# **JavaScript Iterables:**

Iterables are iterable objects (like Arrays).

Iterables can be accessed with simple and efficient code.

Iterables can be iterated over with for..of loops

**Iterating:**

Iterating is easy to understand.

It simply means looping over a sequence of elements.

Here are some easy examples:

* Iterating over a String
* Iterating over an Array

# **Iterate Through JSON Objects in JavaScript:**

**JSON stands for JavaScript Object Notation.**

**It’s a light format for storing and transferring data from one place to another.**

**So in looping, it is one of the most commonly used techniques for transporting data that is the array format or in attribute values.**

**JavaScript Loops:**

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

Loops are handy, if you want to run the same code over and

over again,each time with a different value.

Often this is the case when working with arrays:

**Different Kinds of Loops:**

* for - loops through a block of code a number of times
* for/in - loops through the properties of an object
* for/of - loops through the values of an iterable object
* while - loops through a block of code while a specified condition is true
* do/while - also loops through a block of code while a specified condition is true

## The For Loop:

The for statement creates a loop with 3 optional expressions:

**Syntax:**

for (*expression 1*;*expression 2*;*expression 3*) {  
  // *code block to be executed*  
}

**Expression 1** is executed (one time) before the execution of the code block.

**Expression 2** defines the condition for executing the code block.

**Expression 3** is executed (every time) after the code block has been executed.

### **Example**

<!DOCTYPE html>

<html>

<body>

<h2>JavaScript For Loop</h2>

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

<script>

let text = "";

for (let i = 0; i < 5; i++) {

text += "The number is " + i + "<br>";

}

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

</script>

</body>

</html>

## The For In Loop

The JavaScript for in statement loops through the properties of an Object:

### **Syntax**

for (key in object) {  
  // *code block to be executed*  
}

### **Example:**

<!DOCTYPE html>

<html>

<body>

<h2>JavaScript For In Loop</h2>

<p>The for in statement loops through the properties of an object:</p>

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

<script>

const person = {fname:"John", lname:"Doe", age:25};

let txt = "";

**for (let x in person**) {

txt += person[x] + " ";

}

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

</script>

</body>

</html>

## The For Of Loop

The JavaScript for of statement loops through the values of an iterable object.

It lets you loop over iterable data structures such as Arrays, Strings, Maps, NodeLists, and more:

### **Syntax**

for (variable of iterable) {  
  // *code block to be executed*  
}

**variable** - For every iteration the value of the next property is assigned to the variable. Variable can be declared with const, let, or var.

**iterable** - An object that has iterable properties.

### **Example:**

<!DOCTYPE html>

<html>

<body>

<h2>JavaScript For Of Loop</h2>

<p>The for of statement loops through the values of any iterable object:</p>

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

<script>

const cars = ["BMW", "Volvo", "Mini"];

let text = "";

**for (let x of cars)** {

text += x + "<br>";

}

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

</script>

</body>

</html>

# **The forEach() Loop:**

The forEach() method calls a function for each element in an array.

The forEach() method is not executed for empty elements.

**Syntax:**

*array*.forEach(*function(currentValue, index, arr), thisValue*)

### **Example:**

<!DOCTYPE html>

<html>

<body>

<h1>JavaScript Arrays</h1>

<h2>The forEach() Method</h2>

<p>forEach() calls a function for each element in an array:</p>

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

<script>

let text = "";

const fruits = ["apple", "orange", "cherry"];

**fruits.forEach(myFunction);**

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

function myFunction(item, index) {

text += index + ": " + item + "<br>";

}

</script>

</body>

</html>