: " + z;
        </script>

        </body>
```

V-Ex Tech

```
<body>


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

        <script>
        var x = 5;
        var y = 6;
        var z = x + y;
        document.getElementById("demo").innerHTML =
        "The value of z is: " + z;
        </script>

        </body>
```

```
<body>


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

        <script>
        let x = 5;
        let y = 6;
        let z = x + y;
        document.getElementById("demo").innerHTML =
        "The value of z is: " + z;
        </script>

        </body>
```

```
<body>

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

        <script>
        let carName;
        document.getElementById("demo").innerHTML = carName;
        </script>

        </body>
```

```
<body>


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

        <script>
        const pi = 3.14;
        let person = "John Doe";
        let answer = 'Yes I am!';

        document.getElementById("demo").innerHTML =
        pi + "<br>" + person + "<br>" + answer;
        </script>

        </body>
```

V-Ex Tech

```
<body>

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

        <script>
        let x = "5" + 2 + 3;
        document.getElementById("demo").innerHTML = x;
        </script>

        </body>
```

```
 <body>

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

        <script>
        var x = 2;
        // Now x is 2

        var x = 3;
        // Now x is 3

        document.getElementById("demo").innerHTML = x;
        </script>

        </body>
```

JavaScript Operators

Types of JavaScript Operators

There are different types of JavaScript operators:

- Arithmetic Operators
- Assignment Operators
- Comparison Operators
- String Operators
- Logical Operators
- Bitwise Operators
- Ternary Operators
- Type Operators

JavaScript Arithmetic Operators

Operator	Description
+	Addition
-	Subtraction

*	Multiplication
**	Exponentiation (ES2016)
/	Division
%	Modulus (Division Remainder)
++	Increment
--	Decrement

JavaScript Assignment Operators

Operator	Example	Same As
=	x = y	x = y

+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
**=	x **= y	x = x ** y

JavaScript Comparison Operators

Operator	Description
==	equal to

===	equal value and equal type
!=	not equal
!==	not equal value or not equal type
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
?	ternary operator

JavaScript Logical Operators

Operator	Description
&&	logical and
||	logical or
!	logical not

JavaScript Arithmetic Operators

```
<body>


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

   <script>
   let a = 3;
   let x = (100 + 50) * a;
   document.getElementById("demo").innerHTML = x;
   </script>

   </body>
```

```
<body>


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

       <script>
       let x = 5;
       let y = 2;
       let z = x % y;
       document.getElementById("demo").innerHTML = z;
       </script>

       </body>
```

Asighnment Operators

```
<body>


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

        <script>
        let text = "Hello";
        text += " World";
        document.getElementById("demo").innerHTML = text;
        </script>

        </body>
```

JavaScript Data Types

1String
2. Number

3.Float

```
 <body>
```

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

<script>
let x = 5;
let y = 5;
let z = 6;

document.getElementById("demo").innerHTML =
(x == y) + "<br>" + (x == z);
</script>

</body>
```

The typeof Operator

```
typeof 0
typeof 314
typeof 3.14
```

JavaScript Functions

```
<body>
```

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

<script>
function myFunction(p1, p2) {
  return p1 * p2;
}

let result = myFunction(4, 3);
document.getElementById("demo").innerHTML = result;
</script>

</body>
```

```
<body>


<p id="demo1"></p>
<p id="demo2"></p>

<script>
let text = "Outside: " + typeof carName;
document.getElementById("demo1").innerHTML = text;

function myFunction() {
  let carName = "Volvo";
  let text = "Inside: " + typeof carName + " " + carName;
  document.getElementById("demo2").innerHTML = text;
}

myFunction();
</script>

</body>
```

V-Ex Tech

JavaScript Objects

Objects, Properties, and Methods

A car has **properties** like weight and color, and **methods** like start and stop:

Object	Properties	Methods
	car.name = Fiat	car.start()
	car.model = 500	car.drive()
	car.weight = 850kg	car.brake()
	car.color = white	car.stop()

```
<body>



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

  <script>

  const car = {type:"Fiat", model:"500", color:"white"};


  document.getElementById("demo").innerHTML = "The car type is
" + car.type;
  </script>

  </body>
```

```
<body>

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

  <script>

  const person = {
    firstName: "John",
    lastName : "Doe",
    id     :  5566
  };


  document.getElementById("demo").innerHTML =
  person["firstName"] + " " + person["lastName"];
  </script>

  </body>
```

The this Keyword

In other words, `this.firstName` means the `firstName` property of **this object**.

```
<body>


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

  <script>

  const person = {
    firstName: "John",
    lastName: "Doe",
    id: 5566,
    fullName: function() {
      return this.firstName + " " + this.lastName;
    }
  };


  document.getElementById("demo").innerHTML =
person.fullName();
  </script>

  </body>
```

practice

Student Model

V-Ex Tech

JavaScript Events

```
<body>


  <button onclick="displayDate()">The time is?</button>

  <script>
  function displayDate() {
    document.getElementById("demo").innerHTML = Date();
  }
  </script>

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

  </body>
```

Event	Description
onchange	An HTML element has been changed
onclick	The user clicks an HTML element
onmouseover	The user moves the mouse over an HTML element
onmouseout	The user moves the mouse away from an HTML element
onkeydown	The user pushes a keyboard key
onload	The browser has finished loading the page

The list is much longer: W3Schools JavaScript Reference HTML DOM Events.

JavaScript Strings

```
<body>


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

  <script>
  let answer1 = "It's alright";
  let answer2 = "He is called 'Johnny'";
  let answer3 = 'He is called "Johnny"';

  document.getElementById("demo").innerHTML =
  answer1 + "<br>" + answer2 + "<br>" + answer3;
  </script>

  </body>
```

String Length

```
<body>


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

  <script>
  let text = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
  document.getElementById("demo").innerHTML = text.length;
  </script>

  </body>
```

Line breack

```
<body>

  <h1>JavaScript Strings</h1>

  <p>
  You can break a code line within a text string with a
backslash.
  </p>

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

  <script>
  document.getElementById("demo").innerHTML = "Hello \
  Dolly!";
  </script>

  </body>
```

V-Ex Tech

JavaScript String Methods

String length

String slice()

String substring()

String substr()

String replace()

String replaceAll()

String toUpperCase()

String toLowerCase()

String concat()

String trim()

String trimStart()

String trimEnd()

String padStart()

String padEnd()

String charAt()

String charCodeAt()

String split()

JavaScript String Length

```
<body>

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

    <script>
    let text = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    document.getElementById("demo").innerHTML = text.length;
    </script>

    </body>
```

Extracting String Parts

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

JavaScript String slice()

```
<body>

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

    <script>
    let text = "Apple, Banana, Kiwi";
    let part = text.slice(7,13);
    document.getElementById("demo").innerHTML = part;
    </script>

    </body>
```

```
<body>
```

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

    <script>
    let text = "Apple, Banana, Kiwi";
    let part = text.slice(7)
    document.getElementById("demo").innerHTML = part;
    </script>

    </body>
```

```
<body>

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

    <script>
    let text = "Apple, Banana, Kiwi";
    let part = text.slice(-12,-6)
    document.getElementById("demo").innerHTML = part;
    </script>

    </body>
```

JavaScript String substring()

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

    <script>
    let str = "Apple, Banana, Kiwi";
    document.getElementById("demo").innerHTML = str.substring(7,13);
    </script>

    </body>
```

JavaScript String substr()

```
<body>

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

    <script>
    let str = "Apple, Banana, Kiwi";
    document.getElementById("demo").innerHTML = str.substr(7,6);
    </script>

    </body>
```

Replacing String Content

```
<body>
    <button onclick="myFunction()">Try it</button>

    <p id="demo">Please visit Microsoft!</p>

    <script>
    function myFunction() {
      let text = document.getElementById("demo").innerHTML;
      document.getElementById("demo").innerHTML =
      text.replace("Microsoft","W3Schools");
    }
    </script>

    </body>
```

JavaScript String ReplaceAll()

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

    <script>
    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");

    document.getElementById("demo").innerHTML = text;
    </script>

    </body>
```

JavaScript String toUpperCase()

```
<body>

    <button onclick="myFunction()">Try it</button>

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

    <script>
    function myFunction() {
      let text = document.getElementById("demo").innerHTML;
      document.getElementById("demo").innerHTML =
      text.toUpperCase();
    }
    </script>

    </body>
```

JavaScript String toLowerCase()

```
<body>
    <button onclick="myFunction()">Try it</button>

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

    <script>
    function myFunction() {
      let text = document.getElementById("demo").innerHTML;
      document.getElementById("demo").innerHTML =
      text.toLowerCase();
    }
    </script>

    </body>
```

String concat()

```
<body>
    <body>


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

        <script>
        let text1 = "Hello";
        let text2 = "World!";
        let text3 = text1.concat(" ",text2);
        document.getElementById("demo").innerHTML = text3;
        </script>

        </body>
</body>
```

String trim()

```
<body>

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

    <script>
    let text1 = "     Hello World!     ";
    let text2 = text1.trim();

    document.getElementById("demo").innerHTML =
    "Length text1 = " + text1.length + "<br>Length text2 = " +
text2.length;
    </script>

    </body>
```

String trimStart()

```
<body>

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

    <script>
    let text1 = "     Hello World!     ";
    let text2 = text1.trimStart();

    document.getElementById("demo").innerHTML =
    "Length text1 = " + text1.length + "<br>Length text2 = " +
text2.length;
    </script>

    </body>
```

String trimEnd()

```
<body>


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

    <script>
    let text1 = "     Hello World!     ";
    let text2 = text1.trimEnd();

    document.getElementById("demo").innerHTML =
    "Length text1 = " + text1.length + "<br>Length text2 = " +
text2.length;
    </script>

    </body>
```

String padStart()

```
<body>


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

    <script>
    let text = "5";
    text = text.padStart(4,"0");

    document.getElementById("demo").innerHTML = text;
    </script>

    </body>
```

String padEnd()

```
<body>


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

    <script>
    let text = "5";
    text = text.padEnd(4,"0");

    document.getElementById("demo").innerHTML = text;
    </script>

    </body>
```

Extracting String Characters

There are 3 methods for extracting string characters:

- `charAt(position)`
- `charCodeAt(position)`
- Property access [ ]

String charAt()

```
<body>


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

    <script>
    var text = "HELLO WORLD";
    document.getElementById("demo").innerHTML = text.charAt(0);
    </script>

    </body>
```

String charCodeAt()

```
a = 97,b = 98,c = 99,d = 100,e = 101,f = 102,g = 103,h = 104,i
= 105,j = 106,k = 107l = 108
m = 109,n = 110,o = 111,p = 112,q = 113,r = 114,s = 115,t =
116,u = 117,v = 118,w = 119
x = 120,y = 121,z=122
```

```
A = 6,B = 66,C = 67,D = 68,E = 69,F = 70,G = 71,H = 72,I = 73,J
= 74,
K = 75,L = 76,M = 77,N = 78,O = 79,P = 80,Q = 81,R = 82,S =
83,T = 84,
U = 85,V = 86,W = 87,X = 88,Y = 89,Z = 90
```

```
<body>


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

    <script>
    let text = "AELLO WORLD";
    document.getElementById("demo").innerHTML =
text.charCodeAt(0);
    </script>

    </body>
```

Property Access

```
<body>

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

    <script>
    let text = "HELLO WORLD";
    document.getElementById("demo").innerHTML = text[0];
    </script>
    </body>
```

Converting a String to an Array

String split()

```
<body>

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

    <script>
    let text = "a,b,c,d,e,f";
    const myArray = text.split(",");
    document.getElementById("demo").innerHTML = myArray[0];
    </script>

    </body>
```

String Search

String Search Methods

- String indexOf()
- String lastIndexOf()
- String search()
- String match()
- String matchAll()
- String includes()
- String startsWith()
- String endsWith()

String indexOf()

```
<body>

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

    <script>
    let text = "Please locate where 'locate' occurs!";
    let index = text.indexOf("locate");
    document.getElementById("demo").innerHTML = index;
    </script>

    </body>
```

String lastIndexOf()

```
<body>


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

    <script>
    let text = "Please locate where 'locate' occurs!";
    let index = text.lastIndexOf("locate");
    document.getElementById("demo").innerHTML = index;
    </script>

    </body>
```

```
<body>

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

    <script>
    let text = "Please locate where 'locate' occurs!";
    let index = text.indexOf("locate",15);
    document.getElementById("demo").innerHTML = index;
    </script>

    </body>
```

String search()

```
<body>

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

    <script>
    let text = "Please locate where 'locate' occurs!";
    let index = text.search("locate");
    document.getElementById("demo").innerHTML = index;
    </script>

    </body>
```

- The `search()` method cannot take a second start position argument.
- The `indexOf()` method cannot take powerful search values (regular expressions).

String match()

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

    <script>
    let text = "The rain in SPAIN stays mainly in the plain";
    const myArr = text.match("ain");
    document.getElementById("demo").innerHTML = myArr.length +
" " + myArr;
    </script>

    </body>
```

```
<body>


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

    <script>
    let text = "The rain in SPAIN stays mainly in the plain";
    const myArr = text.match(/ain/g);
    document.getElementById("demo").innerHTML = myArr.length +
" " + myArr;
    </script>

    </body>
```

```
<body>

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

    <script>
    let text = "The rain in SPAIN stays mainly in the plain";
    const myArr = text.match(/ain/gi);
    document.getElementById("demo").innerHTML = myArr.length +
" " + myArr;
    </script>

    </body>
```

String includes()

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

Otherwise it returns `false`.

```
<body>

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



    <script>
    let text = "Hello world, welcome to the universe.";
    document.getElementById("demo").innerHTML =
text.includes("world");
    </script>

    </body>
```

```
<body>

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

    <script>
    let text = "Hello world, welcome to the universe.";
    document.getElementById("demo").innerHTML =
text.includes("world", 12);
    </script>

    </body>
```

String startsWith()

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

    <p>The endsWith() method is not supported in Internet
Explorer.</p>

    <script>
    let text = "John Doe";
    document.getElementById("demo").innerHTML =
text.endsWith("Doe");
    </script>

    </body>
```

Back-Tics Syntax

```
<body>

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

        <p>Template literals are not supported in Internet
Explorer.</p>

        <script>
        let text = `Hello world!`;
        document.getElementById("demo").innerHTML = text;
        </script>

        </body>
```

```
<body>


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

        <p>Template literals are not supported in Internet
Explorer.</p>

        <script>
        let text = `He's often called "Johnny"`;
        document.getElementById("demo").innerHTML = text;
        </script>

        </body>
```

Multiline Strings

```
  <body>


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

        <p>Template literals are not supported in Internet
Explorer.</p>

        <script>
        let firstName = "John";
        let lastName = "Doe";

        let text = `Welcome ${firstName}, ${lastName}!`;

        document.getElementById("demo").innerHTML = text;
        </script>

        </body>
```

JavaScript Numbers

NaN - Not a Number

`NaN` is a JavaScript reserved word indicating that a number is not a legal number.

Trying to do arithmetic with a non-numeric string will result in `NaN` (Not a Number):

```
<body>


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

    <script>
    document.getElementById("demo").innerHTML = 100 / "Apple";
    </script>

    </body>
```

NaN - Not a Number

```
<body>

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

    <script>
    document.getElementById("demo").innerHTML = 100 / "10";
    </script>

    </body>
```

```
<body>

    <h2>JavaScript Numbers</h2>

    <p>The typeof NaN is number:</p>

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

    <script>
    let x = NaN;
    document.getElementById("demo").innerHTML = typeof x;
    </script>

    </body>
```

Infinity

```
<body>


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

    <script>
    let x = 2/0;
    let y = -2/0;
    document.getElementById("demo").innerHTML = x + "<br>" + y;
    </script>

    </body>
```

```
<body>


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

<script>
// x is a number
let x = 500;

// y is an object
let y = new Number(500);
document.getElementById("demo").innerHTML = (x==y);
</script>

</body>
```

```
<body>


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

<script>
// x is a number
let x = 500;

// y is an object
let y = new Number(500);

document.getElementById("demo").innerHTML = (x===y);
</script>

</body>
```

JavaScript BigInt

JavaScript Integer Accuracy

JavaScript integers are only accurate up to 15 digits:

```
<body>

<h1>JavaScript Numbers</h1>
<h2>Integer Precision</h2>

<p>Integers (numbers without a period or exponent notation) are accurate up to 15
digits:</p>

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

<script>
let x = 999999999999999;
let y = 9999999999999999;
document.getElementById("demo").innerHTML = x + "<br>" + y;
</script>

</body>
```

How to create big int

```
<body>

<h1>JavaScript Numbers</h1>
<h2>Integer and BigInt</h2>

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

<script>
let x = 9999999999999999;
let y = BigInt("9999999999999999");
document.getElementById("demo").innerHTML = x + "<br>" + y;
</script>

</body>
```

```
<body>

<h1>JavaScript Numbers</h1>
<h2>BigInt typeof</h2>

<p>The typeof a BigInt is:</p>
<p id="demo"></p>

<script>
let x = BigInt("9999999999999999");
document.getElementById("demo").innerHTML = typeof x;
</script>

</body>
```

JavaScript Number Methods

The valueOf() Method

```
<body>

<h2>JavaScript Number Methods</h2>

<p>The valueOf() method returns a number as a number:</p>

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

<script>
let x = 123;

document.getElementById("demo").innerHTML =
  x.valueOf() + "<br>" +
  (123).valueOf() + "<br>" +
  (100 + 23).valueOf();
</script>

</body>
```

JavaScript Arrays

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Arrays</h1>

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

<script>
const cars = ["Saab", "Volvo", "BMW"];
document.getElementById("demo").innerHTML = cars;
</script>

</body>
</html>
```

Creating an Array

Using an array literal is the easiest way to create a JavaScript Array.

Syntax:

```
const array_name = [item1, item2, ...];
```

```
<body>
<h1>JavaScript Arrays</h1>

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

<script>
const cars = [
  "Saab",
  "Volvo",
  "BMW"
];
document.getElementById("demo").innerHTML = cars;
</script>

</body>
```

```
<body>
<h1>JavaScript Arrays</h1>

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

<script>
const cars = [];
cars[0]= "Saab";
cars[1]= "Volvo";
cars[2]= "BMW";
document.getElementById("demo").innerHTML = cars;
</script>

</body>
```

Accessing Array Elements

```
<body>
<h1>JavaScript Arrays</h1>
<h2>Bracket Indexing</h2>

<p>JavaScript array elements are accessed using numeric indexes (starting from
0).</p>

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

<script>
const cars = ["Saab", "Volvo", "BMW"];
cars[0] = "Opel";
document.getElementById("demo").innerHTML = cars;
</script>

</body>
```

Converting arrey to string

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Arrays</h1>
<h2>The toString() Method</h2>



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

<script>
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo").innerHTML = fruits.toString();
</script>

</body>
</html>
```

```
<body>



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

<script>
const person = ["John", "Doe", 46];
document.getElementById("demo").innerHTML = person[0];
</script>

</body>
```

The length Property

```
<body>


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

<script>
const fruits = ["Banana", "Orange", "Apple", "Mango"];
let size = fruits.length;
document.getElementById("demo").innerHTML = size;
</script>

</body>
```

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


<button onclick="myFunction()">Try it</button>

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

<script>
const fruits = ["Banana", "Orange", "Apple"];
document.getElementById("demo").innerHTML = fruits;

function myFunction() {
  fruits.push("Lemon");
  document.getElementById("demo").innerHTML = fruits;
}
</script>

</body>
</html>
```

The Difference Between Arrays and Objects

In JavaScript, **arrays** use **numbered indexes**.

In JavaScript, **objects** use **named indexes**.

When to Use Arrays. When to use Objects.

- JavaScript does not support associative arrays.
- You should use **objects** when you want the element names to be **strings (text)**.
- You should use **arrays** when you want the element names to be **numbers**

```
const points = new Array();
const points = [];
```

```
<body>


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

<script>
const points = new Array(40, 100, 1, 5, 25, 10);
const points = [40, 100, 1, 5, 25, 10];
document.getElementById("demo").innerHTML = points[0];
</script>

</body>
```

How to Recognize an Array

```
<body>


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

<script>
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo").innerHTML = typeof fruits;
</script>

</body>
```

```
<body>
<h1>JavaScript Arrays</h1>
<h2>The isArray() Method</h2>

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

<script>
const fruits = ["Banana", "Orange", "Apple"];
document.getElementById("demo").innerHTML = Array.isArray(fruits);
</script>

</body>
```

Array Methods

Array length
Array toString()
Array pop()
Array push()
Array shift()
Array unshift()
Array join()
Array delete()
Array concat()
Array flat()
Array splice()
Array slice()

The methods are listed in the order they appear in this tutorial page

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Arrays</h1>


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

<script>
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo").innerHTML = fruits.toString();
</script>
</body>
</html>
```

```html
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Arrays</h1>
<h2>The join() Method</h2>

<p>The join() method joins array elements into a string.</p>
<p>It this example we have used " * " as a separator between the elements:</p>

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

<script>
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo").innerHTML = fruits.join(" * ");
</script>

</body>
</html>
```

Popping and Pushing

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


<p id="demo1"></p>
<p id="demo2"></p>

<script>
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo1").innerHTML = fruits;
fruits.pop();
document.getElementById("demo2").innerHTML = fruits;
</script>

</body>
</html>
```

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


<p id="demo1"></p>
<p id="demo2"></p>

<script>
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo1").innerHTML = fruits;
fruits.push("Kiwi");
document.getElementById("demo2").innerHTML = fruits;
</script>

</body>
</html>
```

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

<p id="demo1"></p>
<p id="demo2"></p>

<script>
const fruits = ["Banana", "Orange", "Apple", "Mango"];

document.getElementById("demo1").innerHTML =
"The first fruit is: " + fruits[0];

delete fruits[0];

document.getElementById("demo2").innerHTML =
"The first fruit is: " + fruits[0];
</script>

</body>
</html>
```

Splicing and Slicing Arrays

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


<p id="demo1"></p>
<p id="demo2"></p>

<script>
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo1").innerHTML = fruits;

fruits.splice(2, 0, "Lemon", "Kiwi");
document.getElementById("demo2").innerHTML = fruits;
</script>

</body>
</html>
```

```
<!DOCTYPE html>
<html>
<body>
</p>

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

<script>
const fruits = ["Banana", "Orange", "Lemon", "Apple", "Mango"];
const citrus = fruits.slice(1);
document.getElementById("demo").innerHTML = fruits + "<br><br>" + citrus;
</script>

</body>
</html>
```

Sorting Arrays

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


<p id="demo1"></p>
<p id="demo2"></p>

<script>
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo1").innerHTML = fruits;

fruits.sort();
document.getElementById("demo2").innerHTML = fruits;
</script>

</body>
</html>
```

fruits.reverse();

V-Ex Tech

Array Iteration

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


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

<script>
const numbers = [45, 4, 9, 16, 25];

let txt = "";
numbers.forEach(myFunction);
document.getElementById("demo").innerHTML = txt;

function myFunction(value, index, array) {
  txt += value + "<br>";
}
</script>

</body>
</html>
```

V-Ex Tech

Map method

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


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

<script>
const numbers1 = [45, 4, 9, 16, 25];
const numbers2 = numbers1.map(myFunction);

document.getElementById("demo").innerHTML = numbers2;

function myFunction(value, index, array) {
  return value * 2;
}
</script>

</body>
</html>
```

Filter method

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


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

<script>
const numbers = [45, 4, 9, 16, 25];
const over18 = numbers.filter(myFunction);

document.getElementById("demo").innerHTML = over18;

function myFunction(value, index, array) {
  return value > 18;
}
</script>

</body>
</html>
```

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


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

<script>
const fruits = ["Apple", "Orange", "Apple", "Mango"];
let position = fruits.indexOf("Apple") + 1;

document.getElementById("demo").innerHTML = "Apple is found in position " +
position;
</script>

</body>
</html>
```

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


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

<p><strong>Note:</strong> The includes method is not supported in Edge 13 (and
earlier versions).</p>

<script>
const fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo").innerHTML = fruits.includes("Mango");
</script>

</body>
</html>
```

JavaScript Array Spread (...)

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


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

<script>
const q1 = ["Jan", "Feb", "Mar"];
const q2 = ["Apr", "May", "Jun"];
const q3 = ["Jul", "Aug", "Sep"];
const q4 = ["Oct", "Nov", "May"];

const year = [...q1, ...q2, ...q3, ...q4];
document.getElementById("demo").innerHTML = year;
</script>

</body>
</html>
```

Date function

JavaScript Date Input

There are generally 3 types of JavaScript date input formats:

Type	Example
ISO Date	"2015-03-25" (The International Standard)
Short Date	"03/25/2015"
Long Date	"Mar 25 2015" or "25 Mar 2015"

Get Date Methods

getFullYear()	Get **year** as a four digit number (yyyy)
getMonth()	Get **month** as a number (0-11)
getDate()	Get **day** as a number (1-31)
getDay()	Get **weekday** as a number (0-6)
getHours()	Get **hour** (0-23)
getMinutes()	Get **minute** (0-59)
getSeconds()	Get **second** (0-59)
getMilliseconds()	Get **millisecond** (0-999)
getTime()	Get **time** (milliseconds since January 1, 1970)

Set Date Methods

setDate()	Set the day as a number (1-31)
setFullYear()	Set the year (optionally month and day)
setHours()	Set the hour (0-23)
setMilliseconds()	Set the milliseconds (0-999)
setMinutes()	Set the minutes (0-59)
setMonth()	Set the month (0-11)
setSeconds()	Set the seconds (0-59)
setTime()	Set the time (milliseconds since January 1, 1970)